

UNIVERSIDADE PAULISTA

EMILY CAMILA DE SOUSA FERREIRA - G768EE3

HENRIQUE CARVALHO DOS SANTOS - G461720

ISAC FERREIRA DE SOUZA - G750381

LEANDRO MORAES DE SOUZA - G7862D1

MARCOS EDUARDO FERNANDES - T264AB1

THAIS REIS DA SILVA - T917AG6

EXPLORANDO MARTE COM TECNOLOGIA

SOROCABA

2024

EMILY CAMILA DE SOUSA FERREIRA - G768EE3
HENRIQUE CARVALHO DOS SANTOS - G461720
ISAC FERREIRA DE SOUZA - G750381
LEANDRO MORAES DE SOUZA - G7862D1
MARCOS EDUARDO FERNANDES - T264AB1
THAIS REIS DA SILVA - T917AG6

Projeto Integrado Multidisciplinar (PIM) para o
curso de Análise e Desenvolvimento de Sistemas
apresentado ao Centro Universitário Paulista –
UNIP de Sorocaba, como forma de avaliação
parcial.
Orientador: Prof. Me. Waldir Silva

SOROCABA
2024

EMILY CAMILA DE SOUSA FERREIRA - G768EE3
HENRIQUE CARVALHO DOS SANTOS - G461720
ISAC FERREIRA DE SOUZA - G750381
LEANDRO MORAES DE SOUZA - G7862D1
MARCOS EDUARDO FERNANDES - T264AB1
THAIS REIS DA SILVA - T917AG6

EXPLORANDO MARTE COM TECNOLOGIA

Projeto Integrado Multidisciplinar (PIM) para o curso de Análise e Desenvolvimento de Sistemas apresentado ao Centro Universitário Paulista – UNIP de Sorocaba, como forma de avaliação parcial.

Aprovado em:

BANCA EXAMINADORA

Prof. Me. Waldir Silva
Universidade Paulista – UNIP

_____ / ____ / ____

Prof. Esp. Samuel Rodrigues
Universidade Paulista – UNIP

_____ / ____ / ____

Prof. Esp. Alex Sampaio
Universidade Paulista – UNIP

_____ / ____ / ____

SOROCABA

2024

RESUMO

O presente trabalho propõe uma análise do uso da tecnologia em espaços culturais como museus multitemáticos. O Museu da Evolução e Conhecimento Humano (MECH) busca por meio do desenvolvimento de totens e sites aumentar a interatividade, experiência e conhecimento dos visitantes sobre o planeta vermelho Marte, expedições no passado e possíveis viagens futuras do homem a fim de novos conhecimentos científicos. A proposta consiste em desenvolver um totem onde o usuário tenha acesso as obras dos museus com descrições detalhadas, mapa do museu e um questionário interativo onde possa testar os conhecimentos adquiridos através das descrições das obras e vivência no museu. No site no museu, o usuário terá acesso a um login onde de forma rápida e prática realizará a compra de ingressos, visualização de obras e outras informações sobre o MECH. O trabalho detalha o processo de desenvolvimento do software para o totem interativo e do site desde a definição dos objetivos e funcionalidades até a escolha dos recursos tecnológicos mais adequados.

Palavras-Chave: tecnologia, interatividade, software, site, totem, API.

ABSTRACT

This work proposes an analysis of the use of technology in cultural spaces such as multi-theme museums. The Museum of Evolution and Human Knowledge (MECH) aims to enhance visitor interactivity, experience, and understanding of the red planet Mars, past expeditions, and potential future journeys of humankind towards new scientific knowledge through the development of totems and websites. The proposal consists of creating a totem that provides users access to the museum's exhibits with detailed descriptions, a map of the museum, and an interactive questionnaire to test the knowledge acquired through the descriptions of the exhibits and experiences in the museum. On the museum's website, users will have access to a login for quick and easy ticket purchases, exhibit views, and other information about MECH. This work details the software development process for the interactive totem and the website, from defining objectives and functionalities to selecting the most suitable technological resources.

Keywords: technology, interactivity, software, website, totem, API.

LISTA DE ILUSTRAÇÕES

Figura 1: Diagrama - Máquina de Estados.....	19
Figura 2: Diagrama - Diagrama de Comunicação	20
Figura 3: TAP - Parte 1	24
Figura 4: TAP - Parte 2	25
Figura 5: TAP - Parte 3	26
Figura 6: TAP - Parte 4	27
Figura 7: Princípios da qualidade de software da norma ISO 9126	33
Figura 8:Diagrama - Modelo de Dados.....	40
Figura 9: Protótipo Sistema Desktop da Tela Inicial	42
Figura 10: Protótipo Sistema Desktop da Exposição	42
Figura 11: Protótipo Sistema Desktop tela Questionario.....	43
Figura 12: Protótipo Sistema Desktop tela Avaliação Notas	43
Figura 13: Protótipo Sistema Desktop tela Avaliação.....	44
Figura 14: Protótipo Sistema Desktop da Tela de Agradecimento	44
Figura 15: Protótipo Sistema Desktop da Tela do Mapa	45
Figura 16: Protótipo Site Menu	46
Figura 17: Protótipo Site da Tela de Compra.....	46
Figura 18: Protótipo Site Tela de Pagamento	47
Figura 19: Protótipo Site Tela de Login.....	48
Figura 20: Protótipo Site Tela Exposição	49
Figura 21: Protótipo Site Tela Questionario	49
Figura 22: API - NovaVenda Json	52
Figura 23: API - NovaVenda Retorno	53
Figura 24: API - RealizarLogin Endpoint.....	54
Figura 25: API - RealizarLogin Sucesso	54
Figura 26: API - RealizarLogin Negativo	55
Figura 27: API - Funções GET	57
Figura 28: Totem - Configuração	59
Figura 29: Totem - Menu Principal	60
Figura 30: Totem - Tela do Mapa	61
Figura 31: Totem - Lista de Exposições.....	61
Figura 32: Totem - Descrição da obra	62
Figura 33: Totem - Questionário.....	63
Figura 34: Totem - Avaliação Notas	64
Figura 35: Totem - Avaliação Sugestão	65
Figura 36: Totem - Respostas do Usuário	66
Figura 37: Totem - Estatística Global	66
Figura 38: Totem - Tela de Agradecimento.....	67
Figura 39: Exemplo - Serialização de Objeto	68
Figura 40: Site - Página Inicial.....	70
Figura 41: Site - Página de Informações	71
Figura 42: Site - Página de Compras	72
Figura 43: Site - Página de Login.....	73
Figura 44: Site - Página de Exposições.....	74
Figura 45: Site - Página do Questionario	75
Figura 46: Site - Página de Exposições - Responsividade	77

Figura 47: Site - Página do Questionario - Responsividade	78
Figura 48: Site - Página Principal - Responsividade	79
Figura 49: Exposição - O Planeta Vermelho.....	85
Figura 50: Exposição - Exploração e Potencial para Vida	86
Figura 51: Exposição - O Terreno Marciano.....	87
Figura 52: Exposição - Água em Marte.....	88
Figura 53: Exposição - Valles Marineris	89
Figura 54: Exposição - O Monte Olimpo.....	90
Figura 55: Exposição - Impacto de Asteroides.....	91
Figura 56: Exposição - Colonização de Marte.....	92
Figura 57: Diagrama - Diagrama de Classe API.....	93
Figura 58: Diagrama – Diagrama de Classe WPF	94
Figura 59: Diagrama - Diagrama de Sequência	95

LISTA DE TABELAS

Tabela 1:Orçamento	28
Tabela 2: RACI.....	29
Tabela 3: Cronograma - Parte 1	30
Tabela 4: Cronograma - Parte 2	30
Tabela 5: Cronograma - Parte 3	31

SUMARIO

1.	INTRODUÇÃO	11
1.1	Objetivo Geral	12
1.2	Objetivos Específicos	12
2.	PESQUISA BIBLIOGRÁFICA.....	13
2.1	Marte.....	13
2.2	Modelos de Dados: Conceitual, Lógico e Físico	14
2.2.1	Modelo de Dados Conceitual	14
2.2.2	Modelo de Dados Lógico.....	15
2.2.3	Modelo de Dados Físico	15
2.3	Diagrama de Pacotes.....	16
2.4	Diagrama de Classes.....	17
2.5	Diagrama de Sequência.....	18
2.6	Diagrama de Máquina de Estados.....	18
2.7	Diagrama de Comunicação.....	19
3.	CONTEXTUALIZAÇÃO DO PROJETO.....	21
4.	PLANEJAMENTO	23
4.1	Ciclo de Vida	23
4.2	Termo de Abertura do Projeto (TAP)	23
4.3	Orçamento do Projeto	27
4.4	Responsabilidade dos membros da equipe	28
4.5	Cronograma.....	29
4.6	Desenvolvimento.....	31
4.7	Qualidade de Software	31
4.8	Requisitos Funcionais	33
4.9	Requisitos Não Funcionais.....	35
4.9.1	Método Ágil.....	35
4.9.2	User Interface e User Experience	35
4.9.3	Implementação do Dispositivo.....	36
4.9.4	Ferramentas para equipe	37
4.9.5	Ferramentas para Desenvolvimento	38
4.9.6	Banco de Dados.....	39
4.9.7	Configuração de Redes para API e Integração.....	40
4.10	Protótipo Sistema Desktop.....	41

4.11 Protótipo Site.....	45
5. API	50
5.1 Visão Geral da API	50
5.2 Endpoints e Funcionalidades.....	51
5.2.1 NovaVenda	52
5.2.2 RealizarLogin	54
5.2.3 ArmazenarRespostas.....	55
5.2.4 Funções de GET	56
5.3 Testes.....	57
6. SISTEMA TOTEM	58
6.1 Manual de instalação	58
6.2 Introdução ao sistema.....	59
6.3 Mapa	60
6.4 Exposições	61
6.5 Questionario.....	62
6.6 Teclado	67
6.7 Imagens	67
6.8 Newtonsoft.Json.....	68
6.9 Tempo de Ausência	69
6.10 Manual de Desinstalação	69
7. SITE	70
7.1 Página Inicial.....	70
7.2 Página de Informações	71
7.3 Compra	71
7.4 Login	72
7.5 Exposições	73
7.6 Questionario.....	74
7.7 Responsividade.....	75
CONCLUSÃO	80
BIBLIOGRAFIA.....	82
APÊNDICE A - PERGUNTAS E RESPOSTAS.....	84
APÊNDICE B - EXPOSIÇÕES E DESCRIÇÕES	85
APÊNDICE C – DIAGRAMAS DE CLASSE	93
APENDICE D – DIAGRAMA DE SEQUÊNCIA.....	95
APÊNDICE E – CODIGO API – MASTERCONTROLLER – C#	96
APÊNDICE F – CODIGO API – INGRESSOCONTROLLER – C#.....	104

APÊNDICE G – CODIGO API – VENDACONTROLLER – C#.....	106
APÊNDICE H – CODIGO API – INGRESSOSVENDASCONTROLLER – C#.....	108
APÊNDICE I – CODIGO API – RELATORIOINGRESSOSCONTROLLER – C#	110
APÊNDICE J – CODIGO API – RELATORIOVENDASCONTROLLER – C#	114
APÊNDICE K – CODIGO API – QUESTIONARIORESPOSTASCONTROLLER – C#	118
APÊNDICE L – CODIGO API – AVALIACAORESPOSTASCONTROLLER – C#	122
APÊNDICE M – CODIGO API – AVALIACAOSUGESTAOCONTROLLER – C#.....	126
APÊNDICE N – CODIGO API – VALIDACAO – C#.....	127
APÊNDICE O – CODIGO API – KEYGENERATOR – C#.....	132
APÊNDICE P – CODIGO API – EXPLORARMARTEDBCONTEXT – C#.....	135
APÊNDICE Q – CODIGO API – INGRESSOMODEL – C#.....	136
APÊNDICE R – CODIGO API – VENDAMODEL – C#.....	137
APÊNDICE S – CODIGO API – INGRESSOVENDAMODEL – C#	138
APÊNDICE T – CODIGO API – QUESTIONARIORESPOSTASMODEL – C#	139
APÊNDICE U – CODIGO API – AVALIACAORESPOSTASMODEL – C#	140
APÊNDICE V – CODIGO API – AVALIACAOSUGESTAOMODEL – C#	141
APÊNDICE W – CODIGO API – RELATORIOINGRESSOSMODEL – C#	142
APÊNDICE X – CODIGO API – RELATORIOVENDASMODEL – C#.....	143
APÊNDICE Y – CODIGO API – AVALIACAORESPOSTASDTO – C#.....	144
APÊNDICE Z – CODIGO API – EMAILKEYDTO – C#	145
APÊNDICE AA – CODIGO API – INGRESSODTO – C#	146
APÊNDICE AB – CODIGO API – LOGINRESPOSTADTO – C#	147
APÊNDICE AC – CODIGO API – NOMEKEYDTO – C#.....	148
APÊNDICE AD – CODIGO API – NOVAVENDADTO – C#.....	149
APÊNDICE AF – CODIGO API – QUESTIONARIOAVALIACAORESPOSTASDTO – C#	150
APÊNDICE AG – CODIGO API – QUESTIONARIORESPOSTASDTO – C#	151
APÊNDICE AH – CODIGO API – RELATORIOINGRESSOSDTO – C#	152
APÊNDICE AI – CODIGO API – RELATORIOVENDASDTO – C#	153
APÊNDICE AJ – CODIGO TOTEM – MAINACTIVITY – XAML	154
APÊNDICE AK – CODIGO TOTEM – MAINACTIVITY – C#	156
APÊNDICE AL – CODIGO TOTEM – CONFIGURACOES – XAML.....	158
APÊNDICE AM – CODIGO TOTEM – CONFIGURACOES – C#	160
APÊNDICE AN – CODIGO TOTEM – EXPOSICOES – XAML.....	161
APÊNDICE AO – CODIGO TOTEM – EXPOSICOES – C#	163
APÊNDICE AP – CODIGO TOTEM – EXPOSICAO – XAML.....	165
APÊNDICE AQ – CODIGO TOTEM – EXPOSICAO – C#	166

APÊNDICE AR – CODIGO TOTEM – MAPA – XAML	170
APÊNDICE AS – CODIGO TOTEM – MAPA – C#	171
APÊNDICE AT – CODIGO TOTEM – QUESTIONARIO– XAML	173
APÊNDICE AU – CODIGO TOTEM – QUESTIONARIO– C#.....	175
APÊNDICE AV – CODIGO TOTEM – AVALIACAO – XAML.....	181
APÊNDICE AW – CODIGO TOTEM – AVALIACAO – C#.....	184
APÊNDICE AX – CODIGO TOTEM – RELATORIORESPOSTAS – XAML.....	203
APÊNDICE AY – CODIGO TOTEM – RELATORIORESPOSTAS – C#.....	206
APÊNDICE AZ – CODIGO TOTEM – CONTROLE – C#	213
APÊNDICE BA – CODIGO TOTEM – ESTATICO – C#.....	217
APÊNDICE BB – CODIGO TOTEM – VALIDACAO – C#.....	218
APÊNDICE BC – CODIGO TOTEM – CONTROLLSERVICE – C#	220
APÊNDICE BD – CODIGO TOTEM – QUESTIONARIORESPOSTASMODEL – C#.....	222
APÊNDICE BE – CODIGO TOTEM – AVALIACORESPOSTASDTO – C#	223
APÊNDICE BF – CODIGO TOTEM – AVALIACAOUSGETAODTO – C#.....	224
APÊNDICE BG – CODIGO TOTEM – QUESTIONARIOAVALIACAOQUESTDTO – C#	225
APÊNDICE BH – CODIGO TOTEM - QUESTIONARIOAVALIACAOSEMSUGESTAODTO – C#	226
APÊNDICE BI – CODIGO TOTEM – QUESTIONARIORESPOSTASDTO – C#.....	227
APÊNDICE BJ – CODIGO SITE – APICONFIG – JAVASCRIPT.....	228
APÊNDICE BK – CODIGO SITE – AUTH – JAVASCRIPT.....	229
APÊNDICE BL – CODIGO SITE – HOME – HTML	230
APÊNDICE BM – CODIGO SITE– HOME – CSS	232
APÊNDICE BN – CODIGO SITE – HOME – JAVASCRIPT	238
APÊNDICE BO – CODIGO SITE – INFORMACOES – HTML	239
APÊNDICE BP – CODIGO SITE – INFORMACOES – CSS.....	242
APÊNDICE BQ – CODIGO SITE – INFORMACOES – JAVASCRIPT	247
APÊNDICE BR – CODIGO SITE – COMPRAS – HTML.....	248
APÊNDICE BS – CODIGO SITE – COMPRAS – CSS.....	252
APÊNDICE BT – CODIGO SITE – COMPRAS – JAVASCRIPT.....	258
APÊNDICE BU – CODIGO SITE – LOGIN – HTML	262
APÊNDICE BV – CODIGO SITE – LOGIN – CSS.....	265
APÊNDICE BW – CODIGO SITE – LOGIN – JAVASCRIPT	270
APÊNDICE BX – CODIGO SITE – ADMIN_DASHBOARD – HTML.....	273
APÊNDICE BY – CODIGO SITE – ADMIN – CSS.....	274
APÊNDICE BZ – CODIGO SITE – RELATORIOQUESTIONARIOAVALIACAO – HTML.....	277
APÊNDICE CA – CODIGO SITE – RELATORIOQUESTIONARIOAVALIACAO – CSS	279

APÊNDICE CB – CODIGO SITE – RELATORIOQUESTIONARIOAVALIACAO – JAVASCRIPT	281
APÊNDICE CC – CODIGO SITE – RELATORIOS – HTML	285
APÊNDICE CD – CODIGO SITE – RELATORIOS – CSS.....	288
APÊNDICE CE – CODIGO SITE – RELATORIOS – JAVASCRIPT	291
APÊNDICE CF – CODIGO SITE – EXPOSICAO – HTML.....	301
APÊNDICE CG – CODIGO SITE – EXPOSICAO – CSS.....	309
APÊNDICE CH – CODIGO SITE – EXPOSICAO – JAVASCRIPT	316
APÊNDICE CI – CODIGO SITE – QUESTIONARIO – HTML	319
APÊNDICE CJ – CODIGO SITE – QUESTIONARIO – CSS.....	326
APÊNDICE CK – CODIGO SITE – QUESTIONARIO – JAVASCRIPT.....	333
APÊNDICE CL – EXPLICAÇÃO TAGS EM CSS.....	340

1. INTRODUÇÃO

Nos dias de hoje, a tecnologia está presente em diferentes áreas da sociedade, sendo utilizada de várias formas. Seu avanço contínuo tem proporcionado uma maior conectividade, transformando a maneira como as pessoas conduzem suas rotinas diárias, oferecendo mais praticidade e conveniência. Quando pensamos em museus, geralmente imaginamos exposições de arte, peças históricas e relíquias antigas. No entanto, existem vários tipos de museus, e um museu multitemático se destaca por reunir coleções e exposições de diferentes temas e épocas. Diferente dos museus especializados, que focam em áreas específicas como arte contemporânea ou história natural, os museus multitemáticos proporcionam aos visitantes uma experiência diversificada, abordando múltiplos assuntos em um único espaço.

A incorporação da tecnologia revolucionou a maneira de interagir com espaços culturais, como os museus. Com cada avanço tecnológico, surgem novas possibilidades de reinventar experiências, desde aplicativos que funcionam como guias até vivências interativas, como acampamentos em realidade aumentada. Além de tornar essas interações mais acessíveis, a tecnologia oferece novas formas de interpretar e comunicar o conteúdo, abrindo portas para perspectivas inovadoras.

O *Museu da Evolução e Conhecimento Humano (MECH)*, uma organização sem fins lucrativos dedicada a disseminar o conhecimento sobre a evolução cultural e científica, especialmente em comunidades carentes do estado de São Paulo, assumiu a iniciativa de criar um museu multitemático, que contará com uma exposição sobre a possível *primeira viagem a Marte*. A exposição explorará todos os aspectos do planeta vermelho, abrangendo desde suas características físicas, como a massa e composição, até os avanços tecnológicos que levaram à criação dos robôs que atualmente exploram sua superfície.

Com o intuito de oferecer uma experiência mais imersiva e acessível aos visitantes, o projeto incluirá a criação de um totem interativo e um site. Ambas as plataformas permitirão que os usuários explorem as exposições, desafiem seus conhecimentos sobre Marte e accessem um mapa interativo para guiar os visitantes durante a visita ao museu.

Este projeto tem como objetivo integrar a inovação tecnológica com a preservação do conhecimento histórico e cultural relacionado à exploração espacial a Marte. Ao combinar interatividade, informação e imersão, o totem interativo

proporcionará aos visitantes do museu uma experiência dinâmica e educativa, incentivando a aprendizagem de maneira envolvente.

O desenvolvimento e a implementação do software para o totêm interativo serão detalhados desde a definição dos objetivos e funcionalidades até os recursos tecnológicos que irão moldar a experiência do usuário. Esse processo incluirá a descrição das metas a serem alcançadas, os recursos necessários para sua execução e as características que garantirão uma interação eficaz e envolvente com os visitantes.

1.1 Objetivo Geral

Desenvolver um software para um totêm interativo e um site responsivo para computadores e dispositivos móveis, visando proporcionar aos visitantes do museu multitemático uma experiência de exploração e compreensão do planeta Marte.

1.2 Objetivos Específicos

Esse projeto tem como objetivos específicos:

- Planejar o sistema de totêm e site
- Criar protótipo e diagramas
- Planejar o banco de dados e infraestrutura de redes
- Desenvolver sistema totêm/site

2. PESQUISA BIBLIOGRÁFICA

Para o desenvolvimento do projeto, foi realizada uma pesquisa bibliográfica utilizando livros, artigos científicos e publicações acadêmicas. Essa abordagem permitiu uma melhor compreensão das práticas e teorias relacionadas ao desenvolvimento de software, sistemas interativos e tecnologias de interface usuário-máquina.

A pesquisa bibliográfica, conforme descrito por Gil (2002), baseia-se na análise de fontes secundárias e oferece uma base sólida de conhecimento teórico, fundamental para uma compreensão aprofundada dos conceitos necessários ao desenvolvimento do projeto.

2.1 Marte

O planeta vermelho, o quarto planeta a partir do sol, Marte, conhecido pelos Romanos como o deus da guerra devido a sua cor vermelha, desperta o interesse da humanidade devido a possibilidade de existir ou não, vida fora da terra. Há muitos anos as potências mundiais planejam viagens e expedições ao planeta a fim de novas descobertas científicas. No entanto, essa jornada apresenta desafios significativos que vão desde questões tecnológicas até problemas de saúde e logística (MARTINS, 2011).

Em 1965 aconteceu a primeira exploração a Marte, com a sonda Mariner 4 da NASA, onde ocorreu o primeiro pouso bem-sucedido com imagens da superfície do planeta, sendo um marco na exploração planetária. A viagem até Marte teve a duração de 8 meses, chegando no dia 14 de julho de 1965. Durante o voo conseguiu capturar 21 imagens da superfície do planeta, revelando um solo cheio de crateras, semelhante a Lua. A expedição estabeleceu um marco para futuras viagens ao planeta vermelho (NASA, 2002).

Atualmente as missões para marte tem como objetivo não apenas uma exploração mais detalhada, mas também analisar uma possível colonização futura, visto que em missões passadas foram observadas a probabilidade de existir vida microbiana passada. As missões Artemis e Mars Sample Return visam estabelecer uma presença sustentável e estudar com mais detalhes o solo do planeta (RAMIREZ, 2017).

Um dos principais desafios é o desenvolvimento de naves espaciais capazes de realizar uma viagem segura e eficiente até Marte. A NASA, por exemplo, está trabalhando em tecnologias avançada para reduzir o tempo de viagem e minimizar os riscos. Estimasse que uma missão tripulada a Marte possa levar entre seis e nove meses, dependendo da tecnologia de propulsão utilizada (NASA, 2021).

Apesar dos desafios, o potencial para descobertas científicas e a possibilidade de estabelecer uma presença humana em Marte são grandes motivadores para a exploração. A missão Mars 2020, que enviou o rover Perseverance ao planeta vermelho, ajuda a identificar locais promissores para a exploração futura e a buscar sinais de vida passada (NASA, 2023).

2.2 Modelos de Dados: Conceitual, Lógico e Físico

O trabalho com modelos de dados conceitual, lógico e físico é fundamental para o sucesso de qualquer projeto. Cada um desses modelos desempenha um papel específico e crucial no desenvolvimento de sistemas de informação. Neste texto, exploraremos cada um desses modelos, detalhando suas características, importâncias e a necessidade real de trabalhar com eles. (SORDI, 2020).

2.2.1 Modelo de Dados Conceitual

O **modelo de dados conceitual** é a representação de alto nível do banco de dados, focada em como os usuários finais enxergam os dados e suas relações. Este modelo é construído sem se preocupar com a implementação no nível de hardware ou software, proporcionando uma visão clara e comprehensível para todos os stakeholders, incluindo aqueles sem conhecimento técnico profundo. (SORDI, 2020)

Importância:

- **Comunicação:** Facilita a comunicação entre desenvolvedores, analistas e usuários finais, garantindo que todos entendam e concordem com a estrutura do banco de dados.
- **Documentação:** Serve como documentação do sistema, ajudando na manutenção e na escalabilidade futura.
- **Planejamento:** Auxilia no planejamento estratégico, identificando todas as entidades e suas relações antes de qualquer desenvolvimento físico.

Necessidade: Trabalhar com o modelo de dados conceitual é essencial para assegurar que as necessidades do negócio e dos usuários sejam claramente compreendidas e adequadamente representadas. Ele evita problemas futuros decorrentes de mal-entendidos ou falhas de comunicação.

2.2.2 Modelo de Dados Lógico

O **modelo de dados lógico** é uma representação mais detalhada e técnica do banco de dados, que define a estrutura dos dados em termos de tabelas, colunas, tipos de dados, relações e regras de integridade. Ele não se preocupa com detalhes físicos como armazenamento ou desempenho, mas se foca na implementação lógica dos dados. (SORDI, 2020)

Importância:

- **Precisão:** Detalha com precisão como os dados serão organizados e estruturados, o que é vital para a integridade e consistência do banco de dados.
- **Validação:** Permite validar a modelagem de dados contra as necessidades e regras do negócio, garantindo que todos os requisitos sejam atendidos.
- **Flexibilidade:** Fornece uma camada de abstração entre o modelo conceitual e a implementação física, permitindo ajustes e melhorias sem afetar a visão do usuário.

Necessidade: Trabalhar com o modelo de dados lógico é necessário para transformar a visão conceitual em uma estrutura técnica que pode ser implementada e mantida de maneira eficiente. Ele assegura que a estrutura do banco de dados seja sólida e que as regras do negócio sejam aplicadas corretamente.

2.2.3 Modelo de Dados Físico

O **modelo de dados físico** descreve a implementação real do banco de dados no sistema de gerenciamento de banco de dados (SGBD). Ele inclui detalhes como a definição de tabelas, índices, partições, clusters, e métodos de acesso aos dados, além de preocupações com desempenho, espaço em disco e backups. (SORDI, 2020)

Importância:

- **Desempenho:** Otimiza a performance do banco de dados, garantindo acesso rápido e eficiente aos dados.

- **Armazenamento:** Gerencia o armazenamento físico dos dados, assegura que o espaço em disco seja usado de forma eficiente.
- **Segurança:** Implementa controles de segurança e backups, garante a proteção e recuperação dos dados.

Necessidade: Trabalhar com o modelo de dados físico é essencial para garantir que o banco de dados funcione de forma eficiente no ambiente de produção. Ele transforma a teoria do modelo lógico em uma realidade prática, lidando com desafios de desempenho, armazenamento e segurança.

Os modelos de dados conceitual, lógico e físico são interdependentes e cada um desempenha um papel fundamental no ciclo de desenvolvimento de um banco de dados. O modelo conceitual garante que todas as necessidades do negócio e do usuário sejam compreendidas e documentadas. O modelo lógico transforma essas necessidades em uma estrutura técnica detalhada, enquanto o modelo físico implementa essa estrutura de maneira eficiente e segura. Trabalhar com esses modelos não é apenas uma prática recomendada, mas uma necessidade real para assegurar o sucesso e a longevidade dos sistemas de informação. (SORDI, 2020)

2.3 Diagrama de Pacotes

Os diagramas de pacotes são usados na UML (Unified Modeling Language) para organizar e mostrar a disposição de vários elementos de um sistema em forma de pacotes. Um pacote é um agrupamento de elementos relacionados, como classes, documentos, diagramas ou até mesmo outros pacotes. (BOOCH, RUMBAUGH, JACOBSON, 2005)

Esses diagramas são úteis para simplificar diagramas de classes complexos, oferecendo uma visão clara da estrutura hierárquica e das dependências entre os elementos. Eles são frequentemente usados para ilustrar a arquitetura de um sistema, mostrando como as diferentes partes do sistema estão agrupadas e interconectadas. Criar um diagrama de pacotes pode ser uma tarefa bastante útil para organizar e visualizar a estrutura de um sistema. (BOOCH, RUMBAUGH, JACOBSON, 2005)

2.4 Diagrama de Classes

Um diagrama de classes é usado na modelagem de sistemas orientados a objetos, geralmente no contexto da UML (Unified Modeling Language), que descreve a estrutura de um sistema, onde mostra suas classes, atributos, métodos e os relacionamentos entre elas. (FURLAN, 1998)

Os elementos principais de um diagrama de classes:

Classes: Representam entidades ou objetos no sistema. São representadas no topo como retângulos com três seções:

- Nome da classe.
- Atributos (propriedades ou características da classe).
- Métodos (operações ou comportamentos da classe).

Atributos: Definem as propriedades de uma classe, como idCliente, nome, idade, telefone, e-mail, etc.

Métodos: São as funções ou operações que uma classe pode executar, como realizarLogin, realizarCadastro, atualizarPerfil, etc

Relacionamentos:

- **Associação:** Representa uma ligação entre duas classes.
- **Herança (ou Generalização):** Indica que uma classe herda atributos e métodos de outra classe.
- **Agregação:** Um relacionamento de "parte-todo", onde a parte pode existir independentemente do todo.
- **Composição:** Um tipo mais forte de agregação, onde a parte só existe enquanto o todo existir.
- **Dependência:** Uma classe depende de outra para funcionar, mas não mantém uma relação duradoura com ela.

Exemplos de Aplicação:

Um diagrama de classes é frequentemente utilizado em projetos de software para:

1. Planejar a estrutura do código.
2. Visualizar as interações entre diferentes partes do sistema.
3. Ajudar na comunicação entre desenvolvedores e stakeholders.

É especialmente útil na fase de design e arquitetura de sistemas complexos, facilitando a organização e a clareza da lógica do sistema. Um diagrama de classes para uma tela de totêmico de autoatendimento como o do museu poderia modelar a interação do usuário com a interface e o sistema de backend. O foco seria nas classes que representam a interface, o pedido, o pagamento, e os dados dos produtos. (BOOCH, RUMBAUGH, JACOBSON, 2005)

2.5 Diagrama de Sequência

O diagrama de sequência mostra a interação entre diferentes componentes de um sistema ao longo do tempo, focando na ordem das mensagens trocadas entre os objetos ou classes. No caso do totêmico de autoatendimento, o diagrama de sequência pode descrever o fluxo de um cliente que interage durante todo o processo de compra, verifica as atrações, escolhe, reserva, realiza o pedido do ingresso, processando o pagamento e recebendo o ingresso. (JACOBSON, BOOCH, RUMBAUGH, 2005)

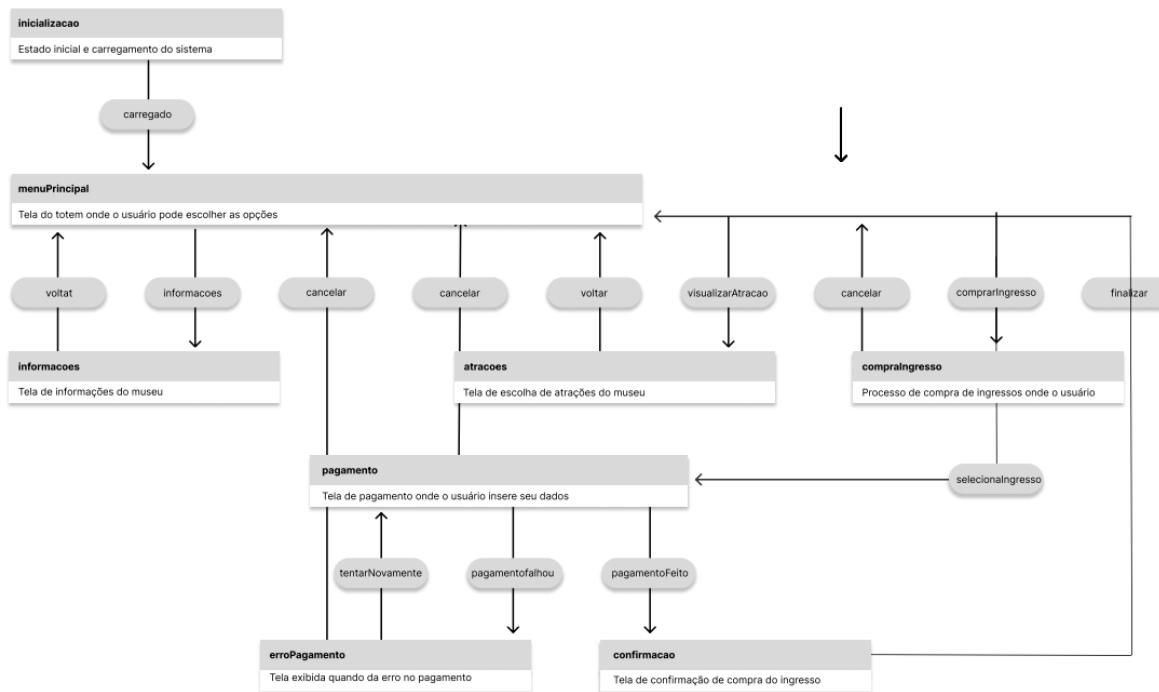
2.6 Diagrama de Máquina de Estados

O diagrama de máquina de estados descreve os estados pelos quais um objeto pode passar durante seu ciclo de vida, assim como as transições entre esses estados baseadas em eventos. Para o caso de um totêmico de autoatendimento, o foco estaria nas diferentes etapas do processo de um pedido, desde o início até a finalização, considerando o comportamento da interface do usuário e do sistema. (JACOBSON, BOOCH, RUMBAUGH, 2005)

Este tipo de diagrama detalha com precisão todos os caminhos possíveis que o usuário pode percorrer, indo e voltando de cada escolha dentro das funções implantadas no totêmico. O usuário consegue percorrer desde a realização completa da compra sem que tenha qualquer erro, falha de pagamento e até cenários como falha de pagamento, onde o diagrama mostra o caminho para retornar e tentar novamente, desta forma pode optar por outra forma de pagamento ou sair do sistema. Em uma

única tela conseguimos apresentar abaixo o processo da tela do totêm que o usuário pode percorrer. Segue a Figura seguinte, onde representa o diagrama máquina de estados do projeto:

Figura 1: Diagrama - Máquina de Estados



Fonte: Autoria Própria

2.7 Diagrama de Comunicação

Esse diagrama é uma alternativa para o diagrama de sequência. A principal função de um diagrama de comunicação, é identificar vínculos, que são ligações existentes entre os objetos, ou seja, obter mensagens enviadas. Vínculos são representados por linhas contínuas unindo dois objetos, que devem ter um relacionamento equivalente ao diagrama de classe. A ordem das mensagens enviadas deve ser equivalente ao diagrama de sequência, porém sem o tempo de atividade. No geral representam a chamada de um método, as mensagens não devem possuir um retorno. Assim como no diagrama de sequência, os atores possuem vínculos com outros objetos e com outros atores. (FURLAN, 1998)

A Figura a seguir demonstra o diagrama de comunicação do projeto:

Figura 2: Diagrama - Diagrama de Comunicação



Fonte: Autoria Própria

3. CONTEXTUALIZAÇÃO DO PROJETO

O Museu da Evolução e Conhecimento Humano (MECH), uma organização sem fins lucrativos, pretende criar um museu multitemático com uma exposição sobre a possível primeira viagem a Marte.

O Museu (MECH) pretende proporcionar experiências educativas ao público, estimulando conhecimento, curiosidade e imaginação. Para tornar a visita mais envolvente, o (MECH) estabeleceu uma parceria com a equipe LIT - Tecnologia, reconhecida por seu compromisso com inovação tecnológica.

A equipe LIT - Tecnologia se comprometeu a desenvolver um software personalizado que será utilizado tanto em um totêm interativo quanto em um site responsivo. O software permitirá que os visitantes explorem informações sobre Marte e participem de desafios de conhecimento interativo. Além disso, tanto o totêm quanto o site oferecerão um mapa interativo para auxiliar os visitantes durante o passeio no museu. Os dados coletados e as interações serão analisados para fornecer relatórios que ajudarão a instituição a decidir sobre a duração da exposição, possíveis melhorias e ajustes necessários.

O (MECH) conta com o apoio de stakeholders para a realização da exposição. Este apoio inclui empresas locais, instituições educacionais, autoridades governamentais e organizações culturais. O envolvimento desses stakeholders é fundamental para garantir o sucesso da exposição, fornecendo recursos essenciais para a realização do evento no museu, bem como para o desenvolvimento do software para o totêm interativo e o site responsivo.

De acordo com Pereira (2021), um museu é um local que preserva objetos históricos em exposição para manter a cultura, história de um local, da região ou do mundo oferecendo conhecimento, aprendizado e reflexão, através do decorrer do tempo, existem museus específicos e museus multitemáticos, que apresenta vários temas, coleções e exposições no mesmo local.

O Museu da Evolução e Conhecimento Humano (MECH) pretende, nesta exposição, proporcionar uma visão abrangente sobre Marte, destacando suas características físicas, origem e relevância científica. Através de recursos tecnológicos, os visitantes poderão explorar informações como a massa, composição, atmosfera e histórico geológico do planeta, além de teorias sobre sua formação e a possibilidade de futuras missões tripuladas. A exposição busca despertar o interesse

pelo estudo do planeta vermelho, que ofereça uma experiência imersiva e educativa que une ciência, tecnologia e curiosidade sobre o universo.

A ideia para o desenvolvimento de um software para o totêm interativo e um site, é proporcionar aos visitantes uma experiência de interação e aprendizado. O software permite que os usuários explorem informações detalhadas sobre o planeta Marte e participem de desafios interativos, enriquecendo a visita e incentivando uma compreensão mais profunda do tema

Com uma interface intuitiva, o totêm interativo e o site responsivo oferecem informações detalhadas sobre as exposições, questionários, conteúdo multimídia e um mapa do museu. O objetivo é integrar a tecnologia ao ambiente de forma interativa e educativa, proporcionando fácil acesso aos visitantes e enriquecendo sua experiência ao explorar o tema da exposição.

A implantação dos dois produtos oferece uma interação aprimorada através da tecnologia, tornando a experiência mais dinâmica e imersiva. Além de proporcionar um envolvimento mais profundo com a exposição, esses sistemas coletam informações valiosas para avaliar a aceitação do tema e identificar áreas que precisam de melhorias, ajudando a otimizar a experiência dos visitantes.

4. PLANEJAMENTO

De acordo com Sommerville (2011), para garantir a execução correta de um projeto ou ação, é essencial realizar um planejamento prévio. Esse planejamento é crucial para o sucesso do projeto, pois torna o processo mais eficiente e ágil, permitindo que o desenvolvimento e a entrega ao cliente ocorram de maneira eficaz. Além disso, um bom planejamento assegura a correção e a otimização dos recursos, resultando em um produto de alta qualidade.

4.1 Ciclo de Vida

De acordo com Kochanski (2013), o ciclo de vida é uma etapa importante de engenharia de software, essa é a etapa onde são decididos os passos de um sistema desde a sua criação até a finalização, ele servirá como guia para planejamento, desenvolvimento e manutenção do sistema. A qualidade do software durante o ciclo de vida é importante para garantir que o produto atenda às necessidades e expectativas dos usuários, desde a concepção, manutenção até a evolução.

Conforme explica o autor citado acima, é preciso garantir a qualidade em cada fase do ciclo de vida do software desde a concepção com definições claras dos requisitos do software, acompanhando o projeto e o desenvolvimento, realizando testes para identificar e corrigir defeitos precocemente, garantindo a implantação adequada do software e realizando manutenção para atender as necessidades dos usuários.

O ciclo de vida de um software inicia no momento em que são registradas as primeiras especificações de funcionamento, desenvolvimento, implementação, teste e lançamento, até o momento em que o software deixa de ser utilizado pelos seus usuários. (KOCHANSKI, p. 52 2013)

4.2 Termo de Abertura do Projeto (TAP)

De acordo com o PMI (2017), o Termo de Abertura do Projeto (TAP) é um documento formal que marca o início do projeto e é elaborado pela equipe responsável em conjunto com a parte interessada, ou seja, o solicitante do projeto. O TAP descreve os recursos, os objetivos, os requisitos iniciais, o cronograma preliminar, os riscos

identificados e outras informações essenciais. Funciona como um contrato interno que estabelece as bases e as expectativas das partes envolvidas no desenvolvimento do projeto, sendo crucial para manter todas as partes alinhadas e garantir a compreensão clara dos objetivos. O documento TAP pode ser encontrado na Figura 3, Figura 4, Figura 5 e Figura 6.

Figura 3: TAP - Parte 1

TERMO DE ABERTURA DO PROJETO

1 – Nome do Projeto	2 – Código
Descobrindo Marte com Tecnologia	2024_04_PIM_v01
3 – Líder do Projeto	3.1 - Área de Iotação
Leandro Moraes de Souza	Desenvolvedor Back-End
3.2 – E-mail	3.3 – Telefone
leandroms.sr1708@gmail.com	(11) 91108-1315
4 – Gestores do Projeto	4.1 – Área de Iotação
Isac Ferreira de Souza	Design Gráfico
4.2 – E-mail	4.3 – Telefone
isacfouza14@gmail.com	11 977438506
5. Objetivo do Documento	
Este documento tem como objetivo autorizar formalmente o início de um projeto e contém informações necessárias para o entendimento do projeto, fornecendo uma visão macro do serviço a ser desenvolvido.	

6 – Histórico de Mudança			
Versão	Data	Descrição	Autor
V01	08/08/2024	Criação de um projeto que auxilia visitantes de um museu, nas consultas às atrações com interação com questionários.	Leandro M. Souza

Fonte: Autoria Própria

Figura 4: TAP - Parte 2

7 – Objetivo do Projeto		
Criar um sistema nas plataformas desktop para uso em totem e web para acesso via smartphone, para os visitantes consultarem as atrações e realizar questionários de pesquisa, as respostas servem para criar um relatório para futuras tomadas de decisões, permitindo uma análise mais precisa das percepções e interesses do público.		
8 – Justificativa		
A criação deste projeto visa proporcionar uma experiência única a pessoas que possuem interesse em exploração espacial, em especial o planeta vermelho, permitindo que mergulhem de forma interativa e informativa nas descobertas e avanços.		
9 – Escopo		
<p>1- Pesquisar: Marte, exploração espacial e implementação de banco de dados.</p> <p>2- Desenvolver: Sistema desktop, sistema Web, layout agradável e banco de dados.</p> <p>3- Escolha de Ferramentas de Prototipagem: Utilizaremos o Figma.</p> <p>4- Adicionar interatividade: Serão incluídos interatividade através de fluxos de trabalho conforme os usuários selecionam as opções na tela.</p> <p>5- Testes: Após a conclusão de toda as telas e inclusão da interatividade, serão feitos testes para verificar se a funcionalidade atende os requisitos.</p> <p>6- Implementar: Após o sistema pronto, será implementado junto ao cliente e pessoas envolvidas com o projeto para uma avaliação para verificar se atende as necessidades da empresa.</p>		
10 – Não-Escopo		
Acessibilidade para PNE		
11 – Parte Interessada	Representante	Relacionamento com o projeto
Me. Richardson Kennedy Luz Me. Waldir Antônio da Silva Dr. Alex Sampaio Esp. Reverdan Spanger Equipe item 12	Me. Waldir Antônio da Silva	

Fonte: Autoria Própria

Figura 5: TAP - Parte 3

12 – Equipe Básica	Papel desempenhado
Leandro Moraes de Souza	Desenvolvedor Back- End
Isac Ferreira de Souza	Designer
Thais Reis da Silva	Documentação
Marcos Eduardo Fernandes	Arquiteto de Software
Henrique Carvalho	Desenvolvedor Front- End
Emily Camila de Sousa Ferreira	Engenheira de Software
13 – Orçamento Previsto	14 – Prazo Previsto
R\$181.740,00	• 14/11/2024

14 – Premissas (Suposições dadas como certas para o projeto)	
1.	Análise de Sistemas
2.	Criar um sistema totem, desenvolvido em plataforma .NET com C# e parte visual com WPF. O sistema WEB será desenvolvido em HTML, CSS e back-end C# .NET.
3.	Apresentar os diagramas de Classes, Sequência, Máquina de Estados, Comunicação e Pacotes.
4.	Apresentar Modelo de Dados normalizado (Mapeamento Objeto-Relacional)
5.	Projetar a Arquitetura do Sistema
6.	Implementar o projeto desenvolvido em C# - .NET
7.	Apresentar um módulo Windows e um módulo WEB responsivo. O banco de dados deve ser único para os módulos. Deverão conter as quatro operações básicas – CRUD.
8.	Demonstrar na fase de codificação as boas práticas de desenvolvimento.
9.	Apresentar o cronograma do projeto.
10.	Apresentar um estudo para iniciar um empreendimento na área de desenvolvimento de sistemas.
11.	Apresentar informações sobre o controle de qualidade utilizado na produção do sistema. Testes automatizados, CI/CD. Este item somente teórico.

Fonte: Autoria Própria

Figura 6: TAP - Parte 4

Aprovação		
Responsável	Data	Assinatura
Me. Richardson Kennedy Luz	08/08/2024	

Fica acordado o Projeto de Serviço, favor imprimir 2 vias da Abertura do Projeto e assinar ambas, para cada Responsável possuir uma (01) via.

15 – Cronograma
2024_04_PIM_v01_Cronograma.xlsx

16 – Atividades Desenvolvidas
Escopo: 2024_04_PIM_v01_Descobrindo_Marte.doc
TAP: 2024_04_PIM_v01_TAP.doc
Telas Figma: 2024_04_PIM_v01_TELAS_FIGMA.pdf

Fonte: Autoria Própria

4.3 Orçamento do Projeto

Para iniciar o projeto, foram conduzidas pesquisas de mercado e estudos de viabilidade para compreender as necessidades do museu e as expectativas do público visitante. Com base nesses dados, foi definido um orçamento apropriado para o desenvolvimento e implantação do totêm interativo. O orçamento, conforme detalhado na tabela a seguir, abrange os custos relacionados ao desenvolvimento do software, aquisição dos dispositivos para o museu, mão de obra, além de despesas com rede e manutenção. A seguir a Tabela de orçamento do projeto:

Tabela 1:Orçamento

ORÇAMENTO		
Desenvolvimento de Software	R\$	135.000,00
Dispositivos	R\$	21.740,00
Mão de Obra	R\$	15.000,00
Outros Custos	R\$	10.000,00
TOTAL	R\$	181.740,00

Fonte: Autoria Própria

Desenvolvimento de Software: Neste projeto, foi colocado 6 membros da Equipe LIT para fazer o planejamento do software, toda documentação e o software para o totem. Durante esses três meses, cada membro terá seu pagamento de R\$7.500 para desenvolver e documentar.

Dispositivos: Na procura de um totem, foi escolhido a Mesa Digital Touch Screen 32 Polegadas da *Index Soluções* para ser instalado no museu. Será instalado dois totens desse modelo, que custarão por volta de R\$21.740.

Mão de Obra: Para a instalação dos dois totens, foi contratado três membros da equipe técnica, o preço total para a equipe será de R\$15.000

Outros Custos: Foi colocado R\$10.000 para questões de cabos de redes, modem, switch ou até mesmo alguma manutenção do dispositivo.

4.4 Responsabilidade dos membros da equipe.

Para organizar o projeto, definindo a função de cada membro da equipe foi utilizada a matriz RACI para distribuir as responsabilidades da equipe, conforme apresentado na Tabela 2.

A matriz RACI é uma ferramenta que auxilia na gestão de projetos ajudando a definir e comunicar os papéis e responsabilidades das atividades para cada membro da equipe evitando conflitos, ela apresenta quatro tipos de responsabilidades que podem ser atribuídas para cada tarefa, atividade ou decisão:

Responsável é aquele que tem a responsabilidade pela execução das atividades;

Aprovador tem a autoridade para tomar decisões e garantir que o trabalho seja feito corretamente.

Consultado é aquele que os membros recorrem para durante o processo, mas não tem autoridade para aprovar ou executar a atividade.

Informado é aquele membro que precisa receber as informações sobre o projeto para se manter atualizado, mas não está diretamente envolvido na execução.

A matriz RACI organiza através de uma tabela onde as atividades são listadas em linhas e os papéis em colunas, sendo preenchidas com as iniciais correspondentes de cada responsabilidade.

Tabela 2: RACI

ATIVIDADES	LEANDRO MORAES	ISAC FERREIRA	THAIS REIS	HENRIQUE CARVALHO	MARCOS EDUARDO	EMILY CAMILA
Documentação	R	I	A	I	I	I
Requisitos Funcionais	R	I	I	I	I	C
Requisitos não Funcionais	R	I	I	I	I	C
Protótipo	C	R	I	C	I	I
Banco de Dados	R	I	I	I	C	I
API	R	I	I	C	C	I
Frontend - Site	A	I	R	A	I	I
Frontend - Totem	A	R	I	I	I	I
Backend - Site	A	I	I	R	C	I
Backend - Totem	R	I	I	A	C	I
Diagramas	I	I	I	I	A	R
Testes	R	I	I	R	I	I

Fonte: Autoridade Própria

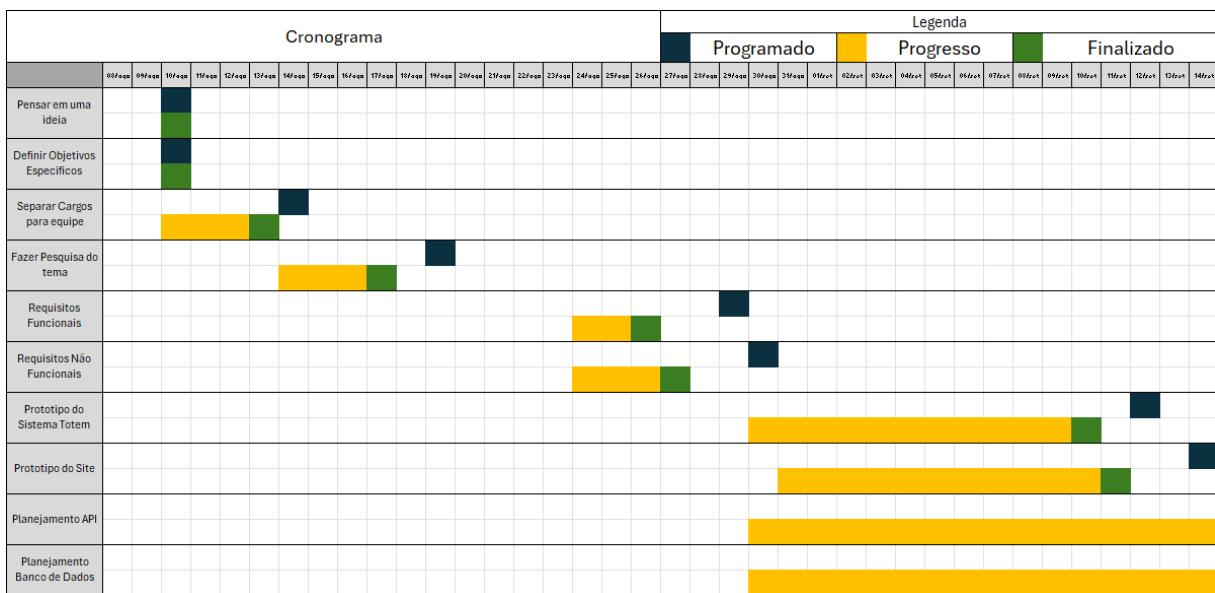
4.5 Cronograma

Um cronograma é uma representação visual, que apresenta as tarefas que foram definidas, e os prazos estabelecidos para entrega, fornecendo uma estrutura organizada para o planejamento e execução do projeto.

Após definir a função de cada membro na equipe se faz necessário montar um cronograma para alcançar as metas estabelecidas no projeto, garantindo seu desenvolvimento nos prazos estabelecidos, possibilita manter a organização e facilita a comunicação da equipe.

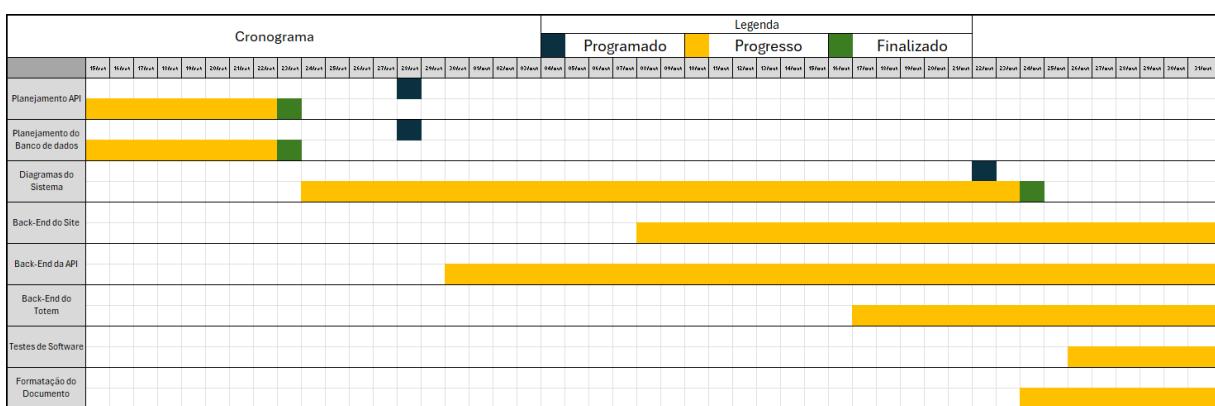
O cronograma da equipe se encontra na Tabela 3, Tabela 4 e Tabela 5

Tabela 3: Cronograma - Parte 1



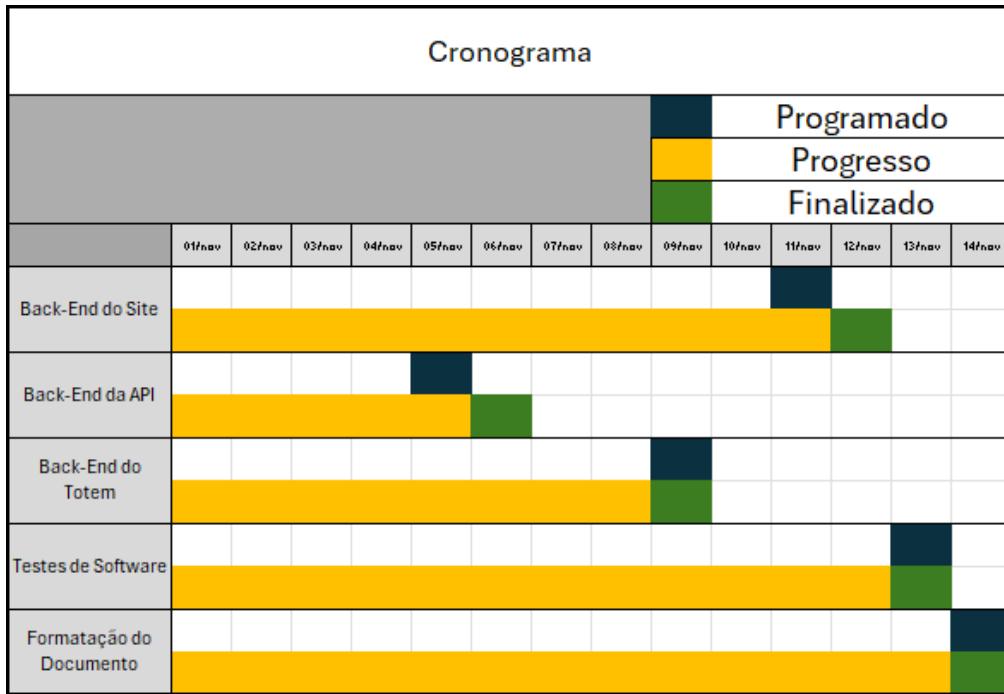
Fonte: Autoridade Própria

Tabela 4: Cronograma - Parte 2



Fonte: Autoridade Própria

Tabela 5: Cronograma - Parte 3



Fonte: Autoridade Própria

4.6 Desenvolvimento

De acordo com Pressman (2020), antes de iniciar o desenvolvimento de um software, é essencial analisar os requisitos do sistema, identificando as necessidades do cliente, suas expectativas e as limitações técnicas do ambiente. Esse processo permite selecionar as tecnologias apropriadas para o desenvolvimento. Após estabelecer a base de hardware e o design de UI/UX, o desenvolvimento do software pode começar a incluir a criação de aplicativos, a integração de sistemas de gerenciamento de conteúdo e o desenvolvimento de funcionalidades específicas. Após o desenvolvimento inicial, o sistema passa por testes de qualidade para garantir que todas as funcionalidades operem conforme o esperado. Ajustes e melhorias são realizados para otimizar a experiência do usuário, e a manutenção contínua assegura que o sistema permaneça operacional durante o tempo necessário.

4.7 Qualidade de Software

De acordo com Pressman (2020), a norma ISO 9126 é uma diretriz internacional para a avaliação da qualidade do software. Composta por duas partes,

a norma define um modelo de qualidade na primeira parte, especificando características e atributos associados que são essenciais para a avaliação do software. Esses atributos incluem funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade, e são fundamentais para garantir que o software atenda aos padrões de qualidade estabelecidos.

Ainda de acordo com Pressman (2020), a segunda parte da ISO 9126 fornece diretrizes para a avaliação do modelo de qualidade do software e define métricas específicas para medir cada uma das características mencionadas na primeira parte. Essas métricas ajudam a quantificar o grau de qualidade do software, oferecendo orientações práticas para a aplicação e avaliação eficaz do produto.

De acordo com a norma ISO 9126, as características estão descritas abaixo e na Figura a seguir:

Funcionalidade: corresponde as capacidades fornecidas pelo software sobre os requisitos especificados, incluindo sua adequação funcional, precisão, e conformidade com os padrões.

Confiabilidade: é a característica que diz respeito à capacidade do software de manter seu nível de desempenho quando usado em condições estabelecidas durante um período determinado, incluindo capacidade de recuperação e estabilidade.

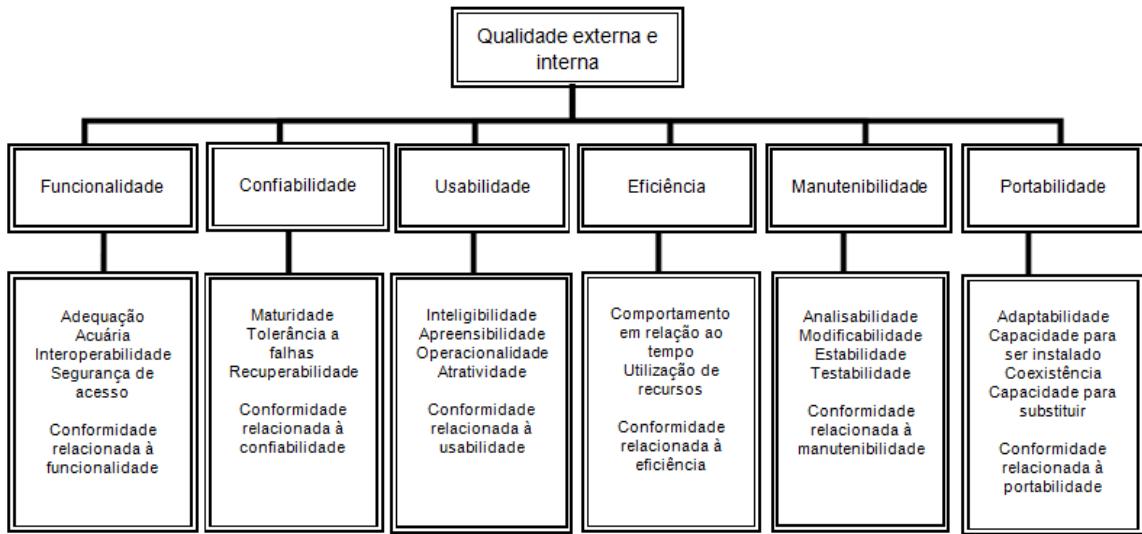
Usabilidade: refere-se à facilidade de uso do software pelo usuário, podendo dizer se considera fácil de interagir, compreender e operar.

Eficiência: avalia o desempenho do software com relação aos recursos utilizados como o consumo de memória, velocidade de processamento, e os recursos usados em condições estabelecidas.

Manutenibilidade: diz respeito a facilidade com que o software pode ser modificado, feito diagnóstico do que precisa ser corrigido e adaptado conforme os requisitos necessários.

Portabilidade: avalia a capacidade do software de poder ser transferido de um ambiente para outro e de se adaptar a diferentes plataformas e sistemas operacionais.

Figura 7: Princípios da qualidade de software da norma ISO 9126



Fonte:<https://www.researchgate.net/publication/332822299/figure/fig5/AS:754285969162240@1556847197082/Figura-5-Modelo-de-qualidade-externa-e-interna-ISO-IEC-9126.ppm>. Acessado em: 30 de abril de 2024

A ISO 9126 se destaca como uma ferramenta essencial para garantir a qualidade do software em diversos contextos. Com seu modelo estruturado e métricas específicas, a norma facilita a comunicação entre stakeholders, promove o desenvolvimento de software de alta qualidade e contribui significativamente para o sucesso de projetos de software.

4.8 Requisitos Funcionais

Requisitos funcionais. São declarações de serviços que o sistema deve fornecer, de como o sistema deve reagir a entradas específicas e de como o sistema deve se comportar em determinadas situações. Em alguns casos, os requisitos funcionais também podem explicitar o que o sistema não deve fazer. (SOMMERVILLE, 2011).

As funções planejadas para o totem e o site são:

- Visualizar mapa do museu
- Selecionar exposição
- Fazer um questionário

- Avaliar o museu

Mas o site tem um pouco mais de função, como:

- Venda de ingressos
- Acesso de conta

O sistema do site oferece funcionalidades adicionais em relação ao totém, incluindo a venda de ingressos e o acesso a contas. Para adquirir ingressos, os usuários podem acessar o site, selecionar a quantidade desejada e realizar o pagamento de forma rápida e segura. Após a compra, o usuário recebe um código por e-mail, que serve como chave de acesso à sua conta. Com esse código, o usuário pode acessar o site para explorar as descrições das exposições e participar do questionário interativo. Essa abordagem proporciona uma experiência personalizada e mais envolvente para os visitantes.

O totém e o site apresentam um mapa que exibe as localizações de cada obra, ao mesmo tempo em que mostram a posição do usuário, facilitando a busca pela obra desejada. Quando o usuário demonstra interesse em alguma das exposições do museu, ele a seleciona, e em seguida, a descrição é exibida na tela, permitindo que o usuário saiba mais sobre a obra apresentada.

Para os usuários que desejam colocar seu conhecimento em prática, o totém disponibiliza um questionário com cinco perguntas sobre o evento. Depois das perguntas sobre as exposições, surgem as perguntas de avaliação, permitindo que os usuários compartilhem suas impressões sobre o museu e comentem um pouco sobre a exposição. Essas respostas são salvas no banco de dados para a elaboração de um relatório de pesquisa.

Após completar todo o questionário, o sistema solicita o email do usuário, e em seguida, é apresentado um relatório com as questões que o usuário acertou e errou, além de uma estatística global que exibe a média de acertos e erros dos demais usuários.

4.9 Requisitos Não Funcionais

Requisitos não funcionais. São restrições aos serviços ou funções oferecidos pelo sistema. Incluem restrições de timing, restrições no processo de desenvolvimento e restrições impostas pelas normas. Ao contrário das características individuais ou serviços do sistema, os requisitos não funcionais, muitas vezes, aplicam-se ao sistema como um todo. (SOMMERVILLE, 2011).

4.9.1 Método Ágil

[...]Métodos ágeis, universalmente, baseiam-se em uma abordagem incremental para a especificação, o desenvolvimento e a entrega do software. Eles são mais adequados ao desenvolvimento de aplicativos nos quais os requisitos de sistema mudam rapidamente durante o processo de desenvolvimento. Destinam-se a entregar o software rapidamente aos clientes, em funcionamento, e estes podem, em seguida, propor alterações e novos requisitos a serem incluídos nas iterações posteriores do sistema. Têm como objetivo reduzir a burocracia do processo, evitando qualquer trabalho de valor duvidoso de longo prazo e qualquer documentação que provavelmente nunca será usada. (SOMMERVILLE, 2011)

No início do projeto, foi decidido utilizar uma abordagem ágil para o desenvolvimento de software. Essa escolha visou garantir a entrega de um produto de qualidade dentro do prazo estipulado. A metodologia adotada permitiu a divisão das atividades em pequenas etapas, chamadas "Sprints". Essa entrega contínua possibilitou avaliações frequentes pelo cliente, proporcionando feedbacks e ajustes rápidos conforme necessário.

4.9.2 User Interface e User Experience

De acordo com Fabricio Teixeira (2014), a experiência do usuário (UX/ User Experience) tem como foco principal assegurar que os produtos sejam de fácil utilização, intuitivos e proporcionem uma experiência positiva para os usuários. Isso inclui compreender as necessidades e os comportamentos dos usuários, desenvolver interfaces que satisfaçam essas necessidades de maneira eficiente e garantir que a interação com o produto seja fluida e agradável.

Ainda segundo Fabricio Teixeira (2014), a interface do usuário (UI/ User Interface) tem como foco o processo de projetar a aparência e a interação de um produto digital, como um site ou aplicativo. O UI Design concentra-se na interface visual e interativa que o usuário utiliza, incluindo o design de elementos como botões, menus, ícones, tipografia, cores e layouts. O objetivo é criar uma interface que seja ao mesmo tempo atraente, intuitiva e eficaz, facilitando a interação do usuário com o produto. Em síntese, o UI Design busca proporcionar uma experiência visual agradável e funcional.

4.9.3 Implementação do Dispositivo

O acesso ao site pode ser realizado a partir de qualquer dispositivo, como computadores, tablets e celulares. Não é necessário um dispositivo específico para acessar o site, já que, com um navegador instalado, é possível acessá-lo de qualquer lugar.

Para o acesso ao sistema de totem, é necessário um dispositivo específico para sua execução, o dispositivo precisa atender a determinadas especificações técnicas, pois o sistema não é compatível com todos os aparelhos. Fatores como velocidade de resposta, qualidade da tela e outras características influenciam diretamente no desempenho, além disso, o dispositivo deve ser de fácil instalação e manutenção, garante que a equipe responsável possa operá-lo e realizar eventuais ajustes com facilidade

O produto escolhido foi o Mesa Digital Touch Screen 32 Polegadas código SA3REMN da marca *Index Soluções*, o totem possui:

- Tela de 32 (16:9) polegadas touchscreen
- Processador Intel core i3.
- Memória Ram de 4GB
- SSD 120GB

O totem Mesa Digital Touch Screen 32 Polegadas, modelo SA3REMN da marca *Index Soluções*, foi selecionado por suas características que aliam eficiência e interatividade. Com uma tela de 32 polegadas e formato 16:9, o dispositivo oferece uma interface touchscreen que facilita a interação direta com o conteúdo exibido. Seu

processador Intel Core i3 garante um bom desempenho para aplicações dinâmicas, enquanto a memória RAM de 4GB e o armazenamento SSD de 120GB proporcionam agilidade e espaço suficiente para o armazenamento de dados. Esse conjunto de especificações torna o totêm uma excelente escolha para ambientes que demandam acessibilidade digital e experiência interativa.

Após a escolha do dispositivo, é necessário planejar as etapas para a instalação do totêm nos museus, iniciando pela logística de transporte até a montagem física no local designado. Durante esse processo, são consideradas as necessidades específicas de cada museu, como o posicionamento ideal e a integração com a infraestrutura existente. A instalação física de dois totens, por exemplo, leva aproximadamente 1 hora para cada dispositivo, dependendo da complexidade do ambiente e dos requisitos envolvidos.

Após a conclusão da instalação física, a equipe técnica concentrará seus esforços na instalação e configuração do software no dispositivo. O processo envolve a transferência dos arquivos de software, seguido pela configuração do sistema para uso. Em seguida, serão realizados testes para verificar se todas as funcionalidades estão operando corretamente. O tempo estimado para a instalação e configuração completa, incluindo os testes e verificações, é de aproximadamente 30 minutos.

Seguindo essas etapas de forma cuidadosa, estaremos preparados para implementar os totens no museu, criando uma experiência envolvente para os visitantes e contribuindo para o fortalecimento da cultura e da educação.

4.9.4 Ferramentas para equipe

Para a comunicação e reuniões da equipe, foram utilizados o WhatsApp e o Discord. Esses aplicativos proporcionaram uma maneira rápida e eficiente de trocar informações entre os membros, garantindo a sincronia e a integração das atividades da equipe.

Para organizar a equipe, foi utilizado o sistema Trello, que possibilitou uma estruturação clara dos objetivos. Usando cartões para dividir as atividades e checklists para detalhar as tarefas, o Trello facilitou o acompanhamento do progresso e a gestão das responsabilidades.

4.9.5 Ferramentas para Desenvolvimento

O Figma foi escolhido como a ferramenta para o desenvolvimento das telas do site e do totém do museu, devido à sua facilidade de uso e ampla variedade de funcionalidades. A plataforma permite a colaboração simultânea de várias pessoas no projeto, o que facilita o trabalho em equipe. Além disso, o Figma oferece uma interface intuitiva, com uma tela de design ampla, ferramentas criativas, *templates* de interfaces predefinidos e a possibilidade de criar fluxos de navegação entre as telas, otimizando o processo de design.

O Floorplanner é um site para desenvolver plantas de projetos, e foi escolhido para fazer o mapa do museu, pois tem um fácil manuseio, e diversas ferramentas, sendo possível criar a espessura da parede, tamanho dos cômodos, altura do local, colocar móveis no local, escolher cores das paredes. sendo a ferramenta ideal para desenvolver o mapa.

Para o desenvolvimento da API, foi utilizado o Visual Studio 2022 Community com a plataforma .NET versão 8, empregando o modelo de projeto "API Web do ASP.NET Core". Essa escolha facilita a criação da API, além de contar com a integração da documentação Swagger, que simplifica a realização de testes das funcionalidades da API. A escolha do Visual Studio também se justifica pela utilização da linguagem C#, conhecida por sua versatilidade e ampla oferta de bibliotecas de código aberto.

Para a criação do banco de dados, foi utilizado o Microsoft SQL Server Management Studio (SSMS), uma ferramenta robusta e amplamente utilizada para o gerenciamento de servidores SQL. O SSMS oferece uma interface intuitiva e uma série de funcionalidades que facilitam a criação, manutenção e otimização de bancos de dados relacionais. Com ele, foi possível projetar tabelas, definir relacionamentos, criar consultas complexas e gerenciar a integridade dos dados de maneira eficiente. Além disso, a integração com o Microsoft SQL Server proporciona um ambiente seguro e escalável, adequado para suportar as necessidades da aplicação.

Para a exposição da API localmente e facilitar os testes, foi utilizado o Ngrok, uma ferramenta que cria túneis seguros para expor serviços de redes locais à internet. O Ngrok foi fundamental para testar a comunicação entre a API e o site ou dispositivos externos, sem a necessidade de configuração complexa de servidores ou redirecionamento de portas. A utilização do Ngrok permitiu que os desenvolvedores

validassem a API em um ambiente remoto, simulando as condições reais de acesso e assegura a integração adequada entre os sistemas de maneira simples e eficiente.

No desenvolvimento do sistema desktop para o totem, também foi utilizado o Visual Studio 2022 Community e a plataforma .NET versão 8, desta vez com o modelo de projeto "Aplicativo WPF". Esse modelo foi escolhido por permitir a criação de interfaces mais modernas e agradáveis, algo que seria mais limitado no modelo "Aplicativo do Windows Forms".

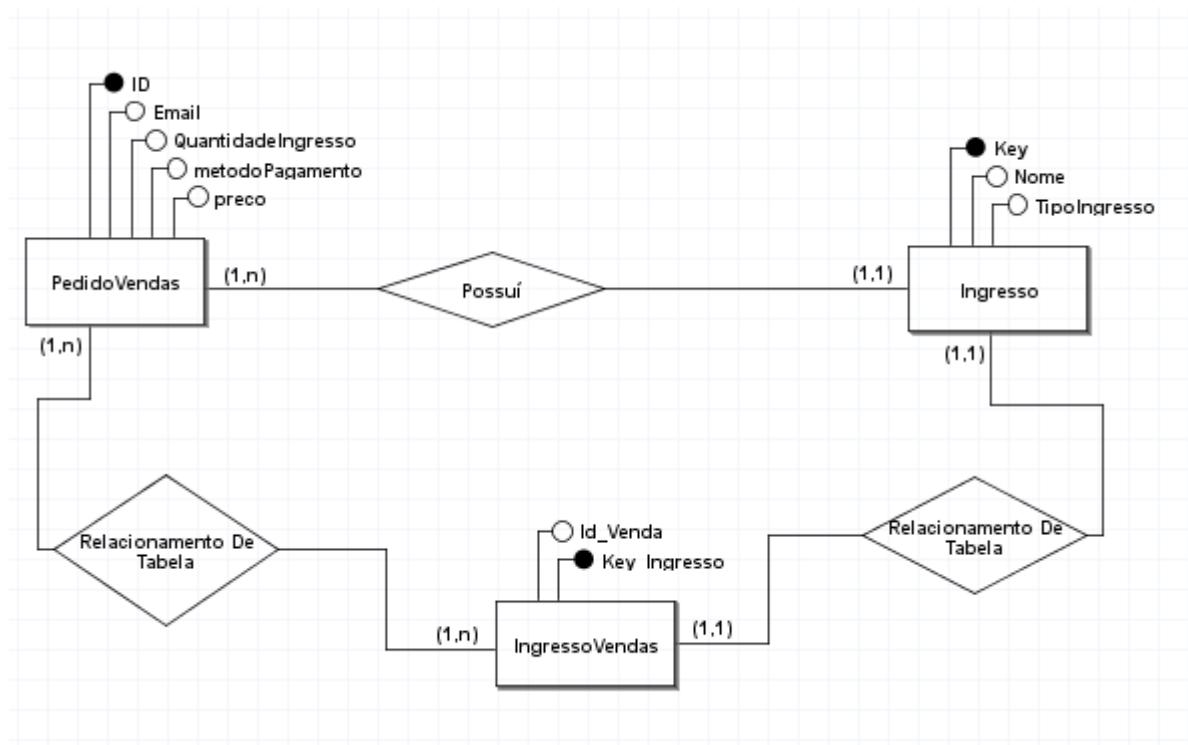
Já para o desenvolvimento do site, o Visual Studio Code foi a ferramenta escolhida, utilizando as linguagens HTML e CSS para a estrutura e estilo das páginas. A comunicação com a API foi implementada por meio de JavaScript, permitindo que as ações do site sejam executadas de forma dinâmica e integrada ao sistema.

4.9.6 Banco de Dados

Conforme mencionado anteriormente, o Microsoft SQL Server Management Studio (SSMS) foi a ferramenta escolhida para o desenvolvimento e gerenciamento do banco de dados utilizado pela API. Essa escolha se deve à robustez e confiabilidade do SSMS, que oferece uma série de recursos para a administração eficiente de dados. O banco de dados foi projetado para armazenar informações essenciais, como dados de vendas, registros de ingressos e resultados dos questionários interativos da exposição. Dessa forma, o sistema garante a segurança e organização de todas as informações, facilitando o acesso e a geração de relatórios.

O banco de dados foi projetado para armazenar informações essenciais, como o registro de todas as vendas de ingressos e os dados dos próprios ingressos, incluindo detalhes sobre o comprador e o tipo de ingresso adquirido. Além disso, ele armazena os relatórios gerados a partir dos questionários interativos preenchidos pelos visitantes, permite a análise posterior dessas informações para avaliar a experiência dos usuários e otimizar o funcionamento da exposição. Um exemplo do banco de dados pode ser visto na Figura a seguir:

Figura 8:Diagrama - Modelo de Dados



Fonte: Autoria própria

4.9.7 Configuração de Redes para API e Integração

O surgimento das redes de computadores revolucionou a organização dos sistemas computacionais. Antes, um único computador executava todas as tarefas de processamento e armazenamento de dados, devido ao alto custo das máquinas. Com avanços tecnológicos, os computadores se tornaram mais potentes, menores e acessíveis, permite sua disseminação. Isso levou a uma nova abordagem, onde múltiplos computadores são utilizados em sistemas distribuídos, em vez de depender de um único computador centralizado. (MACEDO et al, 2018).

O projeto desenvolvido foi concebido com a necessidade de integrar sistemas distintos, como o site e o sistema Totem, por meio de uma API centralizada. Essa API atua como um ponto de comunicação entre esses sistemas e o banco de dados, permite que eles compartilhem informações de forma eficiente. A interação entre os dispositivos é feita por meio de requisições HTTP, onde cada sistema pode acessar os dados e funcionalidades relevantes de maneira organizada, por meio de endpoints específicos. Esse modelo distribuído e baseado em serviços web facilita a

escalabilidade e a manutenção do sistema, uma vez que cada componente pode ser atualizado ou substituído sem impactar diretamente os outros.

Para garantir que a API pudesse ser acessada externamente de forma segura durante os testes e desenvolvimento, foi utilizada a ferramenta Ngrok. O Ngrok permite expor serviços de redes locais para a internet, criando túneis seguros que redirecionam as requisições externas para o ambiente de desenvolvimento, sem a necessidade de configurar complexos redirecionamentos de portas ou servidores públicos. Essa ferramenta foi essencial para testar a API em condições reais, simulando o acesso remoto e facilitando a validação da comunicação entre a API e outros dispositivos de maneira prática e eficaz.

A configuração do Ngrok é simples e rápida, bastando instalar a ferramenta e executá-la com um comando que especifica a porta local que se deseja expor. Após a execução, o Ngrok cria uma URL pública temporária que pode ser usada para realizar testes em dispositivos externos, permite a comunicação com a API em um ambiente real, mesmo enquanto ela está sendo desenvolvida localmente. Essa abordagem não só facilita os testes e a validação das funcionalidades da API, mas também contribui para a segurança do processo de desenvolvimento, já que o Ngrok cria uma conexão criptografada e não exige configurações complexas de firewall ou roteadores. Além disso, o Ngrok oferece relatórios detalhados de requisições, facilita a análise de tráfego e a identificação de problemas durante os testes.

4.10 Protótipo Sistema Desktop

A tela principal do sistema é o Menu, que apresenta as opções "Mapa", "Exposições" e "Questionario", conforme ilustrado na Figura a seguir:

Figura 9: Protótipo Sistema Desktop da Tela Inicial



Fonte: Autoria Própria

Ao selecionar a opção "Exposições", será exibida a lista das obras expostas do Apêndice B que estão em exposição no museu. Ao escolher uma obra, abrirá uma tela mostrando a imagem da obra com a descrição ao lado, conforme detalhado no Apêndice B, como ilustrado na Figura 10

Figura 10: Protótipo Sistema Desktop da Exposição



Fonte: Autoria Própria

Depois de o cliente ler sobre a obra desejada, ele poderá aplicar o que aprendeu. Retornando ao menu e clicando em "Questionário", ele encontrará cinco

perguntas de múltipla escolha, listadas no Apêndice A, conforme ilustrado na Figura a seguir:

Figura 11: Protótipo Sistema Desktop tela Questionario



Fonte: Autoria Própria

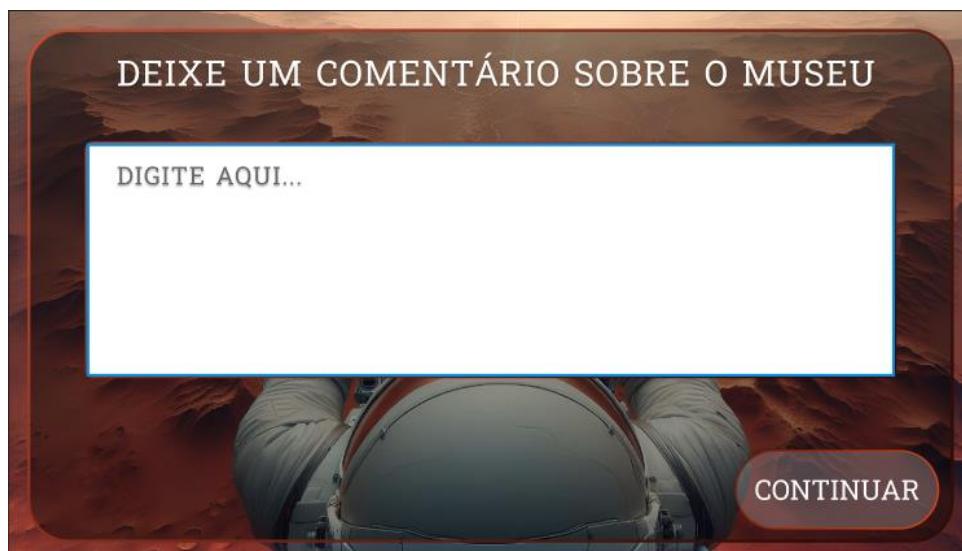
Ao finalizar o questionário, o sistema solicitará que o usuário avalie o museu, atribuindo uma nota de 1 a 5, onde cada valor é representado por uma imagem de Marte. Em seguida, uma caixa de comentários aparece, onde o usuário poderá registrar sua impressão sobre o museu ou sobre o uso do totêm. Isso é representado na Figura 12 e Figura 13.

Figura 12: Protótipo Sistema Desktop tela Avaliação Notas



Fonte: Autoria Própria

Figura 13: Protótipo Sistema Desktop tela Avaliação



Fonte: Autoria Própria

Após isso, ao clicar em "Continuar", o sistema corrigirá o questionário e exibirá as respostas corretas e incorretas. No final, será apresentada uma tela de agradecimento, conforme ilustrado na Figura 14 e, em seguida, o sistema retornará ao menu mostrado na Figura 9.

Figura 14: Protótipo Sistema Desktop da Tela de Agradecimento



Fonte: Autoria Própria

De volta ao menu, o usuário poderá selecionar a opção "Mapa" para se orientar e localizar as obras que deseja ver. Esta opção exibirá uma planta do local, conforme representado na Figura a seguir:

Figura 15: Protótipo Sistema Desktop da Tela do Mapa

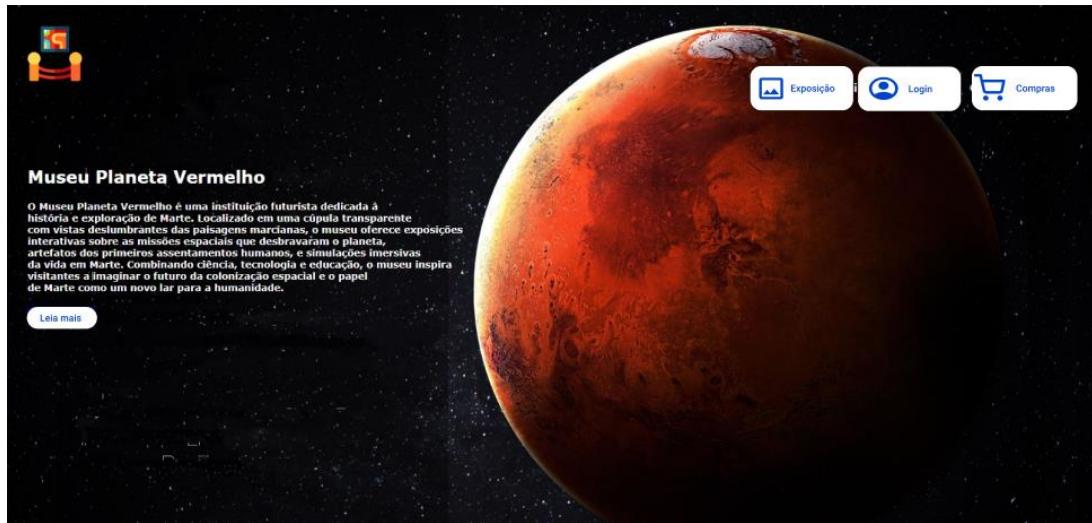


Fonte: Autoria Própria

4.11 Protótipo Site

Ao acessar o site do museu, a primeira tela que o usuário encontrará será o Menu, onde estarão as informações sobre o museu, além de dois botões: "Login" e "Comprar Ingresso", conforme ilustrado na Figura 16

Figura 16: Protótipo Site Menu



Fonte: Autoria Própria

Entrando na opção “Comprar Ingresso”, o usuário poderá escolher entre três tipos de ingresso: “Entrada Inteira”, “Meia Entrada” e “Isento”. Ao selecionar qualquer uma dessas opções, aparecerá uma caixa de texto para inserir o nome do visitante que utilizará o ingresso. Quanto mais ingressos forem adicionados, mais caixas de texto aparecerão para nomear cada ingresso, conforme mostrado na Figura 17.

Figura 17: Protótipo Site da Tela de Compra

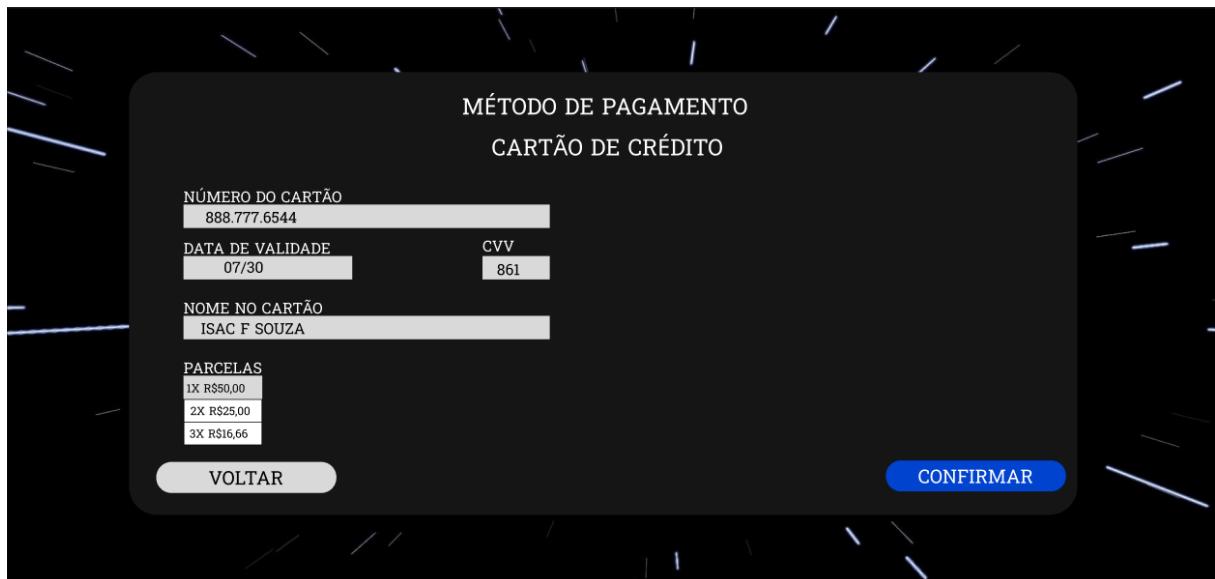


Fonte: Autoria Própria

Após escolher o ingresso, o site solicitará o método de pagamento, onde o usuário poderá inserir as informações necessárias para concluir a compra, conforme

mostrado na Figura 18. Essas informações sensíveis serão descartadas para garantir a segurança do usuário.

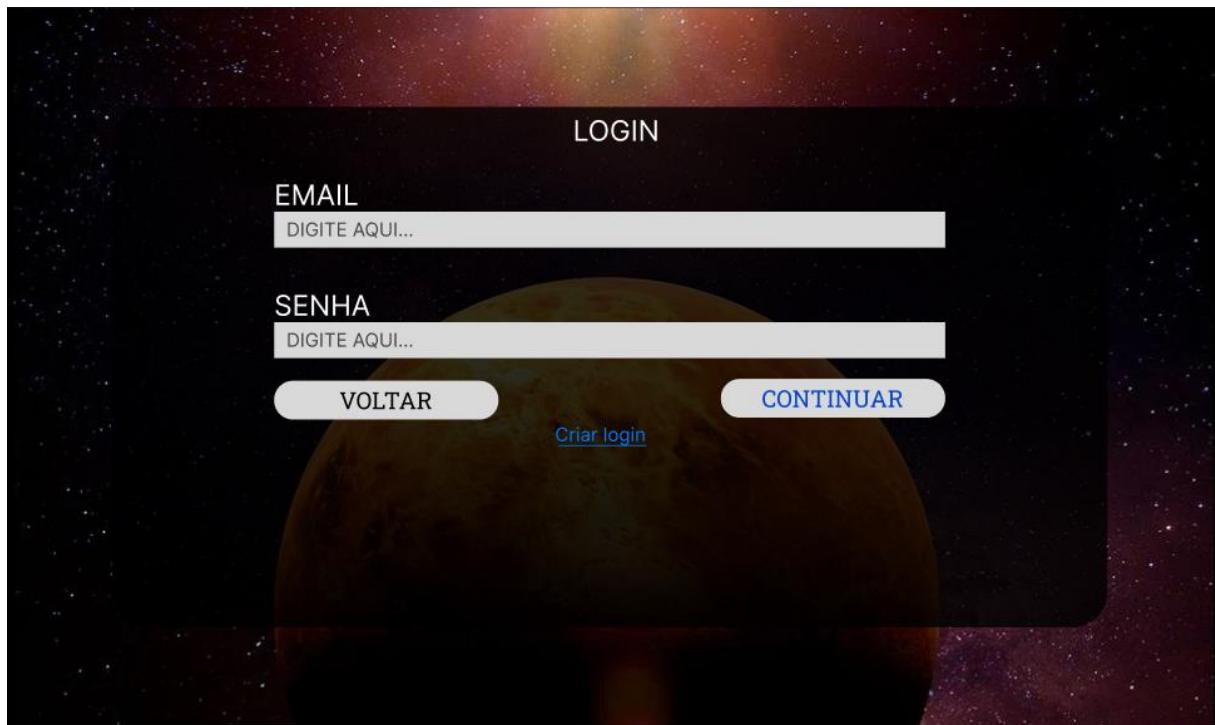
Figura 18: Protótipo Site Tela de Pagamento



Fonte: Autoria Própria

Com a compra concluída, o site exibirá os ingressos, e cada um conterá uma Key. O usuário será direcionado de volta ao menu mostrado na Figura 16. Ao acessar a opção “Login”, o usuário poderá entrar em uma conta criada após a compra do ingresso. Para fazer o login, será necessário inserir o e-mail registrado durante a compra e a Key fornecida pelos ingressos. A página de login está ilustrada na Figura 19.

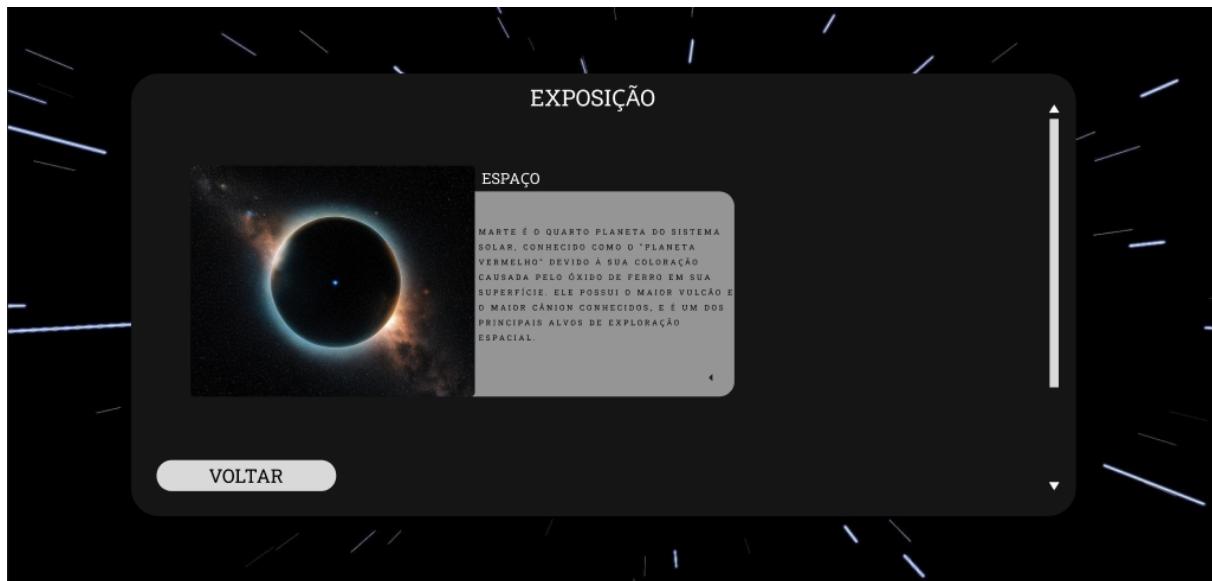
Figura 19: Protótipo Site Tela de Login



Fonte: Autoria Própria

Acessando a conta, o site retorna ao menu mostrado na Figura 16, agora com duas novas opções para o usuário: “Exposições” e “Questionário”. Ao selecionar “Exposições”, é exibido uma lista das exposições que se encontram no museu. Ao escolher uma delas, o site exibirá uma imagem maior da obra, com a descrição ao lado, conforme detalhado no Apêndice B e ilustrado na Figura a seguir:

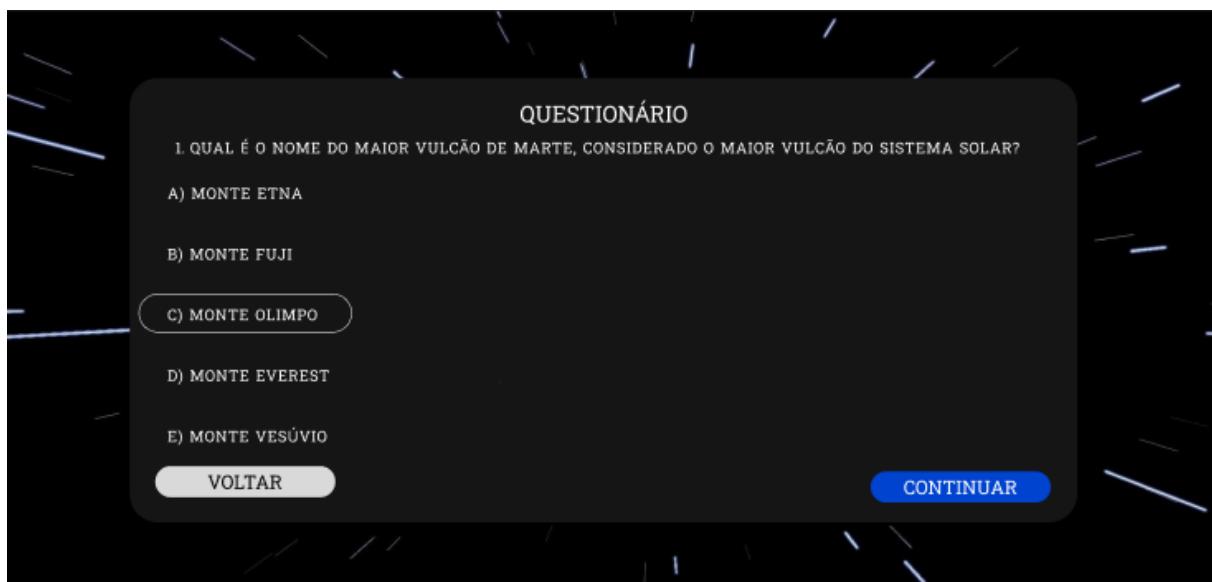
Figura 20: Protótipo Site Tela Exposição



Fonte: Autoria Própria

Após o cliente ler sobre a obra escolhida, ele poderá colocar em prática o que aprendeu. Retornando ao menu e clicando em "Questionário", ele encontrará cinco perguntas de múltipla escolha listadas no Apêndice A, junto das perguntas de avaliação, conforme ilustrado na Figura a seguir:

Figura 21: Protótipo Site Tela Questionario



Fonte: Autoria Própria

5. API

De acordo com Matthias Biehl (2015), as APIs oferecem uma maneira simples de conectar, integrar e expandir um sistema de software. Mais especificamente, as APIs são utilizadas para construir sistemas distribuídos, cujos componentes são fracamente acoplados. As APIs abordadas neste contexto são APIs web, implementadas como serviços web, que fornecem recursos de dados por meio de uma pilha de tecnologias web. Aplicações típicas que utilizam APIs incluem aplicativos móveis, aplicativos em nuvem, aplicações web e dispositivos inteligentes.

Esta API (Application Programming Interface) foi desenvolvida com o objetivo de centralizar a lógica de negócios em um único local, evita a duplicação de código tanto no sistema do Totem quanto no site. Além disso, ela facilita a interação com o banco de dados, otimiza a entrada e saída de dados, e proporciona uma maneira eficiente e padronizada de comunicação entre os sistemas e o banco de dados.

5.1 Visão Geral da API

Como dito anteriormente, a API foi desenvolvida para centralizar a lógica em um único local, evitando a duplicação de código entre os sistemas. Para garantir flexibilidade e eficiência, a lógica da API foi segmentada de acordo com as necessidades de cada plataforma. Por exemplo, as funcionalidades *ArmazenarRespostas* e *NovaVenda* foram implementadas para o site, enquanto no sistema do Totem foi implementada apenas a funcionalidade *ArmazenarRespostas*. Isso ocorre porque o Totem está localizado dentro da exposição, e, para acessá-lo, é necessário já ter adquirido um ingresso para entrar. Dessa forma, cada sistema tem acesso apenas aos serviços relevantes, de maneira organizada e otimizada.

A API também oferece funcionalidades que retornam valores essenciais para a criação de relatórios e análises. Por exemplo, funcionalidades que retornam relatórios de vendas, erros e acertos do questionário e a satisfação do cliente com o museu. Essas funcionalidades são fundamentais para coletar dados valiosos que contribuem para a elaboração de relatórios detalhados e análises que apoiam a tomada de decisões estratégicas. Elas são projetadas para fornecer informações de forma estruturada, facilita a integração com sistemas de relatórios e a visualização eficiente dos dados.

Para garantir uma interação eficiente entre a API e o banco de dados, foi utilizado o framework *Entity Framework*. Esse framework foi escolhido devido à sua capacidade de simplificar a entrada e saída de dados, proporcionando uma maneira estruturada e otimizada de realizar operações de persistência. Com o *Entity Framework*, foi possível mapear as entidades da aplicação diretamente para as tabelas do banco de dados, além de garantir que as consultas, atualizações e migrações fossem executadas de maneira eficiente e sem complexidade adicional para o desenvolvimento.

5.2 Endpoints e Funcionalidades

Um *endpoint* é uma URL específica da API que permite a comunicação entre o cliente e o servidor, executa uma função ou operação específica. Cada *endpoint* é projetado para realizar uma tarefa específica, como armazenar dados, processar informações ou gerar relatórios.

A API conta com um *controller Master*, que serve como ponto de entrada centralizado para as interações entre os sistemas e a API. O *controller Master* serve para coordenar as requisições, permite que chamadas a diferentes funcionalidades da API sejam gerenciadas de forma organizada. Ao invés de expor diretamente todos os endpoints para cada sistema, o *controller Master* direciona as requisições para os controladores responsáveis, garante que a lógica de negócios seja aplicada de forma consistente e centralizada. Essa estrutura facilita a manutenção, pois alterações na lógica de controle podem ser feitas em um único local, sem a necessidade de modificar cada controller individualmente.

No *controller Master*, a maioria das funcionalidades utiliza o método *BeginTransaction* para garantir a integridade dos dados durante operações de inserção ou alteração. Quando uma funcionalidade que altera o banco de dados é chamada, o *BeginTransaction* inicia uma transação que agrupa todas as modificações feitas. Caso ocorra algum erro no meio do processo, o método automaticamente executa um *rollback* (reversão), reverte todas as alterações feitas até aquele ponto, como se tivesse ocorrido a ação dos botões "CTRL+Z". Esse mecanismo de transação assegura que, em caso de falha, o banco de dados não fique em um estado inconsistente, evita que dados parciais ou errados sejam salvos. Esse processo é

essencial para manter a confiabilidade da API e evitar problemas decorrentes de falhas inesperadas durante a inserção de dados.

5.2.1 NovaVenda

A função *NovaVenda* é responsável por finalizar a compra de ingressos no site. Trata-se de um método POST que recebe um objeto JSON contendo dois modelos de dados. O primeiro modelo é a *venda*, que inclui informações do cliente, como o e-mail, a quantidade de ingressos desejada e o método de pagamento escolhido. O segundo modelo é o *ingresso*, onde o cliente informa o nome e escolhe o tipo de ingresso desejado. Esse modelo de ingresso pode ser repetido, permitindo que o cliente compre múltiplos ingressos, criando assim uma lista de ingressos que podem ser adquiridos em uma única transação. Segue um exemplo do JSON na Figura a seguir:

Figura 22: API - NovaVenda Json

```
{
  "ingressoDTO": [
    {
      "nome": "Eduardo",
      "tipo": "Meia"
    },
    {
      "nome": "Monica",
      "tipo": "Meia"
    }
  ],
  "venda": {
    "email": "eduardo.e.monica@gmail.com",
    "quantidade": 2,
    "metodoPagamento": "Debito"
  }
}
```

Fonte: Autoria Própria

Ao iniciar este método, o processo começa com a execução do *BeginTransaction*, conforme mencionado anteriormente. Em seguida, a classe *Validacao* é chamada para garantir que todos os campos estejam corretamente preenchidos. O e-mail, extraído do JSON, é convertido para letras minúsculas, evitando erros futuros com letras maiúsculas. A seguir, é criado um objeto *KeyGenerator*, que invoca a função *AtribuirKey*. Essa função gera uma chave de identificação para cada ingresso, composta por três elementos:

- Três primeiros caracteres do nome do ingresso
- Três primeiros caracteres do email
- Três caracteres aleatórios

A classe *KeyGenerator* converte todos os caracteres para letras maiúsculas. Em seguida, ela inverte a posição de algumas letras e realiza uma conversão para alfanumérico, ou seja, cada letra tem a possibilidade de ser transformada em um número aleatório, no final a função cria um elemento, que contém três caracteres aleatórios. Neste momento a função tem três elementos: Nome modificado, Email modificado e o elemento dos três caracteres aleatórios. A função retorna a chave única, combina esses elementos em diferentes posições. A função *AtribuirKey* realiza esse processo para cada ingresso presente na lista recebida. Para visualizar o código e obter um melhor entendimento sobre sua implementação, o código pode ser consultado no Apêndice O.

Voltando para a função *NovaVenda*, após gerar identificação em cada ingresso, a função calcula o preço da compra, em seguida é registrado a venda no banco de dados, junto do relatório de vendas onde mostra o dinheiro feito no dia, junto do relatório de ingressos onde mostra a quantidade de ingressos vendidos, e junto registra a associação das identificações de cada ingresso com o Email da venda.

No final, depois de todas as informações preenchidas, a função finalmente cria os ingressos no banco de dados e retorna todos os ingressos criados, como é mostrado na Figura 23. O código do *NovaVenda* pode ser consultado no Apêndice E.

Figura 23: API - NovaVenda Retorno

```
[  
  {  
    "nome": "Eduardo",  
    "key": "DEU975DUB"  
  },  
  {  
    "nome": "Monica",  
    "key": "539NOMD3U"  
  }  
]
```

Fonte: Autoria Própria

5.2.2 RealizarLogin

Após a conclusão da função *NovaVenda*, que retorna os ingressos comprados, o usuário pode fazer login para consultar as obras e responder ao questionário. Para isso, a API conta com a função *RealizarLogin*. Esta função é um método GET, no qual o endpoint recebe uma string contendo o Email e a chave combinados, o exemplo pode ser encontrado na Figura 24. É importante dizer que esta função não possui *BeginTransaction*, pois se trata de um método GET, onde não vai acontecer nenhuma mudança no banco de dados, o método só vai pegar valores.

Figura 24: API - RealizarLogin Endpoint

```
/api/Master/login/{emailKey}
```

Fonte: Autoria Própria

A função separa e-mail da chave logo de início e verifica se ambos os elementos estão no formato esperado pela API. Em seguida, um método é chamado para procurar a chave do ingresso na tabela que associa chaves de ingressos aos e-mails. Ao retornar o e-mail vinculado à chave, a função realiza uma comparação: se o e-mail associado à chave for o mesmo que o usuário forneceu, a função retorna um JSON, onde indica o sucesso do login, como mostrado na Figura 25. Caso contrário, um JSON de falha é retornado, juntamente com uma mensagem de erro, como ilustrado na Figura 26.

Figura 25: API - RealizarLogin Sucesso

```
{
  "success": true,
  "email": "eduardo.e.monica@gmail.com",
  "errorMessage": null
}
```

Fonte: Autoria Própria

Figura 26: API - RealizarLogin Negativo

```
{
  "success": false,
  "email": null,
  "errorMessage": "Email ou Key inválidos!"
}
```

Fonte: Autoria Própria

É interessante observar que, nesta API, o sistema busca o e-mail utilizando a chave gerada como “senha” do ingresso. Embora esse mecanismo possa não ser ideal para outras aplicações, ele funciona bem neste contexto, pois a própria API gera chaves exclusivas para cada ingresso. Como mencionado anteriormente, cada chave é realmente única: a chance de uma duplicação é extremamente baixa, pois o código do *KeyGenerator* pode produzir aproximadamente 4.5 quadrilhões de combinações possíveis. Isso torna a probabilidade de duplicação quase inexistente em um contexto prático, mesmo em um sistema de larga escala.

5.2.3 ArmazenarRespostas

Para armazenar as respostas do usuário ao questionário, a API oferece a função *ArmazenarRespostas*. Essa função é implementada como um método POST que recebe até três modelos JSON distintos. O primeiro modelo corresponde ao questionário e contém cinco objetos pré-configurados, cada um representa uma pergunta específica. Cada objeto inclui as variáveis “Acertos” e “Erros”, onde uma delas deve ter o valor 1 e a outra, 0, para registrar a resposta do usuário. O segundo modelo é dedicado à avaliação do museu, onde a função recebe três objetos de avaliação. Cada objeto representa uma pergunta e inclui cinco variáveis: “Excelente”, “Bom”, “Regular”, “Ruim” e “Péssimo”. Similar ao questionário, apenas uma dessas variáveis deve ter o valor 1, enquanto as demais devem ser 0, permite que a avaliação seja registrada adequadamente. Por fim, o último modelo opcional é o de sugestão, que contém apenas a variável “Sugestao”, na qual o usuário pode inserir um comentário livre. Caso o campo de sugestão esteja vazio, ele é removido do modelo JSON, evitando a inserção de valores nulos na tabela de sugestões e garantindo a integridade dos dados.

A função *ArmazenarRespostas* inicia com o processo *BeginTransaction*, ela garante que as operações de armazenamento sejam executadas de forma segura e

possam ser revertidas em caso de erro. Em seguida, a função chama os *controllers* específicos para o questionário e para a avaliação, responsáveis por salvar os valores nas respectivas tabelas do banco de dados. Dentro desses *controllers*, ocorre uma verificação no banco de dados para assegurar que cada pergunta esteja devidamente registrada. Caso uma pergunta ainda não esteja cadastrada, o sistema cria automaticamente os objetos correspondentes, permite que as respostas dos usuários sejam armazenadas sem interrupções. Além disso, os *controllers* verificam se os *IDs* dos objetos enviados correspondem às perguntas corretas, garantindo que as respostas sejam associadas aos registros adequados. Ao final do processo, a função retorna uma mensagem indicando que as respostas foram armazenadas com sucesso.

5.2.4 Funções de GET

Além das funcionalidades de POST, a API também implementa diversas funções de GET que permitem a obtenção de informações armazenadas no sistema. Cada uma dessas funções segue a mesma lógica básica de requisição, mas é configurada para retornar dados específicos conforme o contexto de uso. As funções de GET são projetadas para responder de forma rápida e precisa, fornecendo aos sistemas consumidores as informações necessárias para relatórios, monitoramento e feedback dos usuários.

Cada *endpoint GET* possui um nome específico que indica sua finalidade, facilita o acesso direto a conjuntos de dados distintos conforme as necessidades da aplicação, como pode ser visto na Figura 27. Essas funções foram desenvolvidas para garantir que as informações solicitadas sejam filtradas e estruturadas de forma adequada, evitando sobrecarga na rede e no banco de dados. Com essa organização, o sistema permite consultas eficientes e seguras, que assegura a consistência e a integridade dos dados retornados.

Figura 27: API - Funções GET

GET	/api/Master/QuestionarioAvaliacao/q
GET	/api/Master/QuestionarioAvaliacao/q/{id}
GET	/api/Master/QuestionarioAvaliacao/a
GET	/api/Master/QuestionarioAvaliacao/a/{id}
GET	/api/Master/QuestionarioAvaliacao/s

Fonte: Autoria Própria

5.3 Testes

Para garantir a precisão e a robustez de cada funcionalidade implementada na API, todas as funções foram rigorosamente testadas no aplicativo Postman. Cada endpoint foi submetido a diferentes cenários de teste, simulando tanto requisições válidas quanto entradas inválidas para verificar as respostas da API. Esses testes permitiram identificar e corrigir rapidamente quaisquer inconsistências, garantindo que a API respondesse de forma consistente e segura em diversos contextos. A utilização do Postman como ferramenta de teste facilitou a análise detalhada das respostas JSON, assegurando que cada função atendesse aos requisitos especificados e retornasse dados corretos e bem estruturados.

6. SISTEMA TOTEM

O sistema desenvolvido tem como objetivo fornecer informações sobre o planeta Marte e incentivar os visitantes a responderem um questionário. Além disso, o sistema gera relatórios sobre as avaliações dos usuários a respeito do museu e das perguntas respondidas.

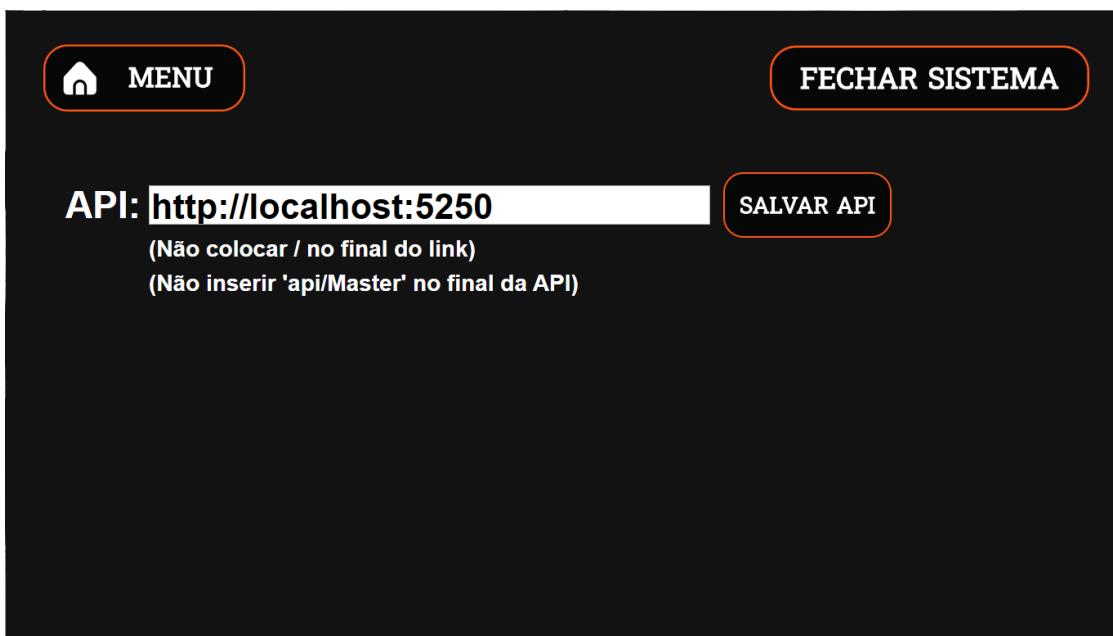
6.1 Manual de instalação

A instalação do sistema de totem é um processo simples, mas requer que uma API esteja em execução para que algumas funcionalidades funcionem corretamente durante o uso do totem. O software é distribuído em um arquivo compactado que contém uma pasta chamada “ExplorandoMarte”, junto com outros arquivos.

A pasta deve ser movida para a Área de Trabalho e, em seguida, o sistema pode ser aberto através do arquivo executável presente na pasta.

Com o sistema do totem iniciado, é necessário configurar a API. Para isso, inicie o sistema da API e copie o link onde ela está sendo hospedada. Em seguida, retorne ao sistema Totem, vá para a tela do menu e pressione a tecla Shift da direita. Isso abre uma tela com um campo para inserir o link da API, conforme ilustrado na Figura a seguir:

Figura 28: Totem - Configuração



Fonte: Autoria Própria

6.2 Introdução ao sistema

As janelas do software são configuradas com uma resolução de 1366 pixels de largura por 768 pixels de altura, garantindo que o sistema se ajuste corretamente à tela do dispositivo do totém. As janelas não possuem bordas, impedindo que os usuários as arrastem. Algumas partes das telas seguiram o estilo planejado nos protótipos, enquanto outras passaram por ajustes e adições para torná-las mais intuitivas e fáceis de usar.

Ao iniciar o sistema, é exibido o *Menu Principal (MainActivity)*. Como mostrado na Figura 29, as opções do *Menu Principal* não foram alteradas na última versão e permanecem as mesmas, sendo elas:

- Mapa
- Exposições
- Questionário

Figura 29: Totem - Menu Principal



Fonte: Autoria Própria

6.3 Mapa

Conforme mencionado, o usuário terá três opções no Menu Principal. A opção "Mapa" permite que o usuário se localize e encontre as obras desejadas. Ao selecionar essa opção, uma nova janela será aberta exibindo a planta do museu em visão superior, conforme ilustrado na Figura a seguir:

Figura 30: Totem - Tela do Mapa



Fonte: Autoria Própria

6.4 Exposições

A opção “Exposições” abre uma nova janela que exibe uma lista de todas as exposições em exibição no museu, conforme mostrado na Figura 31. Todas as exposições estão organizadas em uma única página para facilitar a navegação, evitando que o usuário precise alternar entre várias páginas.

Figura 31: Totem - Lista de Exposições



Fonte: Autoria Própria

Ao clicar na obra desejada, o sistema abre uma página nova chamada *Exposicao*, onde é exibido a imagem e a descrição da obra que o usuário selecionou, conforme é mostrado na Figura 32.

Após o usuário selecionar uma exposição na lista, é chamado a página *Exposicao*, no método construtor é inserido o Id da exposição, o método manda para a função *MudarExposicao* que faz a troca de imagem, descrição e título. Esta função usa a variável em um *switch* para exibir a imagem, descrição e título da obra. Caso o Id da obra não seja encontrado, a janela *Exposicao* é fechada, retornando para a lista de exposições.

Figura 32: Totem - Descrição da obra



Fonte: Autoria Própria

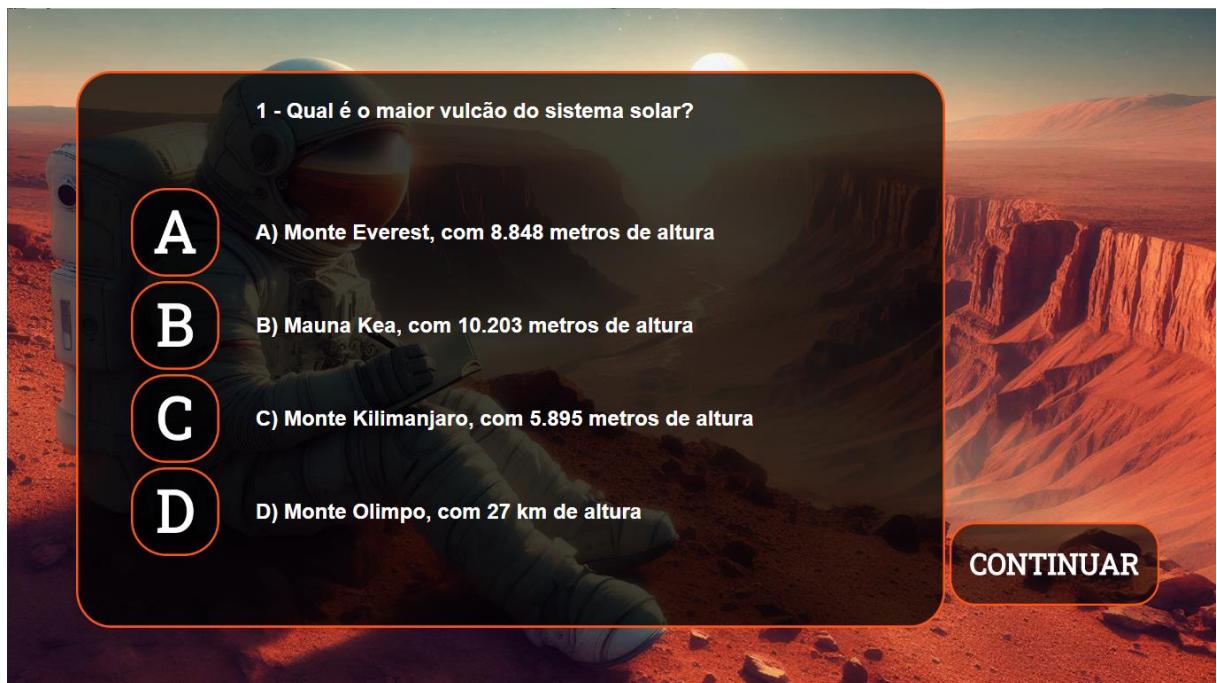
6.5 Questionario

Após ler as descrições das exposições, o usuário pode testar seus conhecimentos na opção *Questionario*.

Ao clicar na opção "Questionário" no Menu Principal, uma janela chamada Questionário é aberta, onde o usuário deve responder cinco perguntas, cada uma com quatro alternativas. O botão "Continuar" permite avançar para a próxima pergunta, desde que uma alternativa tenha sido selecionada, conforme ilustrado na Figura 33.

A janela *Questionario* começa com o método construtor que define valores vazios para algumas variáveis estáticas e chama a função *ProximaPergunta* que começa recebendo o valor 1. A lógica da função *ProximaPergunta*, é semelhante à da função *MudarExposicao* na lista de exposições. Ela recebe a variável *idPergunta* e atualiza a *Label* da pergunta e das alternativas. O botão *Continuar*, ao ser pressionado cria um objeto *Controle* que válida a resposta selecionada e, se correta, é armazenado em uma lista estática, para uma função futura. Após a validação, o *idPergunta* é acrescentado e chama *ProximaPergunta*, para atualizar a tela e continuar o questionário.

Figura 33: Totem - Questionário

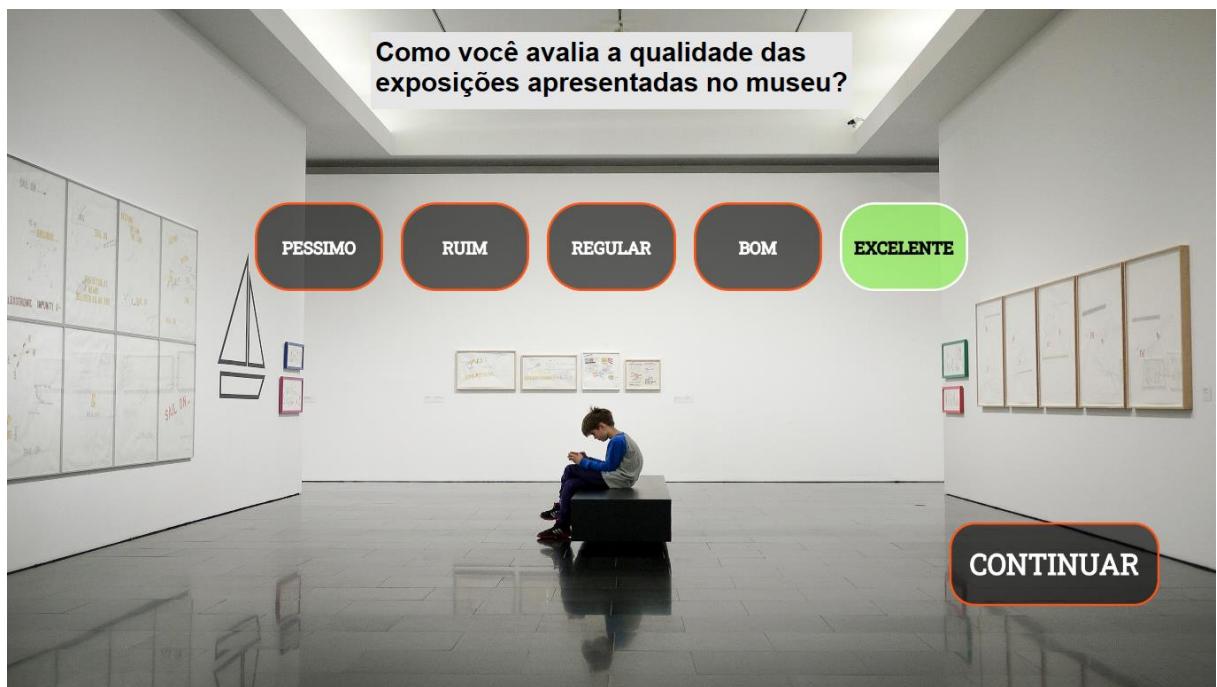


Fonte: Autoria Própria

Após a quinta pergunta, a tela muda para a janela "Avaliação". Nesta janela, o usuário encontra três perguntas sobre o museu, com alternativas de resposta como "Excelente", "Bom", "Regular", "Ruim" e "Péssimo", conforme mostrado na Figura 34. Em seguida, aparece uma caixa de sugestão com um teclado virtual no formato QWERTY, permitindo que o usuário, opcionalmente, escreva um comentário. Esta tela também está representada na Figura 35. Como o sistema é projetado para um totem touchscreen sem teclado físico, o teclado virtual é necessário para que o usuário possa registrar suas sugestões para o museu.

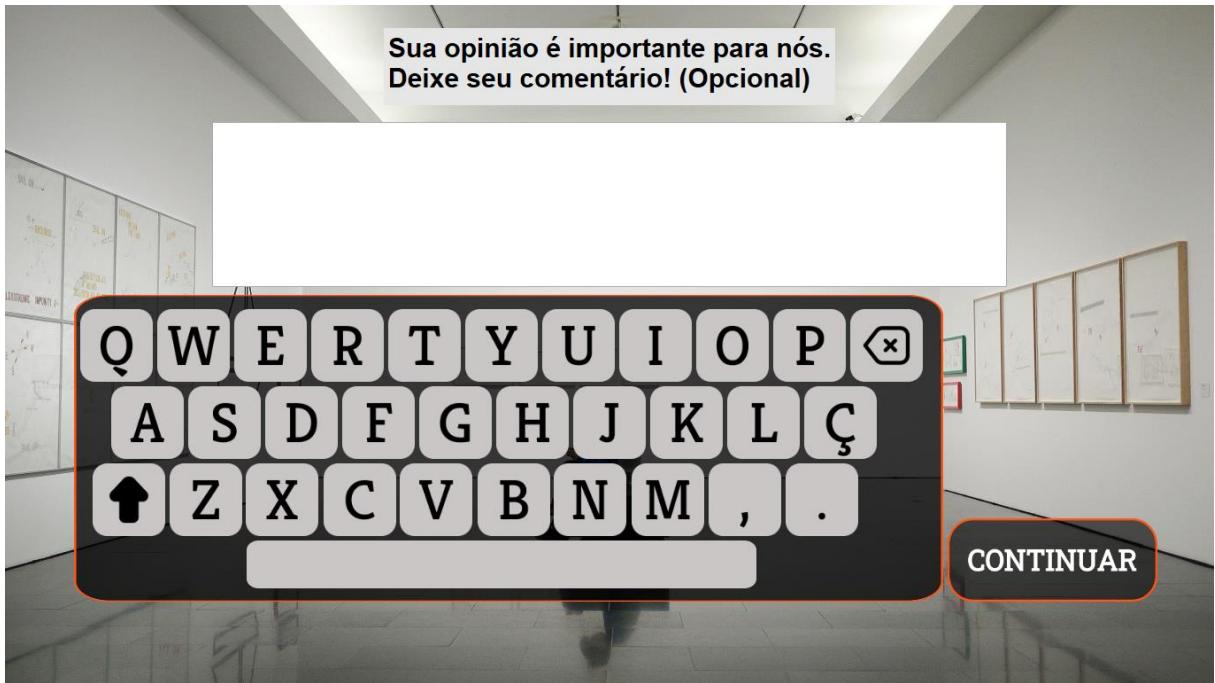
Depois do usuário avaliar o museu e deixar seu comentário com o teclado virtual, ao pressionar o botão *Continuar*, é armazenado as respostas da avaliação em uma lista estática e a sugestão em uma string estática, em seguida uma mensagem é colocada na variável “*MENSAGEMCOMUNICACAO*” e a janela é fechada, voltando para o Menu Principal. Quando a janela é fechada, o Menu verifica a variável de comunicação (“*MENSAGEMCOMUNICACAO*”), caso tenha a mensagem “ARMAZENAR”, se este for o caso, o menu chama a classe Controle e executa a função “*MontarJsonQuestionarioAvaliacao*”, onde ele vai pegar as listas estáticas onde foram armazenadas as respostas do questionário, da avaliação e a sugestão, e transformar em um modelo Json, que é aceitável para enviar na API. Caso o usuário não tenha colocado sugestão, o método cria este Json só com o questionário e avaliação, a API aceita o Json sem o objeto da sugestão.

Figura 34: Totem - Avaliação Notas



Fonte: Autoria Própria

Figura 35: Totem - Avaliação Sugestão



Fonte: Autoria Própria

Com o JSON criado, é necessário enviá-lo para a API. Para isso, utiliza-se a classe “*ControllService*”, que faz a comunicação direta com a API. Nesta classe, deve-se aplicar a configuração da API mostrada na Figura 28, feita no início do sistema. O sistema chama o método “*PutDataAsync*” e envia o JSON gerado. Este método, por sua vez, se comunica com o método da API chamado “*ArmazenarRespostas*”, mais detalhes sobre ele podem ser encontrados no Capítulo 5.2.3

Após o armazenamento bem-sucedido, uma nova janela é aberta, chamada “*RelatórioRespostas*”. Esta janela exibe as perguntas que o usuário respondeu corretamente, destacadas em fonte verde e com um ícone ao lado. Para as respostas incorretas, o texto aparece em vermelho com um ícone indicando o erro, seguido pela resposta correta logo abaixo, como demonstrado na Figura 36. Ao clicar no botão “Continuar”, são exibidas as estatísticas globais. Essas estatísticas incluem frases diferentes, três para quando o usuário acerta uma questão e três para quando erra. As frases são selecionadas aleatoriamente e exibidas ao usuário, conforme mostrado na Figura 37.

Figura 36: Totem - Respostas do Usuário



Fonte: Autoridade Própria

Figura 37: Totem - Estatística Global



Fonte: Autoridade Própria

Apertando o botão *Continuar*, é mostrado a tela de agradecimento como está na Figura 38, e o sistema volta para o *Menu Principal*

Figura 38: Totem - Tela de Agradecimento



Fonte: Autoria Própria

6.6 Teclado

A lógica do teclado no sistema do totem é simples e bem otimizada. Cada tecla possui dois IDs: o primeiro corresponde às letras minúsculas e o segundo às letras maiúsculas. Quando uma tecla é pressionada, a classe *Controle* é acionada e envia o ID da tecla para um switch, que identifica o ID e retorna o caractere correspondente. Esse caractere é então inserido diretamente na *TextBox* selecionada. O switch abrange IDs para letras minúsculas, maiúsculas, números e teclas especiais.

Além disso, o teclado conta com uma funcionalidade para alternar entre letras minúsculas e maiúsculas, além de permitir a inserção de caracteres especiais e números, proporciona ao usuário uma experiência de digitação completa e prática.

6.7 Imagens

O sistema utiliza diversas imagens que mudam conforme o usuário interage com o totem. Essas imagens são armazenadas na pasta do software, organizadas em

categorias para facilitar a programação e manutenção. As categorias incluem imagens de botões, exposições, teclas e planos de fundo.

A organização cuidadosa das imagens permite que os desenvolvedores localizem e atualizem facilmente os recursos visuais, o que melhora a eficiência do desenvolvimento. As imagens são carregadas dinamicamente conforme necessário, garantindo que o sistema seja responsivo e ofereça uma navegação suave e fluida.

6.8 Newtonsoft.Json

O *Newtonsoft.Json* é um dos frameworks mais populares para trabalhar com JSON em .NET. Ele facilita a serialização e a desserialização de dados JSON, ou seja, a conversão de objetos .NET para o formato JSON e vice-versa. Essa funcionalidade é essencial para aplicações que precisam trocar dados entre o servidor e o cliente ou com APIs, já que o JSON é um formato leve e de fácil leitura.

A serialização é o processo de converter objetos em uma representação JSON. Quando você serializa um objeto em C# usando o *Newtonsoft.Json*, você utiliza o método “*JsonConvert.SerializeObject(objeto)*” Este método recebe um objeto como entrada e retorna uma string JSON que representa esse objeto. Por exemplo quando o *Controle* executa o método “*MontarJsonQuestionarioAvaliacao*”, além de organizar, ele serializa as *List’s* dos questionários e da avaliação em um objeto, como mostra a Figura a seguir:

Figura 39: Exemplo - Serialização de Objeto

```
var request = new QuestionarioAvaliacaoRequest
{
    QuestionarioRespostas = questionarioDTO,
    AvaliacaoRespostas = avaliacaoRespostas,
    AvaliacaoSugestao = string.IsNullOrEmpty(sugestao) ? null : new AvaliacaoSugestao { Sugestao = sugestao }
};

return JsonConvert.SerializeObject(request, Formatting.Indented);
```

Fonte: Autoria Própria

O *Formatting.Indented* visto na Figura 39 é um parâmetro usado no método “*JsonConvert.SerializeObject()*” que faz a formatação da saída JSON com indentação, torna o JSON resultante mais legível. Isso significa que cada nível de aninhamento no JSON será recuado com espaços, facilitando a leitura e compreensão da estrutura do JSON.

6.9 Tempo de Ausência

A maioria das janelas do sistema possui um temporizador que monitora a inatividade do usuário. Após um período de inatividade pré-definido, o sistema fecha a janela automaticamente e retorna ao Menu Principal. Essa funcionalidade otimiza o desempenho e evita que novos usuários encontrem janelas abertas no meio do sistema, proporciona uma experiência de uso mais organizada e eficiente.

6.10 Manual de Desinstalação

A desinstalação do sistema é tão simples quanto a instalação, bastando excluir a pasta “*ExplorandoMarte*”, que contém o arquivo executável e outros arquivos. Para desinstalar o banco de dados *ExplorarMarteDB* no Microsoft SQL Server Management, abra o SQL Server Management Studio (SSMS) e conecte-se ao servidor onde o banco está localizado. No painel do “*Object Explorer*”, expanda a seção “*Databases*” para visualizar todos os bancos de dados disponíveis. Encontre o banco de dados *ExplorarMarteDB*, clique com o botão direito sobre ele e selecione a opção *deletar*. Clique em OK para concluir a exclusão. Isso removerá o banco de dados do servidor.

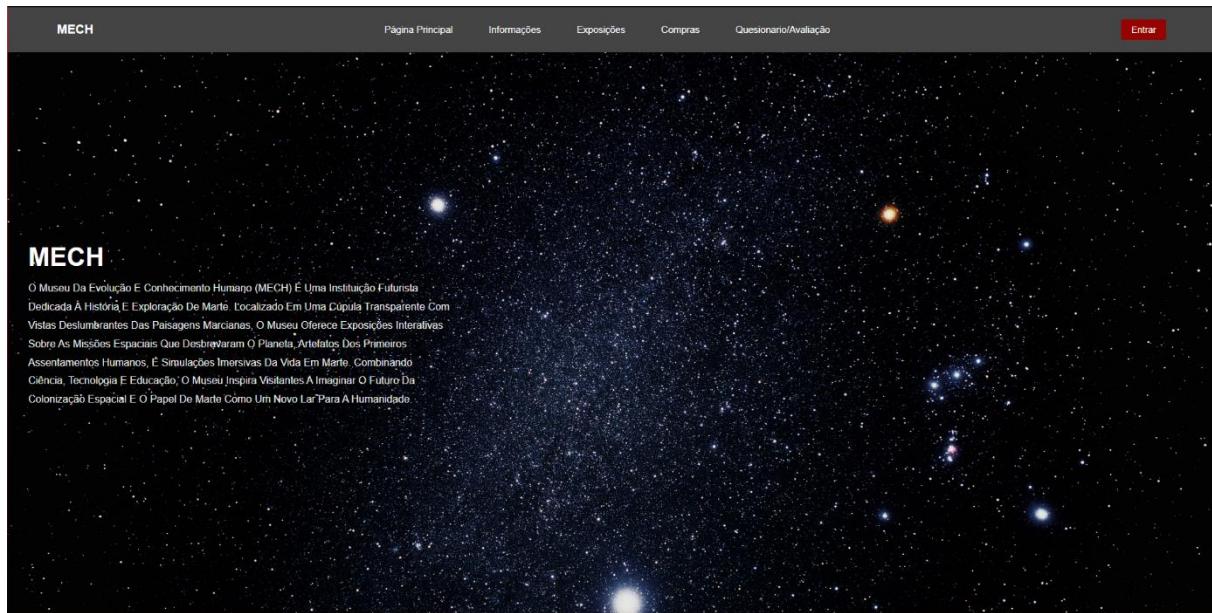
7. SITE

O site desenvolvido, tem como objetivo fazer uma introdução ao museu (MECH), oferece detalhes sobre a localização e garante seu ingresso para a exposição, tambem oferecendo uma opção portátil do sistema totem, onde é possivel ler as descrições das obras e responder a um questionario, tudo isso com uma interface clara e intuitiva. A explicação de algumas das tags usadas no projeto, se encontra do Apêndice CL.

7.1 Página Inicial

A página inicial do site como voce pode ver na Figura 40, oferece ao usuário uma interface clara e intuitiva, proporciona o acesso rápido e direto às principais informações sobre o museu. Por meio desta página, é possível navegar facilmente para seções como "Página Principal", "Mais Informações", "Compras", "Login" e "Questionário/Avaliação". Além disso, a página inicial apresenta um breve texto introdutório sobre o museu e seu acervo, convidando o visitante a explorar o conteúdo oferecido.

Figura 40: Site - Página Inicial

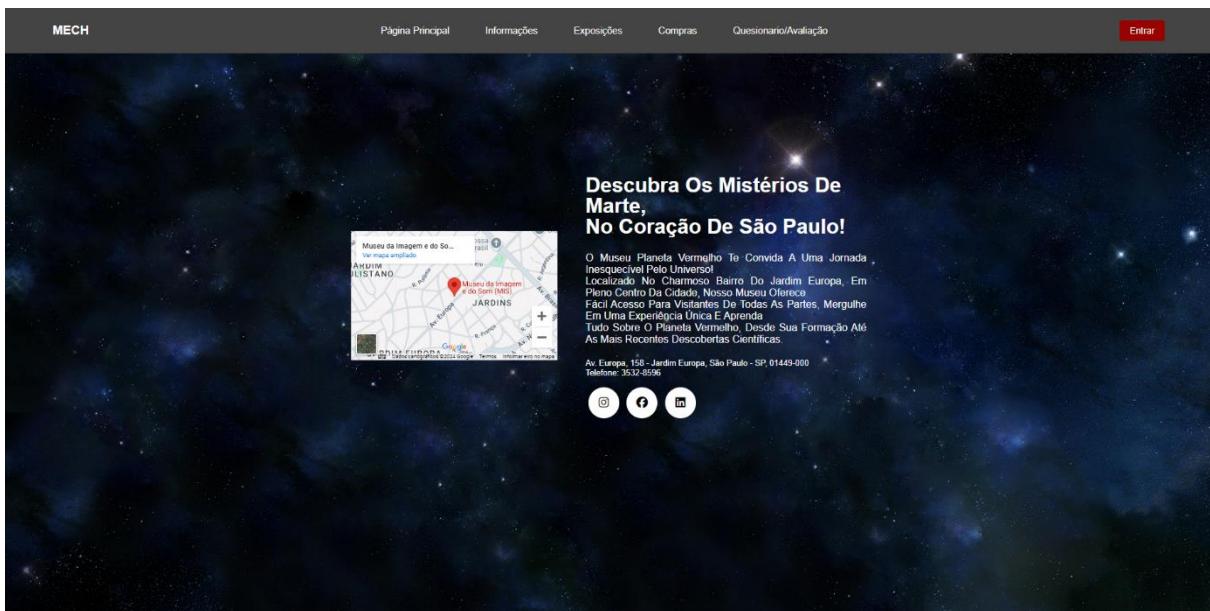


Fonte: Autoria Própria

7.2 Página de Informações

Na página de informações, o usuário encontra um resumo detalhado sobre a localização do museu e as formas de acesso disponíveis, inclui o endereço completo e um mapa interativo para facilitar a navegação. Além disso, a página oferece links para as redes sociais do Museu MECH, permite que os visitantes acompanhem atualizações, eventos e novidades diretamente nas plataformas digitais. A página de informações pode ser vista na Figura a seguir:

Figura 41: Site - Página de Informações

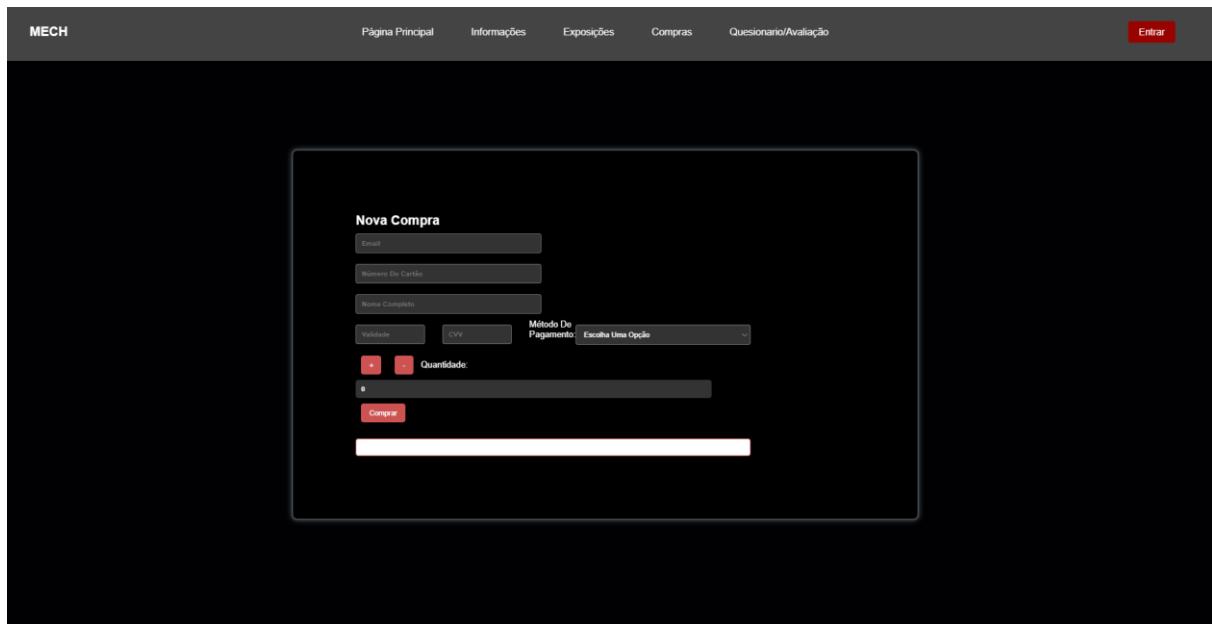


Fonte: Autoria própria

7.3 Compra

Na página de Compras, como ilustrado na Figura 42, o usuário pode adquirir ingressos para visitar o museu. Para realizar a compra, o visitante deve preencher suas credenciais, incluindo o e-mail, escolher a quantidade de ingressos desejada, selecionar o tipo de ingresso e informar o nome de cada visitante. Esta página de compras está diretamente integrada com uma função da API, que pode ser explorada no Capítulo 5.2.1. Após a confirmação da compra, o site exibe os ingressos adquiridos, cada ingresso apresenta o nome de cada comprador e a chave única de cada ingresso.

Figura 42: Site - Página de Compras

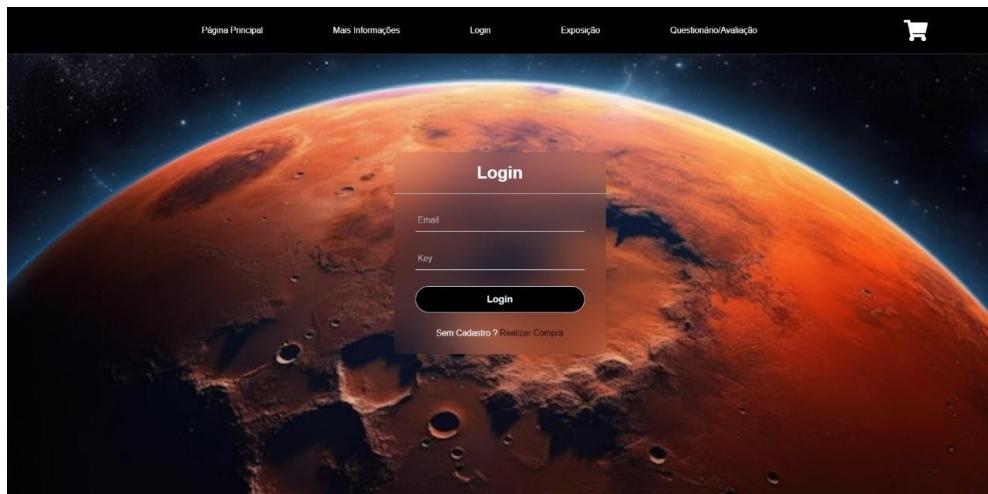


Fonte: Autoria própria

7.4 Login

Após a aquisição dos ingressos, o usuário pode acessar sua conta criada durante a compra. Para isso, basta inserir o e-mail e a chave única na página de Login, conforme ilustrado na Figura 43. Ao realizar o login com sucesso, o usuário ganha acesso à página de exposições, onde poderá visualizar as descrições detalhadas de cada obra em exibição no museu. Além disso, o login permite o acesso ao questionário interativo, proporciona uma experiência completa e dinâmica.

Figura 43: Site - Página de Login

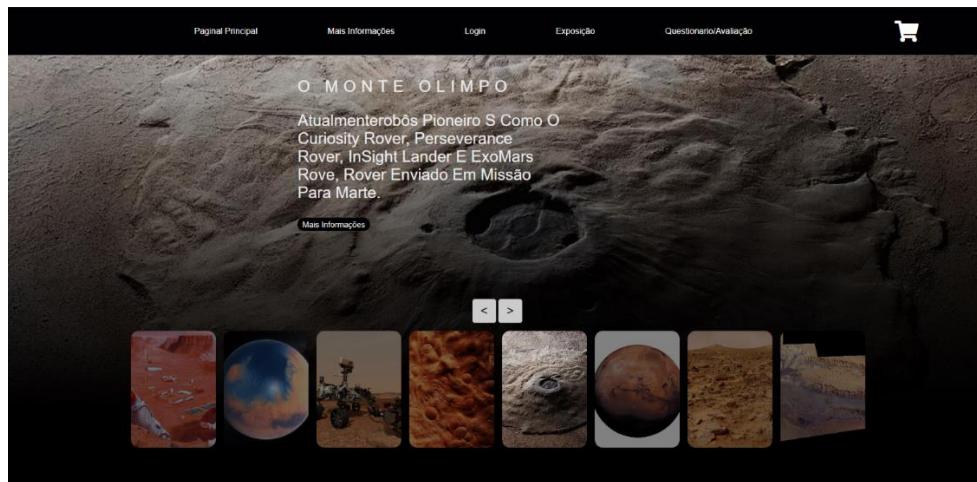


Fonte: Autoria própria

7.5 Exposições

Na página de Exposições como está ilustrado na Figura 44, o usuário tem a oportunidade de explorar as diversas exposições do museu, cada uma acompanhada por descrições detalhadas que destacam as principais características e o contexto histórico ou artístico de cada peça. Essa seção foi projetada para oferecer uma experiência rica e informativa, permite que os visitantes conheçam mais sobre o acervo antes mesmo de visitarem o museu. Além disso, a página serve como uma vitrine para o MECH, evidencia o valor cultural e educativo das suas coleções, e incentiva a interação com o público.

Figura 44: Site - Página de Exposições



Fonte: Autoria própria

7.6 Questionario

A página de questionários tem como objetivo proporcionar uma experiência interativa e dinâmica para o usuário, desafiando-o a testar os conhecimentos adquiridos durante a visita ao museu ou navegação pelo site. Como pode ser visto na Figura 45, a página possui perguntas sobre as obras em exibição e a história do planeta vermelho, o questionário visa aprofundar o aprendizado, incentiva o usuário a expandir seus conhecimentos de forma envolvente e educativa.

Esta página se utiliza de uma função da API que pode ser lida no Capítulo 5.2.3. Após concluir o questionario, o site envia as respostas para a API, onde vai guardar as informações do questionario e da avaliação.

Figura 45: Site - Página do Questionário

Questionário E Avaliação

Questionário

Qual É O Maior Vulcão Do Sistema Solar?

- Monte Kilimangaro, Com 5 895 Metros De Altura
- Mauna Kea, Com 10 203 Metros De Altura
- Monte Everest, Com 8 848 Metros De Altura
- Monte Olimpo, Com 27 Km De Altura

A NASA Tem Planos Concretos Para Enviar Missões Tripuladas Para Marte Na Década De 2030 Com A SpaceX. Quem Fundou A SpaceX?

- Neil Armstrong
- Michael "Corde" Jackson
- Elon Musk
- Sally Ride

Qual É O Fenômeno Que Deixa O Planeta Marte Com A Sua Cor Vermelha?

- O Alto Teor De Carbono Na Atmosfera

Fonte: Autoria própria

7.7 Responsividade

A responsividade em CSS é uma abordagem crucial para garantir que o layout de um site ou aplicativo se adapte corretamente a diferentes tamanhos de tela, como smartphones, tablets e desktops. No código CSS, isso geralmente é feito usando *media queries*, que permitem aplicar estilos diferentes com base nas características do dispositivo, como largura da tela, resolução, entre outras.

A primeira técnica utilizada para criar uma página responsiva é o uso de *media queries*. Essas consultas permitem que o estilo da página se ajuste conforme a largura da tela do dispositivo. Por exemplo, você pode definir um estilo específico para telas pequenas (como smartphones) e outro para telas maiores (como desktops), garantindo que os elementos na página sejam reordenados ou redimensionados de acordo com o espaço disponível. Além disso, o uso de unidades relativas como *%*, *em* ou *rem*, ao invés de unidades fixas como *px*, permite que os elementos se ajustem de forma mais flexível ao tamanho da tela.

Outra técnica importante para a responsividade é o uso de *layout fluido*. Isso pode ser alcançado com o uso de *flexbox* ou *grid*, que são sistemas de layout modernos que permitem organizar os elementos de maneira mais dinâmica e adaptável. Por exemplo, o *flexbox* pode ser usado para alinhar itens dentro de um contêiner, enquanto o CSS *Grid* pode ser utilizado para criar layouts de várias colunas

que se reorganizam automaticamente conforme a largura da tela diminui. Essas ferramentas, combinadas com *media queries*, oferecem uma maneira poderosa de criar páginas e aplicativos que funcionam bem em qualquer dispositivo, sem a necessidade de múltiplas versões do código. Os exemplos de responsividade estão ilustrados na Figura 46, Figura 47 e Figura 48.

Figura 46: Site - Página de Exposições - Responsividade



Fonte: Autoria própria

Figura 47: Site - Página do Questionario - Responsividade



Fonte: Autoria própria

Figura 48: Site - Página Principal - Responsividade



Fonte: Autoria própria

CONCLUSÃO

Este trabalho apresentou uma proposta inovadora no contexto de museus multitemáticos, ao integrar a tecnologia em um ambiente cultural de maneira eficaz. O Museu da Evolução e Conhecimento Humano (MECH), ao desenvolver uma exposição sobre a exploração de Marte, não apenas ofereceu um olhar profundo sobre o planeta vermelho, mas também incorporou soluções tecnológicas que ampliam a experiência do visitante. A criação do totêm interativo e do site responsivo reflete uma abordagem moderna, onde a interação do público com o conteúdo é facilitada e enriquecida por meio de interfaces digitais.

O desenvolvimento do software para o totêm e do site teve como princípio garantir a interatividade e a educação de forma dinâmica. Utilizando tecnologias de ponta, como a plataforma .NET e o banco de dados SQL Server, o projeto foi concebido para oferecer uma experiência fluida e eficiente aos usuários. A integração entre o sistema do totêm, o site e a API centralizada permitiu uma comunicação contínua entre as diferentes plataformas, otimizando a interação e o compartilhamento de dados entre elas.

No que diz respeito à arquitetura do sistema, o uso de metodologias ágeis garantiu um desenvolvimento mais flexível e adaptável às mudanças de requisitos durante o ciclo de vida do projeto. A aplicação do framework Entity Framework na gestão de dados possibilitou uma estrutura robusta e eficiente, facilitando o acesso e a manipulação de informações críticas, como registros de ingressos e resultados dos questionários interativos. A API foi planejada para centralizar a lógica de negócios, evitando duplicações e assegurando que as funcionalidades fossem distribuídas de forma organizada e de fácil manutenção.

A escolha de utilizar o Ngrok para expor a API localmente permitiu a realização de testes com condições de acesso reais, garantindo que a comunicação entre os sistemas fosse eficiente e segura, sem a necessidade de configurações complicadas de servidores. Esse processo de validação contínua foi crucial para assegurar a integridade do sistema, garantindo que todas as funcionalidades do site e do totêm interativo funcionassem conforme o esperado, especialmente em um ambiente de alta interatividade como o de um museu.

A implementação do sistema de totêm interativo, com a funcionalidade de mapa e questionários, acrescentou uma camada de interatividade que foi essencial para a

aprendizagem dos visitantes. A possibilidade de os usuários interagirem diretamente com o conteúdo, desafiando seu conhecimento sobre Marte e explorando as exposições de forma personalizada, não só enriqueceu a visita, mas também forneceu à equipe do museu dados valiosos sobre o engajamento do público, permitindo ajustes e melhorias contínuas na experiência oferecida.

O site, por sua vez, foi desenvolvido com uma interface intuitiva, oferecendo aos usuários a facilidade de realizar compras de ingressos de forma rápida e segura. Além disso, o acesso ao conteúdo do museu e a participação nos questionários interativos proporcionaram uma experiência contínua e envolvente. A coleta de dados, por meio de análises dos questionários, permitiu gerar relatórios detalhados sobre o comportamento e a satisfação dos visitantes, contribuindo para decisões estratégicas sobre a duração e a organização das exposições.

O planejamento cuidadoso, aliado a uma execução bem estruturada, garantiu que os requisitos funcionais e não funcionais do projeto fossem atendidos de forma eficaz. A definição clara das responsabilidades, as estratégias de integração entre os sistemas e o uso de ferramentas como o Figma e o Trello contribuíram para um gerenciamento de projeto eficiente, com a equipe alinhada e focada no cumprimento dos prazos estabelecidos. As funcionalidades adicionais do site, como o acesso a contas e o gerenciamento de ingressos, ampliaram a acessibilidade, tornando o processo de compra e a visita ao museu ainda mais convenientes para o público.

Por fim, o desenvolvimento deste projeto reflete o poder da tecnologia na transformação da maneira como os museus se relacionam com o público. Ao integrar recursos como totem interativo, site responsável e API centralizada, o MECH não apenas ofereceu uma experiência mais dinâmica, mas também se posicionou como um exemplo de como a inovação pode ser aplicada no setor cultural. O impacto desse projeto é claro: ele proporciona um ambiente mais imersivo e educacional, ao mesmo tempo em que coleta dados essenciais para melhorar continuamente a experiência do visitante. Este modelo tecnológico não só é uma ferramenta educativa, mas também um catalisador para o avanço da interação entre o público e o conteúdo museológico.

BIBLIOGRAFIA

ABNT - Associação Brasileira de Normas Técnicas. Engenharia de software – Qualidade de produto. Parte 1: Modelo de qualidade. Disponível em: <https://jkolb.com.br/wp-content/uploads/2014/02/NBR-ISO_IEC-9126-1.pdf>. Acesso em: 02.04.2024.

ISO/IEC 9126-1. 2001. **International Standard. Information Technology. Software engineering – Product quality – Part 2: External metrics.** 2001.

GIL, Antônio Carlos. **Como elaborar projetos de pesquisa.** 4. ed. São Paulo: Atlas, 2002.

KOCHANSKI, DJONE. Engenharia de Software. Disponível em: <<https://www.uniasselvi.com.br/extranet/layout/request/trilha/materiais/livro/livro.php?codigo=14602>>. Acesso em 27 de março de 2024

SORDI, J. O. Modelagem de Dados: Estudos de Casos Abrangentes da Concepção Lógica à Implementação. 1. ed. São Paulo: Editora Saraiva, 2020.

BOOCHE, Grady; RUMBAUGH, James; JACOBSON, Ivar. UML: guia do usuário. 2. ed. Rio de Janeiro: Elsevier, 2005.

MACEDO, R. T. FRANCISCATTO, R. CUNHA, G. B. BERTOLINI, C. **Licenciatura em Computação – Redes de Computadores.** Santa Maria-RS, 2018.

PEREIRA W. A gente precisa ver o Luar. In: COLONESE, P. (Org.) Os Mensageiros das Estrelas: Sistema Solar, volume 6, – Rio de Janeiro: Fiocruz – COC, 2021. P. 13-33. Disponível em: <https://museudavida.fiocruz.br/images/Publicacoes_Educacao/PDFs/OMESSolar2021vol6.pdf>. Acesso em 22 de março. 2024.

PRESSMAN, Roger; MAXIM, Bruce. Engenharia de Software: Uma Abordagem Profissional. Porto Alegre: AMGH Editora Ltda, 9^a ed., 1996. Disponível em: <https://www.academia.edu/89376481/PRESSMAN_Engenharia_de_software_Uma_Abordagem_Profissional_9a_Ed>. Acesso em 20 mar. 2024.

PROJECT MANAGEMENT INSTITUTE (PMI). **Um Guia do Conhecimento em Gerenciamento de Projetos: Guia PMBOK.** Newtown Square, PA: PROJECT MANAGEMENT INSTITUTE, 2017.

SOMMERVILLE, IAN. Engenharia de Software. 2011. Disponível em: <<https://www.uniasselvi.com.br/extranet/layout/request/trilha/materiais/livro/livro.php?codigo=14602>>. Acesso em 27 de março de 2024.

MESA DIGITAL TOUCH SCREEN 32 POLEGADAS – Index Soluções: Disponível em <https://www.solucoesindex.com.br/mesa-digital-touchescreen-32-polegadas-idx-table>. Acesso: 16/10/2024

TEIXEIRA, Fabricio. **Introdução e boas práticas em UX Design.** [s.l.]: Editora Casa do Código, 2014.

BIEHL, Matthias. **API Architecture.** [s.l.]: API-University Press, 2015.

MARTINS, Z. Procura de vida em marte: Futuras missões ao planeta vermelho. Disponível em: <<https://bdigital.ufp.pt/bitstream/10284/2396/3/157-164.pdf>> Acesso: 03/09/2024.

NASA. "Perseverance Rover: Exploring Mars." Disponível em: <<https://mars.nasa.gov/msl/>> Acesso em: 03/09/2024.

NASA. Mariner 4: The First Mars Mission. Disponível em: <<https://ntrs.nasa.gov/api/citations/20030032487/downloads/20030032487.pdf>> Acesso em: 05/09/2024.

RAMIREZ. R.M. A warmer and wetter solution for early Mars and the challenges with transient warming. Icarus 297. New York, 2017.

FURLAN, José Davi. Modelagem de Objetos através da UML- the Unified Modeling Language. São Paulo: Makron Books, 1998.

APÊNDICE A - PERGUNTAS E RESPOSTAS

Qual é o maior vulcão do sistema solar?

- A) Monte Everest, com 8.848 metros de altura
- B) Mauna Kea, com 10.203 metros de altura
- C) Monte Kilimanjaro, com 5.895 metros de altura
- D) Monte Olimpo, com 27 km de altura

(RESPOSTA: D)

A NASA tem planos concretos para enviar missões tripuladas para Marte na década de 2030 com a SpaceX. Quem fundou a SpaceX?

- A) Michael "Lorde" Jackson
- B) Neil Armstrong
- C) Elon Musk
- D) Sally Ride

(RESPOSTA: C)

Qual é o fenômeno que deixa o planeta Marte com a sua cor vermelha?

- A) O alto teor de carbono na atmosfera
- B) A presença de óxido de ferro em sua superfície
- C) A proximidade do Sol, causando oxidação intensa
- D) A presença de muito planeta deixou com vergonha

(RESPOSTA: B)

Qual descoberta sobre Marte pode indicar a possibilidade de vida no passado?

- A) Evidências de água líquida em rios e lagos secos.
- B) A presença de metano na atmosfera.
- C) Traços de fósseis de pequenos organismos.
- D) A descoberta de grandes depósitos de gelo.

(RESPOSTA: A)

Qual das seguintes tecnologias será necessária para explorar o Monte Olimpo em Marte?

- A) Foguetes de propulsão solar.
- B) Módulos infláveis para habitats humanos.
- C) Missões robóticas para mapear a superfície.
- D) Satélites com câmeras de alta resolução para sobrevoos contínuos.

(RESPOSTA: C)

APÊNDICE B - EXPOSIÇÕES E DESCRIÇÕES

O Planeta Vermelho

Figura 49: Exposição - O Planeta Vermelho

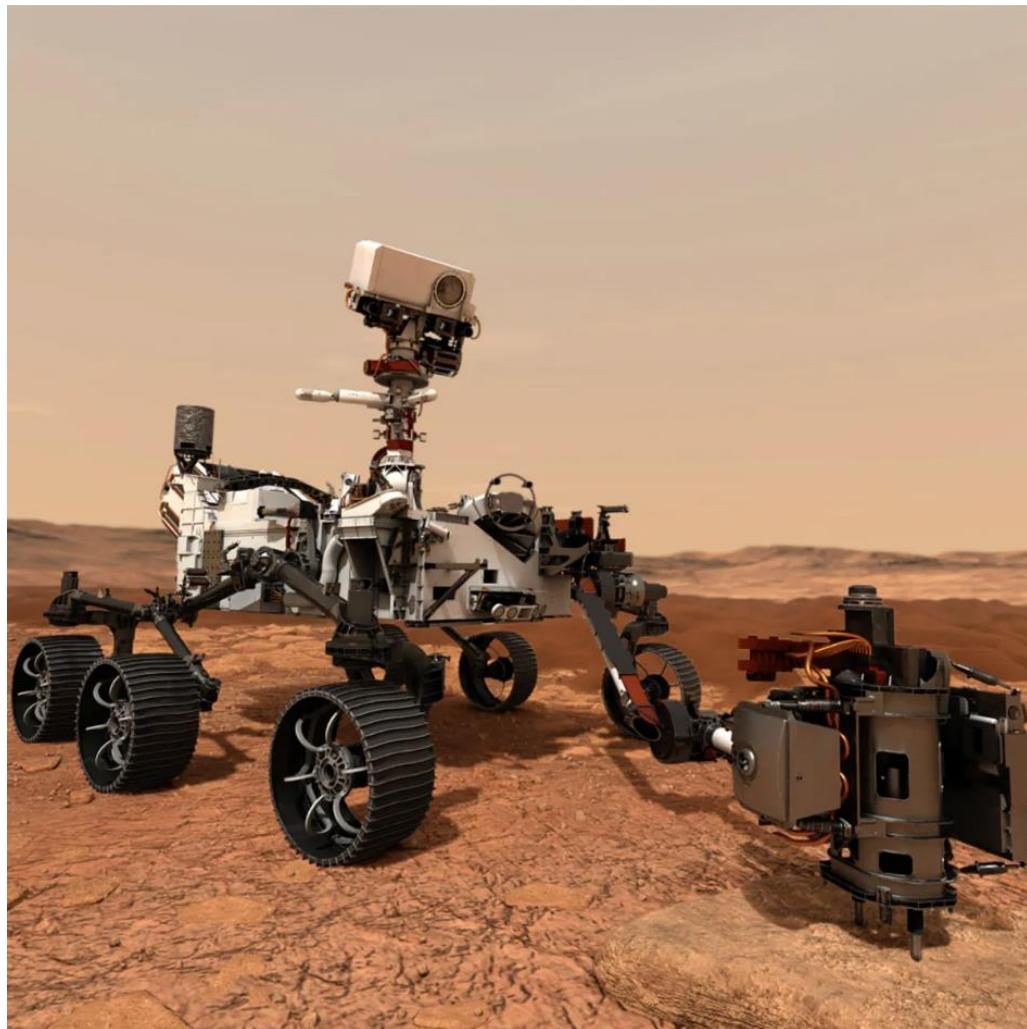


Fonte: <https://www.nationalgeographicbrasil.com/espaco/2023/11/marte-tem-diversas-semelhancas-com-a-terra-descubra-4-curiosidades-sobre-o-planeta>

Marte é conhecido por sua característica cor vermelha, um traço distintivo que fascina cientistas e astrônomos. Essa tonalidade é resultado da presença de óxido de ferro em sua superfície, um produto da oxidação intensa do ferro presente na terra marciana. Em comparação com a Terra, a oxidação em Marte é muito mais acentuada, conferindo ao planeta sua aparência única e deslumbrante.

Exploração e Potencial para Vida

Figura 50: Exposição - Exploração e Potencial para Vida

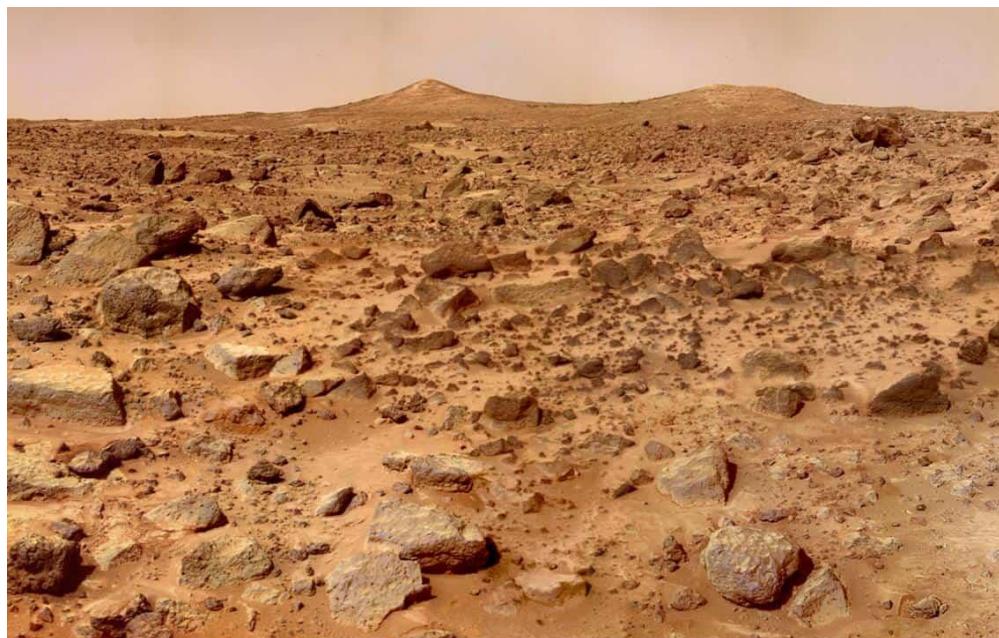


Fonte: <https://www.cnnbrasil.com.br/tecnologia/perseverance-completa-tres-meses-em-marte-veja-as-melhores-imagens/>

A exploração de Marte é fundamental para desvendar os segredos do sistema solar. Este planeta vermelho, com sua superfície árida e paisagens deslumbrantes, é um alvo fascinante para cientistas e engenheiros. Atualmente, robôs pioneiros como o Curiosity Rover, Perseverance Rover, InSight Lander e ExoMars Rover estão explorando Marte, coletando dados sobre sua geologia, composição do solo, clima, atmosfera e potencial para água líquida.

O Terreno Marciano

Figura 51: Exposição - O Terreno Marciano

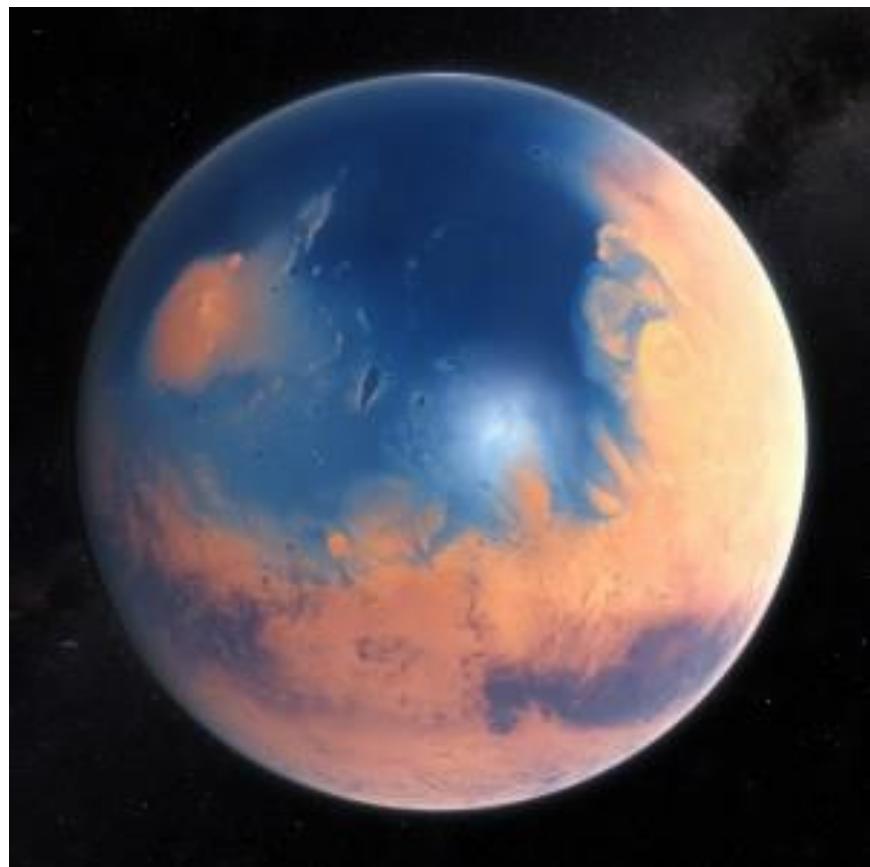


Fonte: <https://olhardigital.com.br/2020/09/03/ciencia-e-espaco/nasa-divulga-impressionante-foto-de-marte-feita-ha-23-anos/>

Marte apresenta uma paisagem diversificada, marcada por vales profundos, crateras gigantescas e vulcões imponentes. O mais notável deles é o Olympus Mons, o maior vulcão do sistema solar, com uma altura impressionante de 27 km. A superfície de Marte é dividida em duas regiões principais: a planície setentrional, caracterizada por suas amplas extensões planas, e a região montanhosa meridional, repleta de montanhas e vales.

Água em Marte

Figura 52: Exposição - Água em Marte

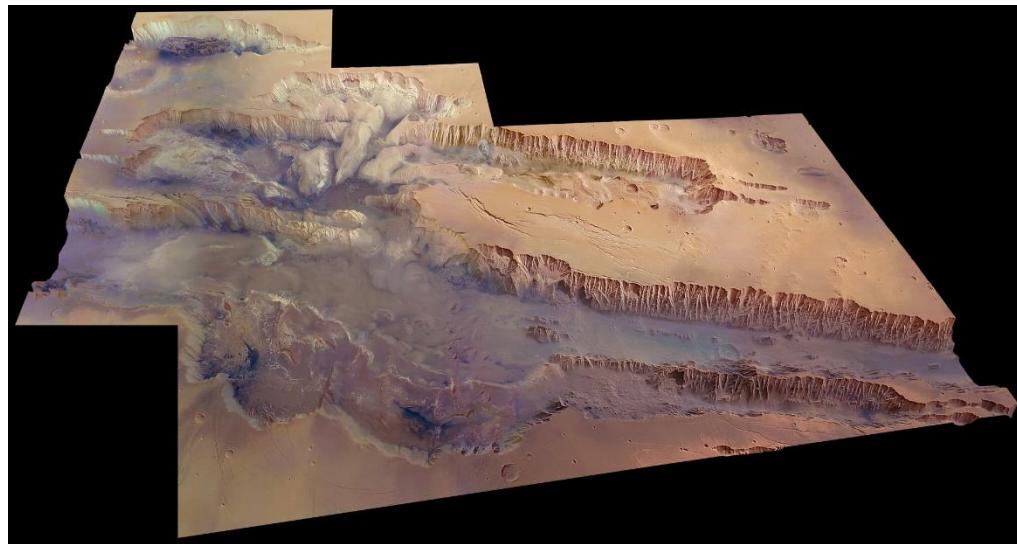


Fonte: <https://jornal.usp.br/atualidades/agua-liquida-em-marte-qual-o-tamanho-dessa-descoberta/>

A possibilidade de vida em Marte é um tema intrigante e complexo. A água é essencial para a vida como conhecemos, e sua presença é um fator crítico para o desenvolvimento de organismos vivos. A NASA e outras agências espaciais já encontraram evidências de que Marte teve água líquida no passado, como rios e lagos secos, deltas de rios e minerais hidratados. Essas descobertas sugerem que o planeta pode ter tido condições favoráveis à vida no passado.

Valles Marineris

Figura 53: Exposição - Valles Marineris



Fonte:

https://www.esa.int/Science_Exploration/Human_and_Robotic_Exploration/Exploration/ExoMars/ExoMars_discovers_hidden_water_in_Mars_Grand_Canyon

Valles Marineris, o maior canyon do sistema solar, é um verdadeiro marco natural em Marte, estendendo-se por uma impressionante distância de 4.000 km. Para colocar essa magnitude em perspectiva, é quatro vezes mais longo do que o Grand Canyon, um dos mais icônicos monumentos naturais da Terra. Sua profundidade e vastidão são um testemunho da força erosiva do vento e da água que, no passado, fluíram em Marte.

O Monte Olimpo

Figura 54: Exposição - O Monte Olimpo



Fonte: https://www.nationalgeographic.pt/ciencia/hoje-e-enorme-vulcao-marte-mas-podera-tempos-ter-sido-ilha_4114

O Monte Olimpo, localizado em Marte, é um verdadeiro gigante, com seus impressionantes 27 km de altura, superando o Monte Everest em cerca de três vezes. Explorar esse vulcão seria um desafio emocionante e complexo. Para explorar o Monte Olimpo, seria necessário desenvolver tecnologias avançadas e estratégias inovadoras. Em primeiro lugar, missões robóticas seriam essenciais para mapear a superfície e coletar dados sobre a geologia e composição do vulcão. Isso ajudaria a identificar áreas de interesse e riscos potenciais.

Impacto de Asteroides

Figura 55: Exposição - Impacto de Asteroides



Fonte: <https://olhardigital.com.br/2022/01/26/ciencia-e-espaco/cientistas-usam-algoritmo-para-saber-frequencia-de-impactos-de-asteroides-que-formaram-as-crateras-de-marte/>

Marte, o Planeta Vermelho, é um testemunho vivo da violência cósmica que marca a história do sistema solar. Sua superfície é pontuada por inúmeras crateras de impacto, resultado da colisão com asteroides e cometas ao longo de milhões de anos, variando em tamanho desde pequenas depressões até gigantescas cavidades como a Cratera Hellas, com mais de 2.200 km de diâmetro. Esses impactos fornecem informações valiosas sobre a composição e estrutura geológica de Marte, além de sugerir que a água pode ter sido trazida por cometas e asteroides.

Colonização de Marte

Figura 56: Exposição - Colonização de Marte

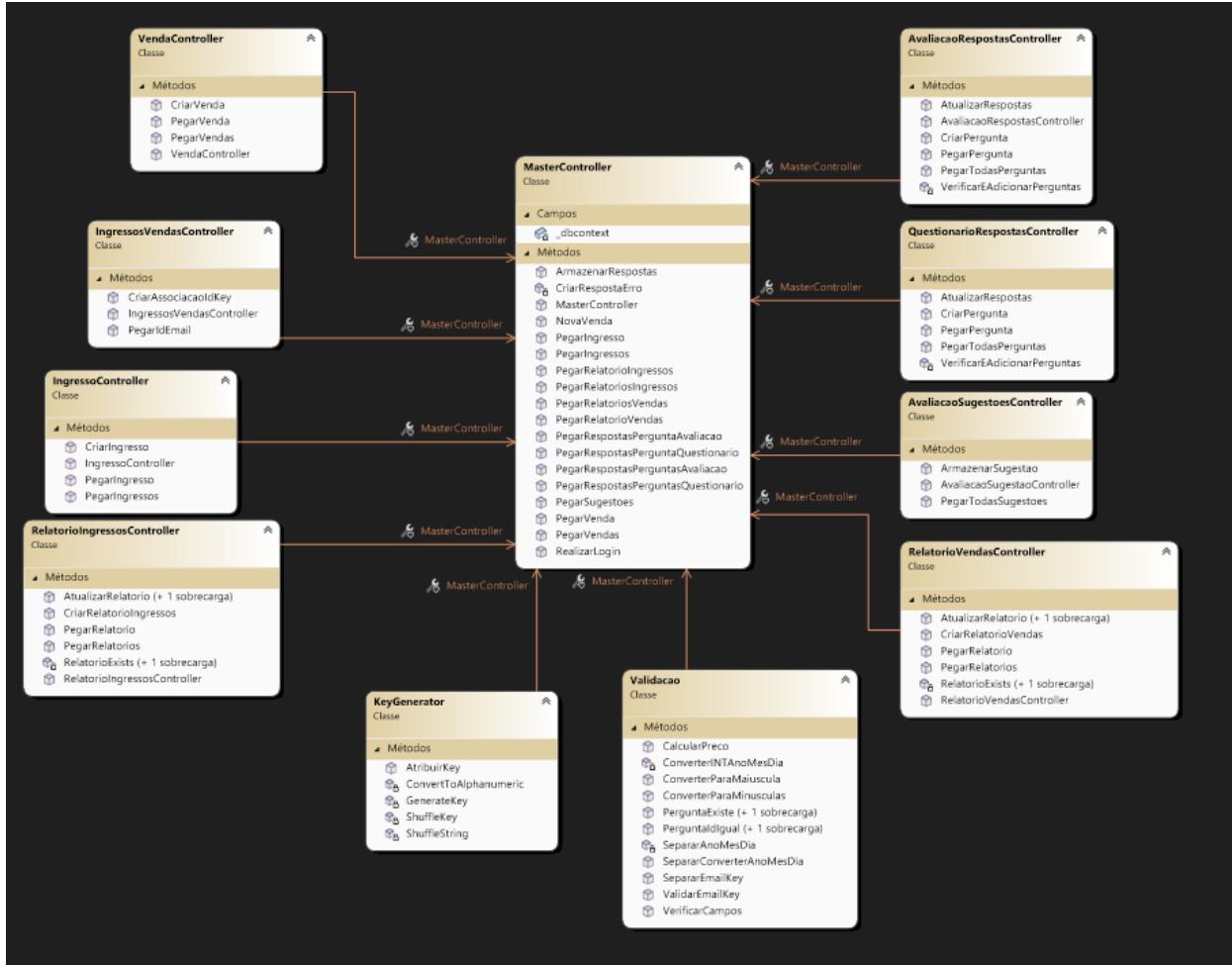


Fonte: <https://olhardigital.com.br/2016/01/28/ciencia-e-espaco/para-escritor-colonizar-marte-permitiria-criar-um-backup-da-humanidade/>

A colonização de Marte é um objetivo ambicioso que tem capturado a imaginação de científicos, engenheiros e visionários por décadas. A NASA e outras agências espaciais já têm planos concretos para enviar missões tripuladas a Marte na década de 2030, com a SpaceX, fundada por Elon Musk, trabalhando arduamente para desenvolver tecnologias necessárias para uma colonização sustentável. Com a tecnologia avançando rapidamente, a possibilidade de estabelecer uma comunidade humana no Planeta Vermelho está se tornando cada vez mais realista. No entanto, a colonização de Marte não será fácil.

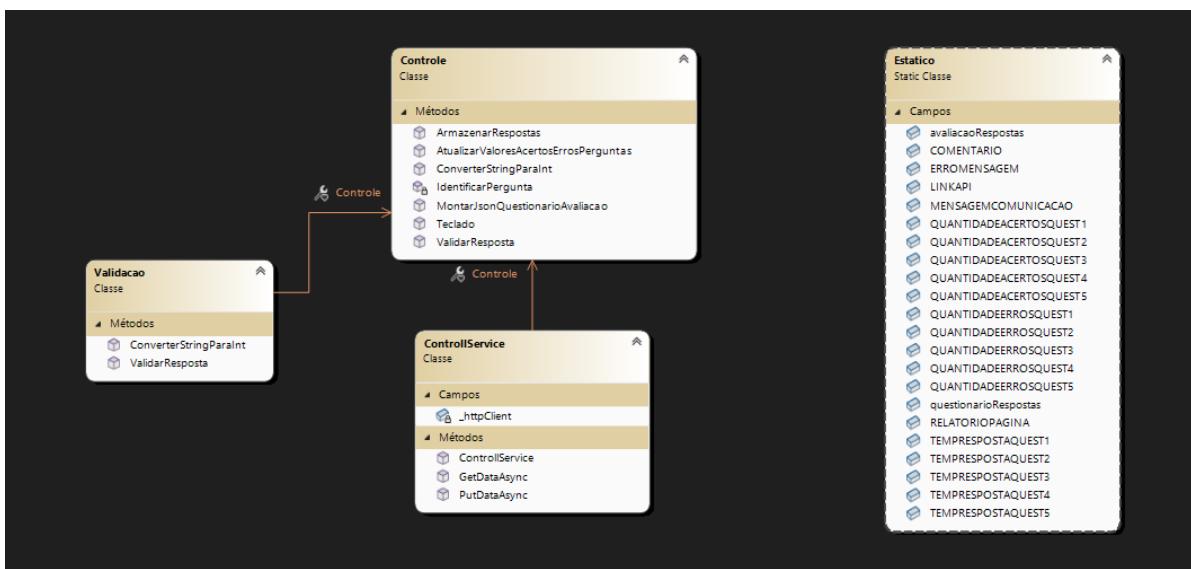
APÊNDICE C – DIAGRAMAS DE CLASSE

Figura 57: Diagrama - Diagrama de Classe API



Fonte: Autoria Própria

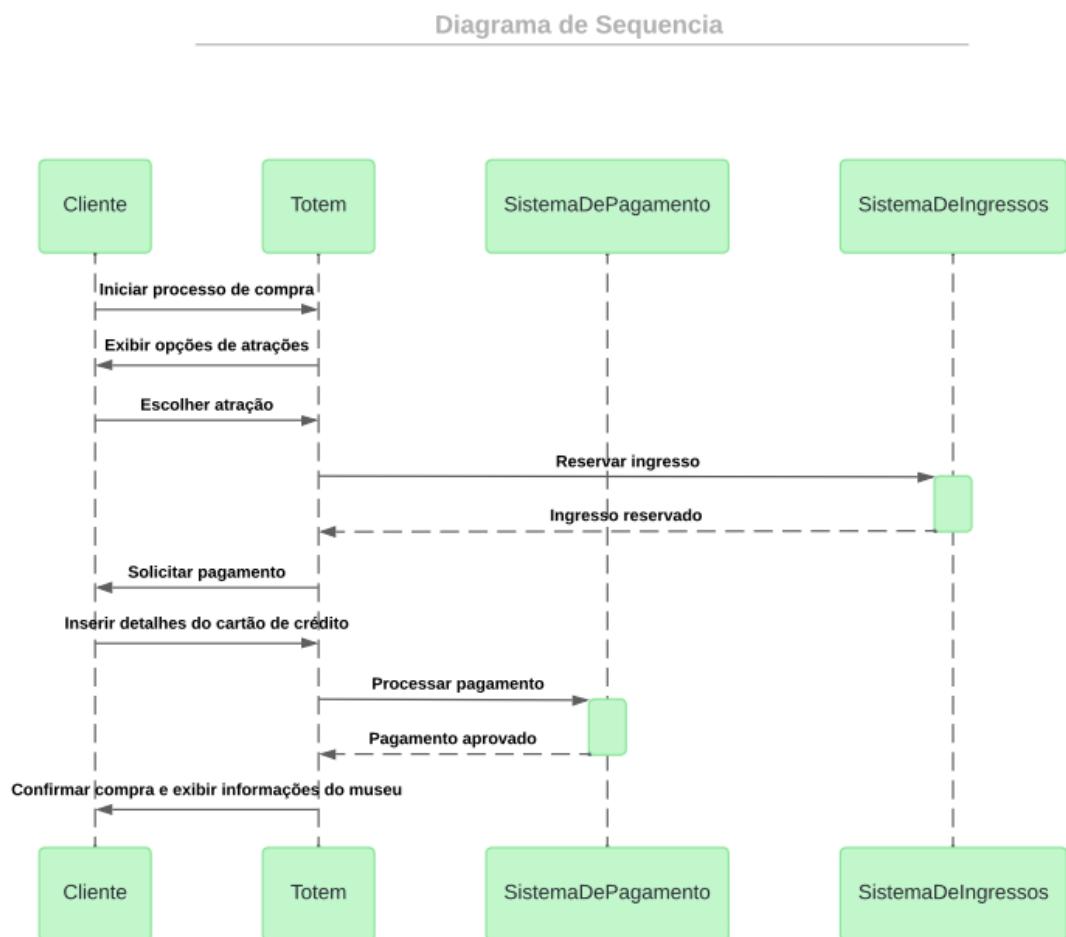
Figura 58: Diagrama – Diagrama de Classe WPF



Fonte: Autoria Própria

APENDICE D – DIAGRAMA DE SEQUÊNCIA

Figura 59: Diagrama - Diagrama de Sequência



APÊNDICE E – CODIGO API – MASTERCONTROLLER – C#

```

using ExplorandoMarteComTecnologia_API.Data;
using ExplorandoMarteComTecnologia_API.DTO;
using ExplorandoMarteComTecnologia_API.Models;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using System.Security.Cryptography.Xml;

namespace ExplorandoMarteComTecnologia_API.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class MasterController : ControllerBase
    {

        private readonly ExplorarMarteDbContext _dbcontext;

        public MasterController(ExplorarMarteDbContext dbcontext)
        {
            _dbcontext = dbcontext;
        }

        [HttpPost("novavenda")]
        public async Task<ActionResult<NomeKeyDTO>> NovaVenda([FromBody]
        NovaVendaDTO novavendaDTO)
        {
            using( var transaction = _dbcontext.Database.BeginTransaction())
            {
                //Verificar se os dados foram enviados corretamente
                Validacao validacao = new Validacao();
                if (!validacao.VerificarCampos(novavendaDTO))
                {
                    await transaction.RollbackAsync();
                    return BadRequest("Os dados enviados estão incorretos.");
                }

                //Separa os modelos do DTO em variaveis separadas
                List<IngressoDTO> ingressoDTO = novavendaDTO.ingressoDTO;
                VendaModel venda = novavendaDTO.venda;

                //Deixa o email com caracteres minusculos
                venda.Email = validacao.ConverterParaMinusculas(venda.Email);

                //Atribui chaves para todos os ingressos
                KeyGenerator keyGenerator = new KeyGenerator();
                List<IngressoDTO> ingressoComKey;
                List<NomeKeyDTO> nomeKeyVisual = new List<NomeKeyDTO>();

                try
                {
                    ingressoComKey = keyGenerator.AtribuirKey(ingressoDTO,
                    venda.Email);
                    foreach(var ingresso in ingressoComKey)
                    {
                        nomeKeyVisual.Add(new NomeKeyDTO { Nome = ingresso.Nome,
                    Key = ingresso.Key });
                    }
                }
                catch (Exception ex)
                {
                    await transaction.RollbackAsync();
                }
            }
        }
    }
}

```

```

        return StatusCode(StatusCodes.Status500InternalServerError,
$"Erro ao atribuir chave aos ingressos: {ex.Message}");
    }

    //Instancia dos Controllers para criar a venda e os ingressos

    VendaController vendaController = new
VendaController(_dbcontext);
    IngressosVendasController ingressoVendasController = new
IngressosVendasController(_dbcontext);
    IngressoController ingressoController = new
IngressoController(_dbcontext);
    RelatorioIngressosController relatorioIngressosController = new
RelatorioIngressosController(_dbcontext);
    RelatorioVendasController relatorioVendasController = new
RelatorioVendasController(_dbcontext);

    //Calcula o preço de cada ingresso e cria a venda
try
{
    var precoIngressos = validacao.CalcularPreco(ingressoComKey);
    venda.Preco = precoIngressos.preco;
    var vendaId = await vendaController.CriarVenda(venda);

    //Verificação do metodo de pagamento para relatorioVendas
    int quantidadeMetodoCredito = 0;
    int quantidadeMetodoDebito = 0;
    int quantidadeMetodoPix = 0;
    switch (venda.MetodoPagamento)
    {
        case "Credito":
            quantidadeMetodoCredito = 1;
            break;

        case "Debito":
            quantidadeMetodoDebito = 1;
            break;

        case "Pix":
            quantidadeMetodoPix = 1;
            break;
    }

    //Criação do relatorioVendas
    var relatorioVendas = new RelatorioVendasDTO
    {
        RelatorioData =
DateOnly.FromDateTime(DateTime.Now).ToString(),
        DinheiroTotal = precoIngressos.preco,
        QuantidadePagamentoCredito = quantidadeMetodoCredito,
        QuantidadePagamentoDebito = quantidadeMetodoDebito,
        QuantidadePagamentoPix = quantidadeMetodoPix
    };

    await
relatorioVendasController.AtualizarRelatorio(relatorioVendas);

    //Criação do relatorioIngressos
    var relatorioIngressos = new RelatorioIngressosDTO
    {
        RelatorioData =
DateOnly.FromDateTime(DateTime.Now).ToString(),

```

```

        TotalIngressosVendidos =
precoIngressos.TotalIngressosInteiro + precoIngressos.TotalIngressosMeia +
precoIngressos.TotalIngressosIsento,
        TotalIngressosInteiro =
precoIngressos.TotalIngressosInteiro,
        TotalIngressosMeia = precoIngressos.TotalIngressosMeia,
        TotalIngressosIsentos =
precoIngressos.TotalIngressosIsento
    };
    await
relatorioIngressosController.AtualizarRelatorio(relatorioIngressos);

        //Associa os ingressos com o Id da venda
foreach (var ingressoAssociar in ingressoComKey)
{
    if (!string.IsNullOrEmpty(ingressoAssociar.Key) &&
ingressoAssociar.Key.Length == 9)
    {
        await
ingressoVendasController.CriarAssociacaoIdKey(vendaId.Value,
ingressoAssociar.Key);
    }
    else
    {
        return BadRequest("Um ou mais ingressos têm a Key
inválida.");
    }
}

        //Cria os ingressos e salva no banco de dados
await ingressoController.CriarIngresso(ingressoComKey);
}
catch (Exception ex)
{
    await transaction.RollbackAsync();
    return StatusCode(StatusCodes.Status500InternalServerError,
$"Erro ao processar a venda: {ex.Message}");
}

try
{
    await _dbcontext.SaveChangesAsync();
    await transaction.CommitAsync();
    return Ok(nomeKeyVisual);
}
catch (Exception ex)
{
    await transaction.RollbackAsync();
    return StatusCode(StatusCodes.Status500InternalServerError,
$"Erro ao criar a venda: {ex.Message}");
}
}

[HttpGet("login/{emailKey}")]
public async Task<ActionResult<LoginRespostaDTO>> RealizarLogin(string
emailKey)
{
    //Chamadas das instancias
    IngressosVendasController ingressosVendasController = new
IngressosVendasController(_dbcontext);
}

```

```

VendaController vendaController = new VendaController(_dbcontext);
Validacao validacao = new Validacao();

if (string.IsNullOrEmpty(emailKey) || emailKey.Length <= 9)
{
    return BadRequest(CriarRespostaErro());
}

EmailKeyDTO emailKeyDTO = new EmailKeyDTO
{
    Email = validacao.SepararEmailKey(emailKey).email,
    Key = validacao.SepararEmailKey(emailKey).key
};

//Valida o Email e a Key que o usuario enviou
if (!validacao.ValidarEmailKey(emailKeyDTO))
{
    return BadRequest(CriarRespostaErro());
}
string email = validacao.ConverterParaMinusculas(emailKeyDTO.Email);
string key = validacao.ConverterParaMaiuscula(emailKeyDTO.Key);

try
{
    //Procura o Id da venda usando a Key que o usuario colocou e
verifica se esta nula
    var idVenda = await ingressosVendasController.PegarIdEmail(key);
    if (string.IsNullOrWhiteSpace(idVenda.ToString()))
    {
        return BadRequest(CriarRespostaErro());
    }

    //Procura a venda usando o Id da venda e armazena em uma variavel
com o modelo VendaModel
    //E valida se existe o Email
    VendaModel venda = await
vendaController.PegarVenda(idVenda.Value);

    if (venda == null || venda.Email == null)
    {
        return BadRequest(CriarRespostaErro());
    }

    if (string.IsNullOrWhiteSpace(venda.Email))
    {
        return BadRequest(CriarRespostaErro());
    }

    //Compara se o email que o usuario escreveu é igual ao da venda
    if (venda.Email != email)
    {
        return BadRequest(CriarRespostaErro());
    }

    //Cria um LoginResposta com o Success true
    LoginRespostaDTO loginResposta = new LoginRespostaDTO
    {
        Success = true,
        Email = venda.Email
    };
}

```

```

        return Ok(loginResposta);
    }
    catch (Exception ex)
    {
        return BadRequest(CriarRespostaErro($"Erro ao realizar login:
{ex.Message}")));
    }
}

[HttpGet("vendas")]
public async Task<ActionResult<IEnumerable<VendaModel>>> PegarVendas()
{
    VendaController vendaController = new VendaController(_dbcontext);
    return await vendaController.PegarVendas();
}

[HttpGet("vendas/{id}")]
public async Task<ActionResult<VendaModel>> PegarVenda(int id)
{
    VendaController vendaController = new VendaController(_dbcontext);
    return await vendaController.PegarVenda(id);
}

[HttpGet("ingressos")]
public async Task<ActionResult<IEnumerable<IngressoModel>>>
PegarIngressos()
{
    IngressoController ingressoController = new
IngressoController(_dbcontext);
    return await ingressoController.PegarIngressos();
}

[HttpGet("ingressos/{key}")]
public async Task<ActionResult<IngressoModel>> PegarIngresso(string key)
{
    if (key.Length != 9)
    {
        return BadRequest("A Key do ingresso deve ter exatamente 9
caracteres e não pode ser nula!");
    }

    IngressoController ingressoController = new
IngressoController(_dbcontext);
    return await ingressoController.PegarIngresso(key);
}

[HttpGet("relatorioIngressos")]
public async Task<ActionResult<IEnumerable<RelatorioIngressosModel>>>
PegarRelatoriosIngressos()
{
    RelatorioIngressosController relatorioIngressosController = new
RelatorioIngressosController(_dbcontext);
    return await relatorioIngressosController.PegarRelatorios();
}

[HttpGet("relatorioVendas")]
public async Task<ActionResult<IEnumerable<RelatorioVendasModel>>>
PegarRelatoriosVenda()
{
    RelatorioVendasController relatorioVendaController = new
RelatorioVendasController(_dbcontext);
}

```

```

        return await relatorioVendaController.PegarRelatorios();
    }

[HttpGet("relatorioIngressos/{anoMesDia}")]
public async Task<ActionResult<RelatorioIngressosModel>>
PegarRelatorioIngressos(string anoMesDia)
{
    RelatorioIngressosController relatorioIngressosController = new
RelatorioIngressosController(_dbcontext);
    return await relatorioIngressosController.PegarRelatorio(anoMesDia);
}

[HttpGet("relatorioVendas/{anoMesDia}")]
public async Task<ActionResult<RelatorioVendasModel>>
PegarRelatorioVenda(string anoMesDia)
{
    RelatorioVendasController relatorioVendaController = new
RelatorioVendasController(_dbcontext);
    return await relatorioVendaController.PegarRelatorio(anoMesDia);
}

[HttpPost("QuestionarioAvaliacao")]
public async Task<ActionResult>
ArmazenarRespostas(QuestionarioAvaliacaoRespostasDTO
questionarioAvaliacaoRespostasDto)
{
    List<QuestionarioRespostasModel> questionarioRespostas =
questionarioAvaliacaoRespostasDto.questionarioRespostas;
    List<AvaliacaoRespostasModel> avaliacaoRespostas =
questionarioAvaliacaoRespostasDto.avaliacaoRespostas;

    QuestionarioRespostasController questionarioRespostasController = new
QuestionarioRespostasController(_dbcontext);
    AvaliacaoRespostasController avaliacaoRespostasController = new
AvaliacaoRespostasController(_dbcontext);
    AvaliacaoSugestaoController avaliacaoSugestaoController = new
AvaliacaoSugestaoController(_dbcontext);
    using(var transaction = _dbcontext.Database.BeginTransaction())
    {

        try
        {

            if (questionarioAvaliacaoRespostasDto.avaliacaoSugestao !=

null)
            {
                await
avaliacaoSugestaoController.ArmazenarSugestao(questionarioAvaliacaoRespostasDto.a
valiacaoSugestao);
            }
        }

        foreach (var respostas in questionarioRespostas)
        {
            await
questionarioRespostasController.AtualizarRespostas(respostas);
        }

        foreach (var respostas in avaliacaoRespostas)
        {
            await
avaliacaoRespostasController.AtualizarRespostas(respostas);
        }
    }
}

```

```

        await _dbcontext.SaveChangesAsync();
        await transaction.CommitAsync();
        return Ok(new { message = "Respostas armazenadas com
sucesso!" });
    }
    catch (Exception ex)
    {
        await transaction.RollbackAsync();
        return StatusCode(StatusCodes.Status500InternalServerError,
$"Erro ao salvar respostas: {ex.Message}");
    }
}

[HttpGet("QuestionarioAvaliacao/q")]
public async Task<ActionResult<IEnumerable<QuestionarioRespostasModel>>>
PegarRespostasPerguntasQuestionario()
{
    QuestionarioRespostasController questionarioRespostasController = new
QuestionarioRespostasController(_dbcontext);
    return await questionarioRespostasController.PegarTodasPerguntas();
}

[HttpGet("QuestionarioAvaliacao/q/{id}")]
public async Task<ActionResult<QuestionarioRespostasModel>>
PegarRespostaPerguntaQuestionario(int id)
{
    QuestionarioRespostasController questionarioRespostasController = new
QuestionarioRespostasController(_dbcontext);
    return await questionarioRespostasController.PegarPergunta(id);
}

[HttpGet("QuestionarioAvaliacao/a")]
public async Task<ActionResult<IEnumerable<AvaliacaoRespostasModel>>>
PegarRespostasPerguntasAvaliacao()
{
    AvaliacaoRespostasController avaliacaoRespostasController = new
AvaliacaoRespostasController(_dbcontext);
    return await avaliacaoRespostasController.PegarTodasPerguntas();
}

[HttpGet("QuestionarioAvaliacao/a/{id}")]
public async Task<ActionResult<AvaliacaoRespostasModel>>
PegarRespostaPerguntaAvaliacao(int id)
{
    AvaliacaoRespostasController avaliacaoRespostasController = new
AvaliacaoRespostasController(_dbcontext);
    return await avaliacaoRespostasController.PegarPergunta(id);
}

[HttpGet("QuestionarioAvaliacao/s")]
public async Task<ActionResult<IEnumerable<AvaliacaoSugestaoModel>>>
PegarSugestoes()
{
    AvaliacaoSugestaoController avaliacaoSugestaoController = new
AvaliacaoSugestaoController(_dbcontext);
    return await avaliacaoSugestaoController.PegarTodasSugestoes();
}

private LoginRespostaDTO CriarRespostaErro()
{
    return new LoginRespostaDTO

```

```
        {
            Success = false,
            ErrorMessage = "Email ou Key inválidos!"
        };
    }

private LoginRespostaDTO CriarRespostaErro(string erro)
{
    return new LoginRespostaDTO
    {
        Success = false,
        ErrorMessage = erro
    };
}

}
```

APÊNDICE F – CODIGO API – INGRESSOCONTROLLER – C#

```

using ExplorandoMarteComTecnologia_API.Data;
using ExplorandoMarteComTecnologia_API.DTO;
using ExplorandoMarteComTecnologia_API.Models;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using System.Net.Sockets;

namespace ExplorandoMarteComTecnologia_API.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class IngressoController : ControllerBase
    {

        private readonly ExplorarMarteDBContext _dbcontext;

        public IngressoController(ExplorarMarteDBContext dbcontext)
        {
            _dbcontext = dbcontext;
        }

        [HttpGet]
        public async Task<ActionResult<IEnumerable<IngressoModel>>>
PegarIngressos()
        {
            var ingressos = await _dbcontext.Ingresso.ToListAsync();
            return Ok(ingressos);
        }

        [HttpGet("{key}")]
        public async Task<ActionResult<IngressoModel>> PegarIngresso(string key)
        {
            var ingresso = await _dbcontext.Ingresso.SingleOrDefaultAsync(i =>
i.Key == key);

            if (ingresso == null)
            {
                return NotFound();
            }

            return Ok(ingresso);
        }

        [HttpPost]
        public async Task<ActionResult<IngressoModel>>
CriarIngresso(List<IngressoDTO> ingressoDTO)
        {
            try
            {
                foreach (var ingressoObjectDTO in ingressoDTO)
                {
                    //Verifica se a Key do ingresso esta nula ou não é de 9
                    caracteres
                    if (string.IsNullOrEmpty(ingressoObjectDTO.Key) ||
                    ingressoObjectDTO.Key.Length != 9)
                    {
                        return BadRequest("A Key do ingresso deve ter exatamente
9 caracteres e não pode ser nula.");
                }
            }
        }
    }
}

```

```
    }

    //Cria o ingresso
    var ingresso = new IngressoModel
    {
        Key = ingressoObjectDTO.Key,
        Nome = ingressoObjectDTO.Nome,
        Tipo = ingressoObjectDTO.Tipo
    };

    _dbcontext.Ingresso.Add(ingresso);
}

await _dbcontext.SaveChangesAsync();

return Ok();
}
catch (Exception ex)
{
    return StatusCode(StatusCodes.Status500InternalServerError,
$"Erro ao criar ingresso: {ex.Message}");
}

}

}
```

APÊNDICE G – CODIGO API – VENDACONTROLLER – C#

```

using ExplorandoMarteComTecnologia_API.Data;
using ExplorandoMarteComTecnologia_API.DTO;
using ExplorandoMarteComTecnologia_API.Models;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;

namespace ExplorandoMarteComTecnologia_API.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class VendaController : ControllerBase
    {

        private readonly ExplorarMarteDBContext _dbcontext;

        public VendaController(ExplorarMarteDBContext dbcontext)
        {
            _dbcontext = dbcontext;
        }

        [HttpGet]
        public async Task<ActionResult<IEnumerable<VendaModel>>> PegarVendas()
        {
            var venda = await _dbcontext.Venda.ToListAsync();
            return Ok(venda);
        }

        [HttpGet("{id}")]
        public async Task<VendaModel> PegarVenda(int id)
        {
            var venda = await _dbcontext.Venda.SingleOrDefaultAsync(v => v.Id == id);

            if (venda == null)
            {
                return null;
            }

            return venda;
        }

        [HttpPost]
        public async Task<ActionResult<int>> CriarVenda(VendaModel venda)
        {
            try
            {
                _dbcontext.Venda.Add(venda);
                await _dbcontext.SaveChangesAsync();

                return venda.Id;
            }
            catch (Exception ex)
            {
                return StatusCode(StatusCodes.Status500InternalServerError,
$"Erro ao criar a venda: {ex.Message}");
            }
        }
    }
}

```

}

APÊNDICE H – CODIGO API – INGRESSOSVENDASCONTROLLER – C#

```

using ExplorandoMarteComTecnologia_API.Data;
using ExplorandoMarteComTecnologia_API.Models;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;

namespace ExplorandoMarteComTecnologia_API.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class IngressosVendasController : ControllerBase
    {

        private readonly ExplorarMarteDbContext _dbcontext;

        public IngressosVendasController(ExplorarMarteDbContext dbcontext)
        {
            _dbcontext = dbcontext;
        }

        [HttpGet("{key_ingresso}")]
        public async Task<ActionResult<int>> PegarIdEmail(string key_ingresso)
        {
            var ingressoVenda = await
            _dbcontext.IngressoVenda.SingleOrDefaultAsync(iv => iv.Key_Ingresso ==
            key_ingresso);

            if (ingressoVenda == null)
            {
                return NotFound();
            }

            return ingressoVenda.Id_Venda;
        }

        [HttpPost]
        public async Task<ActionResult<IngressoVendaModel>>
        CriarAssociacaoIdKey(int id_venda, string key_ingresso)
        {
            var IngressoVenda = new IngressoVendaModel {
                Id_Venda = id_venda,
                Key_Ingresso = key_ingresso
            };

            try
            {
                _dbcontext.IngressoVenda.Add(IngressoVenda);
                await _dbcontext.SaveChangesAsync();
                return Ok(IngressoVenda);
            }
            catch (Exception ex)
            {
                return StatusCode(StatusCodes.Status500InternalServerError,
                    $"Erro ao criar a associação: {ex.Message}");
            }
        }
    }
}

```

}

APÊNDICE I – CODIGO API – RELATORIOINGRESSOSCONTROLLER – C#

```

using ExplorandoMarteComTecnologia_API.Data;
using ExplorandoMarteComTecnologia_API.DTO;
using ExplorandoMarteComTecnologia_API.Models;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using static System.Runtime.InteropServices.JavaScript.JSType;
using System.Security.Cryptography;

namespace ExplorandoMarteComTecnologia_API.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class RelatorioIngressosController : ControllerBase
    {

        private readonly ExplorarMarteDbContext _dbcontext;

        public RelatorioIngressosController(ExplorarMarteDbContext dbContext)
        {
            _dbcontext = dbContext;
        }

        [HttpGet]
        public async Task<ActionResult<IEnumerable<RelatorioIngressosModel>>>
PegarRelatorios()
        {
            var relatorio = await _dbcontext.RelatorioIngressos.ToListAsync();
            return Ok(relatorio);
        }

        [HttpGet("{anoMesDia}")]
        public async Task<ActionResult<RelatorioIngressosModel>>
PegarRelatorio(string anoMesDia)
        {

            //É necessario que anoMesDia seja em string, pois se o metodo for
            receber um DateOnly, o controller vai dar erro
            //Por conta da formatação do DateOnly

            //Converte a string anoMesDia para Int
            Validacao validacao = new Validacao();
            int ano = validacao.SepararConverterAnoMesDia(anoMesDia).ano;
            int mes = validacao.SepararConverterAnoMesDia(anoMesDia).mes;
            int dia = validacao.SepararConverterAnoMesDia(anoMesDia).dia;
            //Cria um DateOnly com as variaveis convertidas
            var data = new DateOnly(ano, mes, dia);

            var relatorio = await
_dbcontext.RelatorioIngressos.FirstOrDefaultAsync(ri => ri.RelatorioData ==
data);

            if (relatorio == null)
            {
                return NotFound("Relatorio não existe!");
            }
            return Ok(relatorio);
        }
    }
}

```

```

    [HttpPost]
    public async Task<ActionResult<RelatorioIngressosModel>>
CriarRelatorioIngressos()
{
    //Pega a data atual e verifica se o relatorio ja existe
    var data = DateOnly.FromDateTime(DateTime.Now);
    if (RelatorioExists(data.Year, data.Month, data.Day))
    {
        return Conflict("Relatorio ja existe!");
    }

    //Faz um novo relatorio com valores zerados
    var relatorio = new RelatorioIngressosModel()
    {
        RelatorioData = data,
        TotalIngressosVendidos = 0,
        TotalIngressosInteiro = 0,
        TotalIngressosMeia = 0,
        TotalIngressosIsentos = 0
    };

    _dbcontext.RelatorioIngressos.Add(relatorio);
    await _dbcontext.SaveChangesAsync();

    return Ok("Relatorio novo criado\nPronto para receber
informações\n\n" + relatorio);
}

[HttpPut("{anoMesDia}")]
public async Task<ActionResult> AtualizarRelatorio(string anoMesDia,
RelatorioIngressosDTO relatorioDTO)
{
    //É necessario que anoMesDia seja em string, pois se o metodo for
    //receber um DateOnly, o controller vai dar erro
    //Por conta da formatação do DateOnly

    //Converte a string anoMesDia para Int
    Validacao validacao = new Validacao();
    int ano = validacao.SepararConverterAnoMesDia(anoMesDia).ano;
    int mes = validacao.SepararConverterAnoMesDia(anoMesDia).mes;
    int dia = validacao.SepararConverterAnoMesDia(anoMesDia).dia;

    //Verifica se a data do relatorioDTO esta nos parametros do DateOnly
    if (!DateOnly.TryParse(relatorioDTO.RelatorioData, out DateOnly
data))
    {
        return BadRequest("Data inválida.");
    }

    //Verifica se a data do relatorioDTO corresponde a data fornecida
    if (data.Year != ano || data.Month != mes || data.Day != dia)
    {
        return BadRequest("Data do relatórioDTO não corresponde à data
fornecida.");
    }

    //Verifica se relatorio existe, caso não, ele cria um
    if (!RelatorioExists(ano, mes, dia))
    {
        await CriarRelatorioIngressos();
    }
}

```

```

        var relatorio = await
_dbcontext.RelatorioIngressos.FirstOrDefaultAsync(ri => ri.RelatorioData ==
data);

        //Faz uma verificação se a variavel esta nula
        if (relatorio == null)
        {
            return NotFound("Relatório não encontrado.");
        }

        //Adiciona os valores do DTO para o relatorio
        relatorio.TotalIngressosVendidos +=
relatorioDTO.TotalIngressosVendidos;
        relatorio.TotalIngressosInteiro +=
relatorioDTO.TotalIngressosInteiro;
        relatorio.TotalIngressosMeia += relatorioDTO.TotalIngressosMeia;
        relatorio.TotalIngressosIsentos +=
relatorioDTO.TotalIngressosIsentos;

        _dbcontext.Entry(relatorio).State = EntityState.Modified;

        try
        {
            await _dbcontext.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!RelatorioExists(ano, mes, dia))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }

        return Ok();
    }

    [HttpPut]
    public async Task<ActionResult> AtualizarRelatorio(RelatorioIngressosDTO
relatorioDTO)
    {

        //Verifica se a data do relatorioDTO esta nos parametros do DateOnly
        if (!DateOnly.TryParse(relatorioDTO.RelatorioData, out DateOnly
data))
        {
            return BadRequest("Data inválida.");
        }

        if (!RelatorioExists(relatorioDTO.RelatorioData))
        {
            await CriarRelatorioIngressos();
        }

        var relatorio = await
_dbcontext.RelatorioIngressos.FirstOrDefaultAsync(ri => ri.RelatorioData ==
data);

        //Faz uma verificação se a variavel esta nula
        if (relatorio == null)

```

```

    {
        return NotFound("Relatório não encontrado.");
    }

    //Adiciona os valores do DTO para o relatorio
    relatorio.TotalIngressosVendidos +=
relatorioDTO.TotalIngressosVendidos;
    relatorio.TotalIngressosInteiro +=
relatorioDTO.TotalIngressosInteiro;
    relatorio.TotalIngressosMeia += relatorioDTO.TotalIngressosMeia;
    relatorio.TotalIngressosIsentos +=
relatorioDTO.TotalIngressosIsentos;

        _dbcontext.Entry(relatorio).State = EntityState.Modified;

    try
    {
        await _dbcontext.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        return BadRequest();
    }

    return Ok();
}

private bool RelatorioExists(int Ano, int Mes, int Dia)
{
    var data = new DateOnly(Ano, Mes, Dia);
    var relatorio = _dbcontext.RelatorioIngressos.SingleOrDefault(ri =>
ri.RelatorioData == data);
    return relatorio != null;
}

private bool RelatorioExists(string relatorioData)
{
    DateOnly.TryParse(relatorioData, out DateOnly data);
    var relatorio = _dbcontext.RelatorioIngressos.SingleOrDefault(ri =>
ri.RelatorioData == data);
    return relatorio != null;
}
}

```

APÊNDICE J – CODIGO API – RELATORIOVENDASCONTROLLER – C#

```

using ExplorandoMarteComTecnologia_API.Data;
using ExplorandoMarteComTecnologia_API.DTO;
using ExplorandoMarteComTecnologia_API.Models;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;

namespace ExplorandoMarteComTecnologia_API.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class RelatorioVendasController : ControllerBase
    {

        private readonly ExplorarMarteDbContext _dbcontext;

        public RelatorioVendasController(ExplorarMarteDbContext dbcontext)
        {
            _dbcontext = dbcontext;
        }

        [HttpGet]
        public async Task<ActionResult<IEnumerable<RelatorioVendasModel>>>
PegarRelatorios()
        {
            var relatorio = await _dbcontext.RelatorioVendas.ToListAsync();
            return Ok(relatorio);
        }

        [HttpGet("{anoMesDia}")]
        public async Task<ActionResult<RelatorioVendasModel>>
PegarRelatorio(string anoMesDia)
        {
            //É necessario que anoMesDia seja em string, pois se o metodo for
            receber um DateOnly, o controller vai dar erro
            //Por conta da formatação do DateOnly

            //Converte a string anoMesDia para Int
            Validacao validacao = new Validacao();
            int ano = validacao.SepararConverterAnoMesDia(anoMesDia).ano;
            int mes = validacao.SepararConverterAnoMesDia(anoMesDia).mes;
            int dia = validacao.SepararConverterAnoMesDia(anoMesDia).dia;
            //Cria um DateOnly com as variaveis convertidas
            var data = new DateOnly(ano, mes, dia);

            var relatorio = await
_dbcontext.RelatorioVendas.FirstOrDefaultAsync(rv => rv.RelatorioData == data);

            if (relatorio == null)
            {
                return NotFound("Relatorio não existe!");
            }
            return Ok(relatorio);
        }

        [HttpPost]
        public async Task<ActionResult<RelatorioVendasModel>>
CriarRelatorioVendas()
        {
            //Pega a data atual e verifica se o relatorio ja existe

```

```

var data = DateOnly.FromDateTime(DateTime.Now);
if (RelatorioExists(data.Year, data.Month, data.Day))
{
    return Conflict("Relatorio ja existe!");
}

//Faz um novo relatorio com valores zerados
var relatorio = new RelatorioVendasModel()
{
    RelatorioData = data,
    DinheiroTotal = 0,
    QuantidadePagamentoCredito = 0,
    QuantidadePagamentoDebito = 0,
    QuantidadePagamentoPix = 0
};

_dbcontext.RelatorioVendas.Add(relatorio);
await _dbcontext.SaveChangesAsync();

return Ok("Relatorio novo criado\nPronto para receber
informações\n" + relatorio);
}

[HttpPut("{anoMesDia}")]
public async Task<ActionResult> AtualizarRelatorio(string anoMesDia,
RelatorioVendasDTO relatorioDTO)
{
    //É necessario que anoMesDia seja em string, pois se o metodo for
    receber um DateOnly, o controller vai dar erro
    //Por conta da formatação do DateOnly

    //Converte a string anoMesDia para Int
    Validacao validacao = new Validacao();
    int ano = validacao.SepararConverterAnoMesDia(anoMesDia).ano;
    int mes = validacao.SepararConverterAnoMesDia(anoMesDia).mes;
    int dia = validacao.SepararConverterAnoMesDia(anoMesDia).dia;

    //Verifica se a data do relatorioDTO esta nos parametros do DateOnly
    if (!DateOnly.TryParse(relatorioDTO.RelatorioData, out DateOnly
data))
    {
        return BadRequest("Data inválida.");
    }

    //Verifica se a data do relatorioDTO corresponde a data fornecida
    if (data.Year != ano || data.Month != mes || data.Day != dia)
    {
        return BadRequest("Data do relatórioDTO não corresponde à data
fornecida.");
    }

    //Verifica se relatorio existe, caso não, ele cria um
    if (!RelatorioExists(ano, mes, dia))
    {
        await CriarRelatorioVendas();
    }

    var relatorio = await
_dbcontext.RelatorioVendas.FirstOrDefaultAsync(rv => rv.RelatorioData == data);

    //Faz uma verificação se a variavel esta nula
    if (relatorio == null)
    {

```

```

        return NotFound("Relatório não encontrado.");
    }

    //Adiciona os valores do DTO para o relatorio
    relatorio.DinheiroTotal += relatorioDTO.DinheiroTotal;
    relatorio.QuantidadePagamentoCredito +=
relatorioDTO.QuantidadePagamentoCredito;
    relatorio.QuantidadePagamentoDebito +=
relatorioDTO.QuantidadePagamentoDebito;
    relatorio.QuantidadePagamentoPix +=
relatorioDTO.QuantidadePagamentoPix;

    _dbcontext.Entry(relatorio).State = EntityState.Modified;

    try
    {
        await _dbcontext.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!RelatorioExists(ano, mes, dia))
        {
            return NotFound();
        }
        else
        {
            throw;
        }
    }

    return Ok();
}

[HttpPut]
public async Task<ActionResult> AtualizarRelatorio(RelatorioVendasDTO
relatorioDTO)
{
    //Verifica se a data do relatorioDTO esta nos parametros do DateOnly
    if (!DateOnly.TryParse(relatorioDTO.RelatorioData, out DateOnly
data))
    {
        return BadRequest("Data inválida.");
    }

    if (!RelatorioExists(relatorioDTO.RelatorioData))
    {
        await CriarRelatorioVendas();
    }

    var relatorio = await
_dbcontext.RelatorioVendas.FirstOrDefaultAsync(rv => rv.RelatorioData == data);

    //Faz uma verificação se a variavel esta nula
    if (relatorio == null)
    {
        return NotFound("Relatório não encontrado.");
    }

    //Adiciona os valores do DTO para o relatorio
    relatorio.DinheiroTotal += relatorioDTO.DinheiroTotal;
    relatorio.QuantidadePagamentoCredito +=
relatorioDTO.QuantidadePagamentoCredito;
}

```

```
        relatorio.QuantidadePagamentoDebito +=
relatorioDTO.QuantidadePagamentoDebito;
        relatorio.QuantidadePagamentoPix +=
relatorioDTO.QuantidadePagamentoPix;

        _dbcontext.Entry(relatorio).State = EntityState.Modified;

        try
{
    await _dbcontext.SaveChangesAsync();
}
catch (DbUpdateConcurrencyException)
{
    return BadRequest();
}

        return Ok();
    }

private bool RelatorioExists(int Ano, int Mes, int Dia)
{
    var data = new DateOnly(Ano, Mes, Dia);
    var relatorio = _dbcontext.RelatorioVendas.SingleOrDefault(rv =>
rv.RelatorioData == data);
    return relatorio != null;
}

private bool RelatorioExists(string relatorioData)
{
    DateOnly.TryParse(relatorioData, out DateOnly data);
    var relatorio = _dbcontext.RelatorioIngressos.SingleOrDefault(ri =>
ri.RelatorioData == data);
    return relatorio != null;
}

}
```

APÊNDICE K – CODIGO API – QUESTIONARIORESPOSTASCONTROLLER – C#

```

using ExplorandoMarteComTecnologia_API.Data;
using ExplorandoMarteComTecnologia_API.DTO;
using ExplorandoMarteComTecnologia_API.Models;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;

namespace ExplorandoMarteComTecnologia_API.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class QuestionarioRespostasController : ControllerBase
    {

        private readonly ExplorarMarteDbContext _dbcontext;

        public QuestionarioRespostasController(ExplorarMarteDbContext dbcontext)
        {
            _dbcontext = dbcontext;
        }

        [HttpGet]
        public async Task<ActionResult<IEnumerable<QuestionarioRespostasModel>>>
PegarTodasPerguntas()
{
    var perguntas = await _dbcontext.QuestionarioRespostas.ToListAsync();
    return Ok(perguntas);
}

        [HttpGet("{id}")]
        public async Task<ActionResult<QuestionarioRespostasModel>>
PegarPergunta(int id)
{
    var pergunta = await
_dbcontext.QuestionarioRespostas.SingleOrDefaultAsync(p => p.Id == id);

    if (pergunta == null)
    {
        return NotFound();
    }

    return Ok(pergunta);
}

        [HttpPost]
        public async Task<ActionResult<QuestionarioRespostasModel>>
CriarPergunta(QuestaoRespostasModel questionarioRespostas)
{
    Validacao validacao = new Validacao();

    //Garantia que na criação de pergunta, os valores iniciam com 0
    questionarioRespostas.Acertos = 0;
    questionarioRespostas.Erros = 0;

    try
    {
        List<QuestionarioRespostasModel> perguntas = await
_dbcontext.QuestionarioRespostas.ToListAsync();
        bool existe = validacao.PerguntaExiste(perguntas,
questionarioRespostas.Pergunta);
    }
}
}

```

```

        if (existe == true)
        {
            return BadRequest("A pergunta já existe no banco de dados.");
        }

        _dbcontext.QuestionarioRespostas.Add(questionarioRespostas);
        await _dbcontext.SaveChangesAsync();
        return Ok();
    }
    catch (DbUpdateException dbEx)
    {
        return StatusCode(StatusCodes.Status500InternalServerError,
"$Erro no banco de dados: {dbEx.Message}");
    }
    catch (Exception ex)
    {
        return StatusCode(StatusCodes.Status500InternalServerError,
"$Erro ao criar pergunta: {ex.Message}");
    }
}

[HttpPut]
public async Task<ActionResult>
AtualizarRespostas(QuestionarioRespostasModel questionarioRespostas)
{
    await VerificarEAdicionarPerguntas();

    Validacao validacao = new Validacao();

    var pergunta = await
_dbcontext.QuestionarioRespostas.FirstOrDefaultAsync(p => p.Id ==
questionarioRespostas.Id);

    if (pergunta == null)
    {
        return NotFound("Pergunta não encontrada!");
    }

    List<QuestionarioRespostasModel> perguntas = await
_dbcontext.QuestionarioRespostas.ToListAsync();
    bool perguntaIdIgual = validacao.PerguntaIdIgual(perguntas,
questionarioRespostas.Id, questionarioRespostas.Pergunta);

    if (perguntaIdIgual == false)
    {
        return NotFound("O ID não bate com a Pergunta no banco de
dados.");
    }

    pergunta.Acertos += questionarioRespostas.Acertos;
    pergunta.Erros += questionarioRespostas.Erros;

    _dbcontext.Entry(pergunta).State = EntityState.Modified;

    try
    {
        await _dbcontext.SaveChangesAsync();
        return Ok();
    }
    catch (DbUpdateConcurrencyException)
    {
}
}

```

```

        return BadRequest();
    }

}

private async Task VerificarEAdicionarPerguntas()
{
    var perguntasNecessarias = new List<string>
    {
        "VulcaoSistemaSolar",
        "FundadorSpaceX",
        "FenomenoVermelho",
        "VidaPassada",
        "MonteOlimpo"
    };

    var perguntasExistentes = await
_dbcontext.QuestionarioRespostas.ToListAsync();

// Verificar se as perguntas existem e estão na ordem correta
for (int i = 0; i < perguntasNecessarias.Count; i++)
{
    var pergunta = perguntasExistentes.FirstOrDefault(p => p.Pergunta
== perguntasNecessarias[i]);

    // Se a pergunta não existir, cria ela
    if (pergunta == null)
    {
        QuestionarioRespostasModel novaPergunta = new
QuestionarioRespostasModel
        {
            Pergunta = perguntasNecessarias[i],
            Acertos = 0,
            Erros = 0
        };
        _dbcontext.QuestionarioRespostas.Add(novaPergunta);
    }
    // Caso a pergunta exista mas a ordem esteja errada, remova e
insira novamente
    else
    {
        if (pergunta.Pergunta != perguntasNecessarias[i])
        {
            _dbcontext.QuestionarioRespostas.Remove(pergunta);
            QuestionarioRespostasModel novaPergunta = new
QuestionarioRespostasModel
            {
                Pergunta = perguntasNecessarias[i],
                Acertos = 0,
                Erros = 0
            };
            _dbcontext.QuestionarioRespostas.Add(novaPergunta);
        }
    }
}

await _dbcontext.SaveChangesAsync();
}

}
}
```


APÊNDICE L – CODIGO API – AVALIACAORESPOSTASCONTROLLER – C#

```

using ExplorandoMarteComTecnologia_API.Data;
using ExplorandoMarteComTecnologia_API.Models;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;

namespace ExplorandoMarteComTecnologia_API.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class AvaliacaoRespostasController : ControllerBase
    {
        private readonly ExplorarMarteDbContext _dbcontext;

        public AvaliacaoRespostasController(ExplorarMarteDbContext dbcontext)
        {
            _dbcontext = dbcontext;
        }

        [HttpGet]
        public async Task<ActionResult<IEnumerable<AvaliacaoRespostasModel>>>
PegarTodasPerguntas()
        {
            var perguntas = await _dbcontext.AvaliacaoRespostas.ToListAsync();
            return Ok(perguntas);
        }

        [HttpGet("{id}")]
        public async Task<ActionResult<AvaliacaoRespostasModel>>
PegarPergunta(int id)
        {
            var pergunta = await
_dbcontext.AvaliacaoRespostas.SingleOrDefaultAsync(p => p.Id == id);

            if (pergunta == null)
            {
                return NotFound();
            }

            return Ok(pergunta);
        }

        [HttpPost]
        public async Task<ActionResult<AvaliacaoRespostasModel>>
CriarPergunta(AvaliacaoRespostasModel avaliacaoRespostas)
        {

            Validação validação = new Validação();

            //Garantia que na criação de pergunta, os valores iniciam com 0
            avaliacaoRespostas.Excelente = 0;
            avaliacaoRespostas.Bom = 0;
            avaliacaoRespostas.Regular = 0;
            avaliacaoRespostas.Ruim = 0;
            avaliacaoRespostas.Pessimo = 0;

            try
            {
                List<AvaliacaoRespostasModel> perguntas = await
_dbcontext.AvaliacaoRespostas.ToListAsync();

```

```

        bool existe = validacao.PerguntaExiste(perguntas,
avaliacaoRespostas.Pergunta);
        if (existe == true)
        {
            return BadRequest("A pergunta já existe no banco de dados.");
        }

        _dbcontext.AvaliacaoRespostas.Add(avaliacaoRespostas);
        await _dbcontext.SaveChangesAsync();
        return Ok();
    }
    catch (DbUpdateException dbEx)
    {
        return StatusCode(StatusCodes.Status500InternalServerError,
$"Erro no banco de dados: {dbEx.Message}");
    }
    catch (Exception ex)
    {
        return StatusCode(StatusCodes.Status500InternalServerError,
$"Erro ao criar pergunta: {ex.Message}");
    }
}

[HttpPut]
public async Task<ActionResult>
AtualizarRespostas(AvaliacaoRespostasModel avaliacaoRespostas)
{
    await VerificarEAdicionarPerguntas();

    Validacao validacao = new Validacao();

    var pergunta = await
_dbcontext.AvaliacaoRespostas.FirstOrDefaultAsync(p => p.Id ==
avaliacaoRespostas.Id);

    if (pergunta == null)
    {
        return NotFound("Pergunta não encontrada!");
    }

    List<AvaliacaoRespostasModel> perguntas = await
_dbcontext.AvaliacaoRespostas.ToListAsync();
    bool perguntaIdIgual = validacao.PerguntaIdIgual(perguntas,
avaliacaoRespostas.Id, avaliacaoRespostas.Pergunta);

    if (perguntaIdIgual == false)
    {
        return NotFound("O ID não bate com a Pergunta no banco de
dados.");
    }

    pergunta.Excelente += avaliacaoRespostas.Excelente;
    pergunta.Bom += avaliacaoRespostas.Bom;
    pergunta.Regular += avaliacaoRespostas.Regular;
    pergunta.Ruim += avaliacaoRespostas.Ruim;
    pergunta.Pessimo += avaliacaoRespostas.Pessimo;

    _dbcontext.Entry(pergunta).State = EntityState.Modified;

    try
    {
        await _dbcontext.SaveChangesAsync();
    }
}

```

```

        return Ok();
    }
    catch (DbUpdateConcurrencyException)
    {
        return BadRequest();
    }
}

private async Task VerificarEAdicionarPerguntas()
{
    var perguntasNecessarias = new List<string>
    {
        "QualidadeExposicoes",
        "InteracaoTotemSite",
        "InformacoesClaras"
    };

    var perguntasExistentes = await
_dbcontext.AvaliacaoRespostas.ToListAsync();

    // Verificar se as perguntas existem e estão na ordem correta
    for (int i = 0; i < perguntasNecessarias.Count; i++)
    {
        var pergunta = perguntasExistentes.FirstOrDefault(p => p.Pergunta
== perguntasNecessarias[i]);

        // Se a pergunta não existir, cria ela
        if (pergunta == null)
        {
            AvaliacaoRespostasModel novaPergunta = new
AvaliacaoRespostasModel
            {
                Pergunta = perguntasNecessarias[i],
                Excelente = 0,
                Bom = 0,
                Regular = 0,
                Ruim = 0,
                Pessimo = 0
            };
            _dbcontext.AvaliacaoRespostas.Add(novaPergunta);
        }
        // Caso a pergunta exista mas a ordem esteja errada, remova e
insira novamente
        else
        {
            if (pergunta.Pergunta != perguntasNecessarias[i])
            {
                _dbcontext.AvaliacaoRespostas.Remove(pergunta);
                AvaliacaoRespostasModel novaPergunta = new
AvaliacaoRespostasModel
                {
                    Pergunta = perguntasNecessarias[i],
                    Excelente = 0,
                    Bom = 0,
                    Regular = 0,
                    Ruim = 0,
                    Pessimo = 0
                };
                _dbcontext.AvaliacaoRespostas.Add(novaPergunta);
            }
        }
    }
}

```

```
        }

        await _dbcontext.SaveChangesAsync();
    }

}
```

APÊNDICE M – CODIGO API – AVALIACAO SUGESTAO CONTROLLER – C#

```

using ExplorandoMarteComTecnologia_API.Data;
using ExplorandoMarteComTecnologia_API.Models;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;

namespace ExplorandoMarteComTecnologia_API.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class AvaliacaoSugestaoController : ControllerBase
    {
        private readonly ExplorarMarteDbContext _dbcontext;

        public AvaliacaoSugestaoController(ExplorarMarteDbContext dbcontext)
        {
            _dbcontext = dbcontext;
        }

        [HttpPost]
        public async Task<ActionResult> ArmazenarSugestao(AvaliacaoSugestaoModel
avaliacaoSugestaoModel)
        {
            _dbcontext.AvaliacaoSugestao.Add(avaliacaoSugestaoModel);
            await _dbcontext.SaveChangesAsync();
            return Ok();
        }

        [HttpGet]
        public async Task<ActionResult<IEnumerable<AvaliacaoSugestaoModel>>>
PegarTodasSugestoes()
        {
            var sugestoes = await _dbcontext.AvaliacaoSugestao.ToListAsync();
            return Ok(sugestoes);
        }
    }
}

```

APÊNDICE N – CODIGO API – VALIDACAO – C#

```

using ExplorandoMarteComTecnologia_API.DTO;
using ExplorandoMarteComTecnologia_API.Models;
using Microsoft.EntityFrameworkCore;
using System.Text.RegularExpressions;

namespace ExplorandoMarteComTecnologia_API.Controllers
{
    public class Validacao
    {

        public (int ano, int mes, int dia) SepararConverterAnoMesDia(string
anoMesDia)
        {
            string anoString = SepararAnoMesDia(anoMesDia).anoString;
            string mesString = SepararAnoMesDia(anoMesDia).mesString;
            string diaString = SepararAnoMesDia(anoMesDia).diaString;

            int ano = ConverterINTAnoMesDia(anoString, mesString, diaString).ano;
            int mes = ConverterINTAnoMesDia(anoString, mesString, diaString).mes;
            int dia = ConverterINTAnoMesDia(anoString, mesString, diaString).dia;
            return (ano, mes, dia);
        }

        public (decimal preco, int TotalIngressosInteiro, int TotalIngressosMeia,
int TotalIngressosIsento) CalcularPreco(List<IngressoDTO> ingressosDTO)
        {
            decimal preco = 0;
            int TotalIngressosInteiro = 0;
            int TotalIngressosMeia = 0;
            int TotalIngressosIsento = 0;

            //Pega um ingresso, verifica o tipo escolhido
            //da o preço correspondente ao tipo, e adiciona na variavel Preco
            foreach (var ingressoObjectDTO in ingressosDTO)
            {
                switch (ingressoObjectDTO.Tipo)
                {
                    case "Inteiro":
                        preco += 20;
                        TotalIngressosInteiro += 1;
                        break;

                    case "Meia":
                        preco += 10;
                        TotalIngressosMeia += 1;
                        break;

                    case "Isento":
                        preco += 0;
                        TotalIngressosIsento += 1;
                        break;

                    default:
                        preco += 20;
                        TotalIngressosInteiro += 1;
                        break;
                }
            }
            //Retorna o preço cheio
        }
    }
}

```

```

        return (preco, TotalIngressosInteiro, TotalIngressosMeia,
TotalIngressosIsento);
    }

    public bool VerificarCampos(NovaVendaDTO novavendaDTO)
{
    List<IngressoDTO> ingressosValidar = novavendaDTO.ingressoDTO;
    VendaModel vendaValidar = novavendaDTO.venda;
    int totalIngressos = 0;

    foreach( var ingresso in ingressosValidar)
    {
        if (string.IsNullOrWhiteSpace(ingresso.Nome) ||
string.IsNullOrWhiteSpace(ingresso.Tipo))
        {
            return false;
        }

        if (ingresso.Nome.Length < 3 || ingresso.Nome.Length > 100)
        {
            return false;
        }

        if (ingresso.Tipo != "Inteiro" && ingresso.Tipo != "Meia" &&
ingresso.Tipo != "Isento")
        {
            return false;
        }

        totalIngressos += 1;
    }

    if (string.IsNullOrWhiteSpace(vendaValidar.Email) ||
vendaValidar.Email.Length < 3)
    {
        return false;
    }

    if (!Regex.IsMatch(vendaValidar.Email,
@"^[\w\.-]+@[^\w\.-]+\.\w{2,}$"))
    {
        return false;
    }

    if (vendaValidar.Quantidade != totalIngressos)
    {
        return false;
    }

    if (vendaValidar.MetodoPagamento != "Credito" &&
vendaValidar.MetodoPagamento != "Debito" && vendaValidar.MetodoPagamento !=
"Pix")
    {
        return false;
    }

    return true;
}

public bool ValidarEmailKey(EmailKeyDTO emailKeyDTO)
{

```

```

        if (string.IsNullOrWhiteSpace(emailKeyDTO.Email) ||
string.IsNullOrWhiteSpace(emailKeyDTO.Key))
{
    return false;
}

if (emailKeyDTO.Email.Length < 3 || emailKeyDTO.Key.Length != 9)
{
    return false;
}

if (!Regex.IsMatch(emailKeyDTO.Email, @"^[@\s]+@[^\s]+\.[^\s]+$"))
{
    return false;
}

return true;
}

public (string email, string key) SepararEmailKey(string emailKey)
{
    // Verifica se a string tem pelo menos o tamanho da key
    if (string.IsNullOrWhiteSpace(emailKey) || emailKey.Length <= 9)
    {
        throw new ArgumentException("O valor fornecido para emailKey não
é válido.");
    }

    // Pega os últimos 9 caracteres para a key
    string key = emailKey.Substring(emailKey.Length - 9);

    // Pega o restante da string para o email
    string email = emailKey.Substring(0, emailKey.Length - 9);

    // Converte email para minúsculas e key para maiúsculas, se
necessário
    email = ConverterParaMinusculas(email);
    key = ConverterParaMaiuscula(key);

    return (email, key);
}

public string ConverterParaMinusculas(string input)
{
    return string.IsNullOrWhiteSpace(input) ? input :
input.ToLowerInvariant();
}
public string ConverterParaMaiuscula(string input)
{
    return string.IsNullOrWhiteSpace(input) ? input :
input.ToUpperInvariant();
}

private (string anoString, string mesString, string diaString)
SepararAnoMesDia(string anoMesDia)
{
    string anoString = anoMesDia.Substring(0, 4);
    string mesString = anoMesDia.Substring(4, 2);
    string diaString = anoMesDia.Substring(6, 2);

    return (anoString, mesString, diaString);
}

```

```

    private (int ano, int mes, int dia) ConverterINTAnoMesDia(string
anoString, string mesString, string diaString)
{
    int ano = Convert.ToInt32(anoString);
    int mes = Convert.ToInt32(mesString);
    int dia = Convert.ToInt32(diaString);

    return (ano, mes, dia);
}

public bool PerguntaExiste(List<QuestionarioRespostasModel> perguntas,
string pergunta)
{
    var perguntaAchada = perguntas.SingleOrDefault(p =>
p.Pergunta.ToLower() == pergunta.ToLower());

    if (perguntaAchada == null)
    {
        return false;
    }

    return true;
}

public bool PerguntaExiste(List<AvaliacaoRespostasModel> perguntas,
string pergunta)
{
    var perguntaAchada = perguntas.SingleOrDefault(p =>
p.Pergunta.ToLower() == pergunta.ToLower());

    if (perguntaAchada == null)
    {
        return false;
    }

    return true;
}

public bool PerguntaIdIgual(List<QuestionarioRespostasModel> perguntas,
int id, string pergunta)
{
    var perguntaAchada = perguntas.SingleOrDefault(p => p.Id == id);

    if (perguntaAchada == null)
    {
        return false; // Não encontrou uma pergunta com o ID especificado
    }

    if (!perguntaAchada.Pergunta.ToLower().Equals(pergunta.ToLower()))
    {
        return false;
    }
    return true;
}

public bool PerguntaIdIgual(List<AvaliacaoRespostasModel> perguntas, int
id, string pergunta)
{
    var perguntaAchada = perguntas.SingleOrDefault(p => p.Id == id);

    if (perguntaAchada == null)

```

```
{      return false; // Não encontrou uma pergunta com o ID especificado
}

if (!perguntaAchada.Pergunta.ToLower().Equals(pergunta.ToLower()))
{
    return false;
}
return true;
}

}
```

APÊNDICE O – CODIGO API – KEYGENERATOR – C#

```

using ExplorandoMarteComTecnologia_API.DTO;
using System.Text.RegularExpressions;

namespace ExplorandoMarteComTecnologia_API.Controllers
{
    public class KeyGenerator
    {

        private readonly Random random = new Random();

        public List<IngressoDTO> AtribuirKey(List<IngressoDTO> ingressoDTO,
string email)
        {
            List<IngressoDTO> ingressoComKey = new List<IngressoDTO>();

            foreach (var ingressoObjectDTO in ingressoDTO)
            {
                ingressoObjectDTO.Key = GenerateKey(ingressoObjectDTO.Nome,
email);

                if (!string.IsNullOrEmpty(ingressoObjectDTO.Key) &&
ingressoObjectDTO.Key.Length == 9)
                {
                    ingressoComKey.Add(ingressoObjectDTO);
                }
                else
                {
                    throw new Exception($"Erro ao gerar a Key para
{ingressoObjectDTO.Nome}. A Key deve ter exatamente 9 caracteres e não pode ser
vazia.");
                }
            }

            return ingressoComKey;
        }

        private string GenerateKey(string nome, string email)
        {

            // Remove caracteres não alfanuméricos e converte para maiúsculas
            string nomeParte = Regex.Replace(nome, @"[^a-zA-Z]", "").ToUpper();
            string emailParte = Regex.Replace(email, @"[^a-zA-Z0-9]",
 "").ToUpper();

            // Garantindo que cada parte tenha no mínimo 3 caracteres
            string keyParteNome = nomeParte.Length >= 3 ? nomeParte.Substring(0,
3) : nomeParte.PadRight(3, 'X');
            string keyParteEmail = emailParte.Length >= 3 ?
emailParte.Substring(0, 3) : emailParte.PadRight(3, 'Y');

            // Embaralha e converte as partes do nome e email
            string shuffledNomeParte = ShuffleString(keyParteNome);
            string shuffledEmailParte = ShuffleString(keyParteEmail);

            //Converte alguns caracteres para Alfanumerico
            string convertedNomeParte = ConvertToAlphanumeric(shuffledNomeParte);
            string convertedEmailParte =
ConvertToAlphanumeric(shuffledEmailParte);

            // Gera uma parte numérica aleatória de 3 dígitos

```

```

        string randomPart = random.Next(100, 999).ToString();

        // Gera o Key final
        string keyFinal = ShuffleKey(convertedNomeParte, convertedEmailParte,
randomPart);

        if (keyFinal.Length != 9 || string.IsNullOrEmpty(keyFinal))
        {
            return GenerateKey(nome, email);
        }

        return keyFinal;
    }

    private string ShuffleKey(string keyParteNome, string keyParteEmail,
string randomPart)
{
    int keySelecao = random.Next(1, 7); // Gera um número aleatório entre
1 e 6

    return keySelecao switch
    {
        1 => keyParteNome + keyParteEmail + randomPart,
        2 => keyParteNome + randomPart + keyParteEmail,
        3 => keyParteEmail + randomPart + keyParteNome,
        4 => randomPart + keyParteEmail + keyParteNome,
        5 => randomPart + keyParteNome + keyParteEmail,
        6 => keyParteEmail + keyParteNome + randomPart,
        _ => keyParteEmail + randomPart + keyParteNome // Default, mas
não deve ser chamado
    };
}

private string ShuffleString(string input)
{
    var chars = input.ToCharArray();
    for (int i = chars.Length - 1; i > 0; i--)
    {
        int swapIndex = random.Next(i + 1);
        (chars[i], chars[swapIndex]) = (chars[swapIndex], chars[i]);
    }
    return new string(chars);
}

private string ConvertToAlphanumeric(string input)
{
    // Mapeamento simples de substituições de letras por números
    var substitutionMap = new Dictionary<char, char>
    {
        { 'A', '4' }, { 'E', '3' }, { 'I', '1' }, { 'O', '0' },
        { 'S', '5' }, { 'T', '7' }, { 'B', '8' }, { 'G', '6' },
        { 'Z', '2' }, { 'L', '1' }
    };

    // Substitui letras de acordo com o mapeamento de forma randômica
    return new string(input.Select(c =>
        random.Next(2) == 0 && substitutionMap.ContainsKey(c) ?
substitutionMap[c] : c
    ).ToArray());
}
}

```

}

APÊNDICE P – CODIGO API – EXPLORAR MARTE DB CONTEXT – C#

```
using ExplorandoMarteComTecnologia_API.Models;
using Microsoft.EntityFrameworkCore;

namespace ExplorandoMarteComTecnologia_API.Data
{
    public class ExplorarMarteDBContext : DbContext
    {

        public ExplorarMarteDBContext(DbContextOptions<ExplorarMarteDBContext>
options) : base(options) { }

        public DbSet<IngressoModel> Ingresso { get; set; }
        public DbSet<PasseModel> Passe { get; set; }
        public DbSet<VendaModel> Venda { get; set; }

        public DbSet<IngressoPasseModel> IngressoPasse { get; set; }
        public DbSet<IngressoVendaModel> IngressoVenda { get; set; }

        public DbSet<RelatorioIngressosModel> RelatorioIngressos { get; set; }
        public DbSet<RelatorioVendasModel> RelatorioVendas { get; set; }

        public DbSet<QuestionarioRespostasModel> QuestionarioRespostas { get;
set; }
        public DbSet<AvaliacaoRespostasModel> AvaliacaoRespostas { get; set; }
        public DbSet<AvaliacaoSugestaoModel> AvaliacaoSugestao { get; set; }

    }
}
```

APÊNDICE Q – CODIGO API – INGRESSOMODEL – C#

```
using System.ComponentModel.DataAnnotations;

namespace ExplorandoMarteComTecnologia_API.Models
{
    public class IngressoModel
    {
        public int Id { get; set; }

        [MaxLength(9)]
        public string Key { get; set; }

        [MaxLength(100)]
        public string Nome { get; set; }

        [MaxLength(30)]
        public string Tipo { get; set; }

    }
}
```

APÊNDICE R – CODIGO API – VENDAMODEL – C#

```
using System.ComponentModel.DataAnnotations;

namespace ExplorandoMarteComTecnologia_API.Models
{
    public class VendaModel
    {
        public int Id { get; set; }
        public string Email { get; set; }
        public int Quantidade { get; set; }

        [MaxLength(30)]
        public string MetodoPagamento { get; set; }
        public decimal Preco { get; set; }
    }
}
```

APÊNDICE S – CODIGO API – INGRESSOVENDAMODEL – C#

```
using System.ComponentModel.DataAnnotations;

namespace ExplorandoMarteComTecnologia_API.Models
{
    public class IngressoVendaModel
    {
        public int Id { get; set; }
        public int Id_Venda { get; set; }

        [MaxLength(9)]
        public string Key_Ingresso { get; set; }

    }
}
```

APÊNDICE T – CODIGO API – QUESTIONARIORESPOSTASMODEL – C#

```
namespace ExplorandoMarteComTecnologia_API.Models
{
    public class QuestionarioRespostasModel
    {
        public int Id { get; set; }
        public required string Pergunta { get; set; }
        public int Acertos { get; set; }
        public int Erros { get; set; }

    }
}
```

APÊNDICE U – CODIGO API – AVALIACAORESPOSTASMODEL – C#

```
namespace ExplorandoMarteComTecnologia_API.Models
{
    public class AvaliacaoRespostasModel
    {
        public int Id { get; set; }
        public required string Pergunta { get; set; }
        public int Excelente { get; set; }
        public int Bom { get; set; }
        public int Regular { get; set; }
        public int Ruim { get; set; }
        public int Pessimo { get; set; }
    }
}
```

APÊNDICE V – CODIGO API – AVALIACAOSUGESTAOMODEL – C#

```
namespace ExplorandoMarteComTecnologia_API.Models
{
    public class AvaliacaoSugestaoModel
    {
        public int Id { get; set; }
        public required string Sugestao { get; set; }
    }
}
```

APÊNDICE W – CODIGO API – RELATORIOINGRESSOSMODEL – C#

```
namespace ExplorandoMarteComTecnologia_API.Models
{
    public class RelatorioIngressosModel
    {

        public int Id { get; set; }
        public DateOnly RelatorioData { get; set; }
        public int TotalIngressosVendidos { get; set; }
        public int TotalIngressosInteiro { get; set; }
        public int TotalIngressosMeia { get; set; }
        public int TotalIngressosIsentos { get; set; }

    }
}
```

APÊNDICE X – CODIGO API – RELATORIOVENDASMODEL – C#

```
namespace ExplorandoMarteComTecnologia_API.Models
{
    public class RelatorioVendasModel
    {
        public int Id { get; set; }
        public DateOnly RelatorioData { get; set; }
        public decimal DinheiroTotal { get; set; }
        public int QuantidadePagamentoCredito { get; set; }
        public int QuantidadePagamentoDebito { get; set; }
        public int QuantidadePagamentoPix { get; set; }

    }
}
```

APÊNDICE Y – CODIGO API – AVALIACAORESPOSTASDTO – C#

```
namespace ExplorandoMarteComTecnologia_API.DTO
{
    public class AvaliacaoRespostasDTO
    {
        public required string Pergunta { get; set; }
        public int? Excelente { get; set; }
        public int? Bom { get; set; }
        public int? Regular { get; set; }
        public int? Ruim { get; set; }
        public int? Pessimo { get; set; }
    }
}
```

APÊNDICE Z – CODIGO API – EMAILKEYDTO – C#

```
namespace ExplorandoMarteComTecnologia_API.DTO
{
    public class EmailKeyDTO
    {
        public string Email { get; set; }
        public string Key { get; set; }
    }
}
```

APÊNDICE AA – CODIGO API – INGRESSODTO – C#

```
using System.ComponentModel.DataAnnotations;

namespace ExplorandoMarteComTecnologia_API.DTO
{
    public class IngressoDTO
    {
        public string? Key { get; set; }
        [MaxLength(100)]
        public string Nome { get; set; }
        public string Tipo { get; set; }
    }
}
```

APÊNDICE AB – CÓDIGO API – LOGINRESPOSTADTO – C#

```
namespace ExplorandoMarteComTecnologia_API.DTO
{
    public class LoginRespostaDTO
    {
        public bool Success { get; set; }
        public string? Email { get; set; }
        public string? ErrorMessage { get; set; }

    }
}
```

APÊNDICE AC – CODIGO API – NOMEKEYDTO – C#

```
namespace ExplorandoMarteComTecnologia_API.DTO
{
    public class NomeKeyDTO
    {
        public string Nome { get; set; }
        public string Key { get; set; }
    }
}
```

APÊNDICE AD – CODIGO API – NOAVENDADTO – C#

```
using ExplorandoMarteComTecnologia_API.Models;

namespace ExplorandoMarteComTecnologia_API.DTO
{
    public class NovaVendaDTO
    {
        public List<IngressoDTO> ingressoDTO { get; set; }
        public VendaModel venda { get; set; }
    }
}
```

APÊNDICE AF – CODIGO API – QUESTIONARIOAVALIACAORESPOSTASDTO – C#

```
using ExplorandoMarteComTecnologia_API.Models;

namespace ExplorandoMarteComTecnologia_API.DTO
{
    public class QuestionarioAvaliacaoRespostasDTO
    {
        public List<QuestionarioRespostasModel> questionarioRespostas { get; set; }
        public List<AvaliacaoRespostasModel> avaliacaoRespostas { get; set; }
        public AvaliacaoSugestaoModel ?avaliacaoSugestao { get; set; }
    }
}
```

APÊNDICE AG – CODIGO API – QUESTIONARIORESPOSTASDTO – C#

```
namespace ExplorandoMarteComTecnologia_API.DTO
{
    public class QuestionarioRespostasDTO
    {
        public required string Pergunta { get; set; }
        public int Acertos { get; set; }
        public int Erros { get; set; }
    }
}
```

APÊNDICE AH – CODIGO API – RELATORIOINGRESSOSDTO – C#

```
namespace ExplorandoMarteComTecnologia_API.DTO
{
    public class RelatorioIngressosDTO
    {
        public string RelatorioData { get; set; }
        public int TotalIngressosVendidos { get; set; }
        public int TotalIngressosInteiro { get; set; }
        public int TotalIngressosMeia { get; set; }
        public int TotalIngressosIsentos { get; set; }
    }
}
```

APÊNDICE AI – CODIGO API – RELATORIOVENDASDTO – C#

```
namespace ExplorandoMarteComTecnologia_API.DTO
{
    public class RelatorioVendasDTO
    {
        public string RelatorioData { get; set; } // String para o formato yyyy-
MM-dd
        public decimal DinheiroTotal { get; set; }
        public int QuantidadePagamentoCredito { get; set; }
        public int QuantidadePagamentoDebito { get; set; }
        public int QuantidadePagamentoPix { get; set; }
    }
}
```

APÊNDICE AJ – CODIGO TOTEM – MAINACTIVITY – XAML

```

<Window x:Class="ExplorandoMarteComTecnologia_WPF.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:ExplorandoMarteComTecnologia_WPF"
    mc:Ignorable="d"
    Title="MainWindow"
    Height="768" Width="1366"
    WindowStyle="None"
    ResizeMode="NoResize"
    WindowStartupLocation="CenterScreen" KeyDown="Window_KeyDown">

    <Grid>
        <Grid.Background>
            <ImageBrush ImageSource="/Imagens/Fundos/Fundo1.png"/>
        </Grid.Background>

        <!-- Botão Mapa -->
        <Button x:Name="btnMapa" HorizontalAlignment="Left"
            VerticalAlignment="Bottom" Margin="60,0,0,13" Width="400" Height="152"
            Click="btnMapa_Click">
            <Button.Template>
                <ControlTemplate TargetType="Button">
                    <Grid>
                        <Image x:Name="imgButton"
                            Source="/Imagens/Botoes/TelaPrincipal/BotaoMapa.png"/>
                    </Grid>
                    <ControlTemplate.Triggers>
                        <Trigger Property="IsMouseOver" Value="True">
                            <Setter TargetName="imgButton" Property="Source"
                                Value="/Imagens/Botoes/TelaPrincipal/BotaoMapaSelecionado.png"/>
                        </Trigger>
                    </ControlTemplate.Triggers>
                </ControlTemplate>
            </Button.Template>
        </Button>

        <!-- Botão Exposições -->
        <Button x:Name="btnExposicoes" Margin="0,0,0,13"
            Click="btnExposicoes_Click" Height="152" VerticalAlignment="Bottom"
            HorizontalAlignment="Center" Width="400">
            <Button.Template>
                <ControlTemplate TargetType="Button">
                    <Grid>
                        <Image x:Name="imgButton"
                            Source="/Imagens/Botoes/TelaPrincipal/BotaoExposicoes.png"/>
                    </Grid>
                    <ControlTemplate.Triggers>
                        <Trigger Property="IsMouseOver" Value="True">
                            <Setter TargetName="imgButton" Property="Source"
                                Value="/Imagens/Botoes/TelaPrincipal/BotaoExposicoesSelecionado.png"/>
                        </Trigger>
                    </ControlTemplate.Triggers>
                </ControlTemplate>
            </Button.Template>
        </Button>

        <!-- Botão Questionário -->
    
```

```
<Button x:Name="btnQuestionario" HorizontalAlignment="Right"
VerticalAlignment="Bottom" Margin="0,0,60,13" Width="400" Height="152"
Click="btnQuestionario_Click">
    <Button.Template>
        <ControlTemplate TargetType="Button">
            <Grid>
                <Image x:Name="imgButton"
Source="/Imagens/Botoes/TelaPrincipal/BotaoQuestionario.png"/>
            </Grid>
            <ControlTemplate.Triggers>
                <Trigger Property="IsMouseOver" Value="True">
                    <Setter TargetName="imgButton" Property="Source"
Value="/Imagens/Botoes/TelaPrincipal/BotaoQuestionarioSelecionado.png"/>
                </Trigger>
            </ControlTemplate.Triggers>
        </ControlTemplate>
    </Button.Template>
</Button>

<!-- Frame para conteúdo dinâmico (ajustado para ser desenhado por
último) --&gt;
&lt;Frame x:Name="MainFrame" NavigationUIVisibility="Hidden"
HorizontalAlignment="Stretch" VerticalAlignment="Stretch" /&gt;
&lt;Image x:Name="imgPainel" Margin="0,0,0,0" OpacityMask="#FF616161"
Source="/Imagens/Fundos/FundoDeElementos1.png" Visibility="Hidden"/&gt;
&lt;Label x:Name="lblMensagemQuestionario" Content="Corrigindo suas
respostas! Aguarde um momento..." Margin="86,0,86,0" VerticalAlignment="Center"
FontSize="48" FontFamily="Arial" FontWeight="Bold" Foreground="White"
Visibility="Hidden"/&gt;

&lt;/Grid&gt;
&lt;/Window&gt;</pre>
```

APÊNDICE AK – CODIGO TOTEM – MAINACTIVITY – C#

```

using ExplorandoMarteComTecnologia_WPF.Controllers;
using ExplorandoMarteComTecnologia_WPF.Views;
using System.Windows;
using System.Windows.Input;
using System.Windows.Navigation;

namespace ExplorandoMarteComTecnologia_WPF
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }

        private async void btnQuestionario_Click(object sender, RoutedEventArgs
e)
        {
            btnExposicoes.IsEnabled = false;
            btnMapa.IsEnabled = false;
            btnQuestionario.IsEnabled = false;

            Estatico.RELATORIOPAGINA = 1;

            Estatico.MENSAGEMCOMUNICACAO = "";
            Estatico.ERROMENSAGEM = "";

            if (String.IsNullOrEmpty(Estatico.LINKAPI))
            {
                MessageBox.Show($"API não configurada\nImpedindo Questionario
para evitar erros\nPedir ajuda de um funcionario!");

                btnExposicoes.IsEnabled = true;
                btnMapa.IsEnabled = true;
                btnQuestionario.IsEnabled = true;
                return;
            }

            Questionario questionario = new Questionario();
            this.Hide();
            questionario.ShowDialog();

            imgPainel.Visibility = Visibility.Visible;
            lblMensagemQuestionario.Visibility = Visibility.Visible;

            this.Show();
            if (Estatico.MENSAGEMCOMUNICACAO == "ARMAZENAR")
            {
                Controle controle = new Controle();
                string json =
controle.MontarJsonQuestionarioAvaliacao(Estatico.questionarioRespostas,
Estatico.avaliacaoRespostas, Estatico.COMENTARIO);
                if (json != "")
                {

```

```

        await
controle.ArmarRespostas("/api/Master/QuestionarioAvaliacao", json);
                //voltando para o MainFrame, abrir uma pagina nova de
respostas
                //Mostrar respostas certas (e a que o usuario errou)
if (Estatico.ERROMENSAGEM == "ERRO")
{
    MessageBox.Show("Erro ao armazenar respostas\nVerificar
conexão com a API\nPedir ajuda de um funcionario!");
    imgPainel.Visibility = Visibility.Hidden;
    lblMensagemQuestionario.Visibility = Visibility.Hidden;
    btnExposicoes.IsEnabled = true;
    btnMapa.IsEnabled = true;
    btnQuestionario.IsEnabled = true;
    return;
}
RelatorioRespostas relatorioRespostas = new
RelatorioRespostas();
MainFrame.NavigationService.Navigate(relatorioRespostas);
}

imgPainel.Visibility = Visibility.Hidden;
lblMensagemQuestionario.Visibility = Visibility.Hidden;

btnExposicoes.IsEnabled = true;
btnMapa.IsEnabled = true;
btnQuestionario.IsEnabled = true;

Estatico.RELATORIOPAGINA = 1;

}

private void btnExposicoes_Click(object sender, RoutedEventArgs e)
{
    Exposicoes exposicoes = new Exposicoes();
    MainFrame.NavigationService.Navigate(exposicoes);
}

private void btnMapa_Click(object sender, RoutedEventArgs e)
{
    Mapa mapa = new Mapa();
    MainFrame.NavigationService.Navigate(mapa);
}

private void MainFrame_Navigated(object sender, NavigationEventArgs e)
{

}

private void Window_KeyDown(object sender, KeyEventArgs e)
{
    if (e.Key == Key.RightShift)
    {
        Configuracoes configuracoes = new Configuracoes();
        MainFrame.NavigationService.Navigate(configuracoes);
    }
}
}

```

APÊNDICE AL – CODIGO TOTEM – CONFIGURACOES – XAML

```

<Page x:Class="ExplorandoMarteComTecnologia_WPF.Views.Configuracoes"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:local="clr-namespace:ExplorandoMarteComTecnologia_WPF.Views"
    mc:Ignorable="d"
    d:DesignHeight="768" d:DesignWidth="1366"
    Title="Configuracoes">

    <Grid Background="#FF121212">

        <Button x:Name="btnMenu" HorizontalAlignment="Left" Height="88"
            Margin="46,38,0,0" VerticalAlignment="Top" Width="247" Click="btnMenu_Click">
            <Button.Template>
                <ControlTemplate TargetType="Button">
                    <Grid>
                        <Image x:Name="imgButton"
                            Source="/Imagens/Botoes/BotaoMenu.png"/>
                    </Grid>
                    <ControlTemplate.Triggers>
                        <Trigger Property="IsMouseOver" Value="True">
                            <Setter TargetName="imgButton" Property="Source"
                                Value="/Imagens/Botoes/BotaoMenuSelecionado.png"/>
                        </Trigger>
                    </ControlTemplate.Triggers>
                </ControlTemplate>
            </Button.Template>
        </Button>

        <Button x:Name="btnFecharSistema" HorizontalAlignment="Left" Height="136"
            Margin="935,14,0,0" VerticalAlignment="Top" Width="391"
            Click="btnFecharSistema_Click" FontSize="24">
            <Button.Template>
                <ControlTemplate TargetType="Button">
                    <Grid>
                        <Image x:Name="imgButton"
                            Source="/Imagens/Botoes/BotaoFechar.png"/>
                    </Grid>
                    <ControlTemplate.Triggers>
                        <Trigger Property="IsMouseOver" Value="True">
                            <Setter TargetName="imgButton" Property="Source"
                                Value="/Imagens/Botoes/BotaoFecharSelecionado.png"/>
                        </Trigger>
                    </ControlTemplate.Triggers>
                </ControlTemplate>
            </Button.Template>
        </Button>

        <Button x:Name="btnSalvarApi" HorizontalAlignment="Left" Height="80"
            Margin="867,197,0,0" VerticalAlignment="Top" Width="228"
            Click="btnSalvarApi_Click" FontSize="24">
            <Button.Template>
                <ControlTemplate TargetType="Button">
                    <Grid>
                        <Image x:Name="imgButton"
                            Source="/Imagens/Botoes/BotaoSalvarAPI.png"/>
                    </Grid>
                    <ControlTemplate.Triggers>

```

```
        <Trigger Property="IsMouseOver" Value="True">
            <Setter TargetName="imgButton" Property="Source"
Value="/Imagens/Botoes/BotaoSalvarAPISelecionado.png"/>
        </Trigger>
    </ControlTemplate.Triggers>
</ControlTemplate>
</Button.Template>
</Button>

<Label x:Name="lblAPI" Content="API:" FontFamily="Arial" FontSize="46"
HorizontalAlignment="Left" Margin="68,206,0,0" VerticalAlignment="Top"
FontWeight="Bold" Foreground="White"/>

<Label x:Name="lblTexto1" Content="(Não colocar / no final do link)"
FontFamily="Arial" FontSize="28" HorizontalAlignment="Left" Margin="170,269,0,0"
VerticalAlignment="Top" FontWeight="Bold" Foreground="White"/>
<Label x:Name="lblTexto2" Content="(Não inserir 'api/Master' no final da
API)" FontFamily="Arial" FontSize="28" HorizontalAlignment="Left"
Margin="170,311,0,0" VerticalAlignment="Top" FontWeight="Bold"
Foreground="White"/>
<TextBox x:Name="txbApiLink" HorizontalAlignment="Left" Height="48"
Margin="175,213,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top"
Width="687" FontSize="43" FontFamily="Arial" FontWeight="Bold"/>

</Grid>

</Page>
```

APÊNDICE AM – CODIGO TOTEM – CONFIGURACOES – C#

```

using ExplorandoMarteComTecnologia_WPF.Controllers;
using System.Windows;
using System.Windows.Controls;

namespace ExplorandoMarteComTecnologia_WPF.Views
{
    /// <summary>
    /// Interação lógica para Configuracoes.xaml
    /// </summary>
    public partial class Configuracoes : Page
    {
        public Configuracoes()
        {
            InitializeComponent();
            txbApiLink.Text = Estatico.LINKAPI;
        }

        private void btnMenu_Click(object sender, RoutedEventArgs e)
        {
            var mainWindow = (MainWindow)Application.Current.MainWindow;

            // Limpar o conteúdo do Frame, removendo a Page
            mainWindow.MainFrame.Content = null;
        }

        private void btnFecharSistema_Click(object sender, RoutedEventArgs e)
        {
            Window janela = Window.GetWindow(this);
            if (janela != null)
            {
                janela.Close();
            }
        }

        private void btnSalvarApi_Click(object sender, RoutedEventArgs e)
        {
            if (!string.IsNullOrEmpty(txbApiLink.Text))
            {
                Estatico.LINKAPI = txbApiLink.Text;
                MessageBox.Show("API Salva");
                return;
            }
        }
    }
}

```

APÊNDICE AN – CODIGO TOTEM – EXPOSICOES – XAML

```

<Page x:Class="ExplorandoMarteComTecnologia_WPF.Views.Exposicoes"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:local="clr-namespace:ExplorandoMarteComTecnologia_WPF.Views"
      Unloaded="OnPageUnloaded"
      mc:Ignorable="d"
      d:DesignHeight="768" d:DesignWidth="1366"
      Title="Exposicoes">

    <Grid>
        <Grid.Background>
            <ImageBrush ImageSource="/Imagens/Fundos/Fundo6.png"/>
        </Grid.Background>

        <Image x:Name="pcbFundoElementos" Margin="50,20,50,20"
               Source="/Imagens/Fundos/FundoDeElementos1.png" Stretch="Fill"/>

        <StackPanel Margin="20">
            <TextBlock Text="Exposições" FontSize="30" FontWeight="Bold"
                       Foreground="White" HorizontalAlignment="Center" Margin="0,20,0,20"/>

            <WrapPanel Margin="40, 80, 0, 0">
                <!-- Obra 1 -->
                <Button Click="Obra_Click" Tag="0" Margin="12">
                    <Image Source="/Imagens/Exposicoes/0_Planeta_Vermelho.jpg"
                           Width="220" Height="190" Stretch="Fill"/>
                </Button>

                <!-- Obra 2 -->
                <Button Click="Obra_Click" Tag="1" Margin="12">
                    <Image
                        Source="/Imagens/Exposicoes/Exploracao_e_Potencial_para_Vida.png" Width="220"
                        Height="190" Stretch="Fill"/>
                </Button>

                <!-- Obra 3 -->
                <Button Click="Obra_Click" Tag="2" Margin="12">
                    <Image Source="/Imagens/Exposicoes/Terreno_Marciano.jpg"
                           Width="220" Height="190" Stretch="Fill"/>
                </Button>

                <!-- Obra 4 -->
                <Button Click="Obra_Click" Tag="3" Margin="12">
                    <Image Source="/Imagens/Exposicoes/Agua_em_Marte.jpg"
                           Width="220" Height="190" Stretch="Fill"/>
                </Button>

                <!-- Obra 5 -->
                <Button Click="Obra_Click" Tag="4" Margin="12">
                    <Image Source="/Imagens/Exposicoes/Valles_Marineris.jpg"
                           Width="220" Height="190" Stretch="Fill"/>
                </Button>

                <!-- Obra 6 -->
                <Button Click="Obra_Click" Tag="5" Margin="142,50,0,12">
                    <Image Source="/Imagens/Exposicoes/0_Monte_Olimpo.jpg"
                           Width="220" Height="190" Stretch="Fill"/>
                </Button>
            </WrapPanel>
        </StackPanel>
    </Grid>

```

```
</Button>

<!-- Obra 7 -->
<Button Click="Obra_Click" Tag="6" Margin="142,50,0,12">
    <Image Source="/Imagens/Exposicoes/Impacto_de_Asteroides.jpg"
Width="220" Height="190" Stretch="Fill"/>
</Button>

<!-- Obra 8 -->
<Button Click="Obra_Click" Tag="7" Margin="142,50,0,12">
    <Image Source="/Imagens/Exposicoes/Colonizacao_de_Marte.jpg"
Width="220" Height="190" Stretch="Fill"/>
</Button>

</WrapPanel>
</StackPanel>

<Button x:Name="btnMenu" HorizontalAlignment="Left" Height="88"
Margin="83,51,0,0" VerticalAlignment="Top" Width="247" Click="btnMenu_Click">
    <Button.Template>
        <ControlTemplate TargetType="{x:Type Button}">
            <Grid>
                <Image x:Name="imgButton"
Source="/Imagens/Botoes/BotaoMenu.png" />
            </Grid>
            <ControlTemplate.Triggers>
                <Trigger Property="IsMouseOver" Value="True">
                    <Setter TargetName="imgButton" Property="Source"
Value="/Imagens/Botoes/BotaoMenuSelecionado.png"/>
                </Trigger>
            </ControlTemplate.Triggers>
        </ControlTemplate>
    </Button.Template>
</Button>

</Grid>
</Page>
```

APÊNDICE AO – CODIGO TOTEM – EXPOSICOES – C#

```

using ExplorandoMarteComTecnologia_WPF.Controllers;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Navigation;

using System.Windows.Threading;

namespace ExplorandoMarteComTecnologia_WPF.Views
{
    /// <summary>
    /// Interação lógica para Exposicoes.xaml
    /// </summary>
    public partial class Exposicoes : Page
    {

        DispatcherTimer timer = new DispatcherTimer();
        public Exposicoes()
        {
            InitializeComponent();

            ConfigurarTimer();
        }

        private void ConfigurarTimer()
        {
            timer.Interval = TimeSpan.FromMinutes(1); // 1 minuto
            timer.Tick += Timer_Tick;
            timer.Start();
        }

        private void Timer_Tick(object sender, EventArgs e)
        {
            // Código para retornar ao menu

            // Parar o timer após o tick
            timer.Stop();
            var mainWindow = (MainWindow)Application.Current.MainWindow;
            mainWindow.MainFrame.Content = null;
        }

        private void ResetarTimer()
        {
            timer.Stop(); // Para o timer
            timer.Start(); // Reinicia o timer, voltando a contar do zero
        }

        private void OnPageUnloaded(object sender, RoutedEventArgs e)
        {
            // Para o timer quando a página é descarregada
            timer.Stop();
        }

        private void Obra_Click(object sender, RoutedEventArgs e)
        {

            ResetarTimer();

            // Obter o indice da obra selecionada
            string obraIndexString = (string)((Button)sender).Tag;
        }
    }
}

```

```
Controle controle = new Controle();
int obraIndex = controle.ConverterStringParaInt(obraIndexString);

// Navegar para a página de detalhes da obra
Exposicao exposicao = new Exposicao(obraIndex);
NavigationService.Navigate(exposicao);
}

private void btnMenu_Click(object sender, RoutedEventArgs e)
{
    // Acessar a janela principal
    var mainWindow = (MainWindow)Application.Current.MainWindow;

    // Limpar o conteúdo do Frame, removendo a Page
    mainWindow.MainFrame.Content = null;
}
}
}
```

APÊNDICE AP – CODIGO TOTEM – EXPOSICAO – XAML

```

<Page x:Class="ExplorandoMarteComTecnologia_WPF.Views.Exposicao"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:local="clr-namespace:ExplorandoMarteComTecnologia_WPF.Views"
      Unloaded="OnPageUnloaded"
      mc:Ignorable="d"
      d:DesignHeight="768" d:DesignWidth="1366"
      Title="Exposicao">

    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="143*"/>
            <RowDefinition/>
        </Grid.RowDefinitions>
        <Grid.Background>
            <ImageBrush ImageSource="/Imagens/Fundos/Fundo6.png"/>
        </Grid.Background>

        <Image x:Name="pcbFundoElementos" Margin="26,20,26,20"
               Source="/Imagens/Fundos/FundoDeElementos1.png" Stretch="Fill"/>

        <Label x:Name="lblObraTitulo" Content="Titulo" FontSize="45"
               FontWeight="Bold" Foreground="White" HorizontalAlignment="Center"
               HorizontalContentAlignment="Center" Margin="0,20,0,602" Width="926"/>
        <Label x:Name="lblDescricao" Content="Descrição" FontSize="28"
               Foreground="White" Margin="665,166,50,37" FontFamily="Arial Black"/>
        <Image x:Name="imgObra" Source="/Imagens/PlaceHolder/Obra.png"
               Margin="50,167,706,69"/>

        <Button x:Name="btnVoltar" HorizontalAlignment="Left" Height="106"
               Margin="50,30,0,0" VerticalAlignment="Top" Width="212" Click="btnVoltar_Click">
            <Button.Template>
                <ControlTemplate TargetType="{x:Type Button}">
                    <Grid>
                        <Image x:Name="imgButton"
                               Source="/Imagens/Botoes/BotaoVoltar.png" />
                    </Grid>
                    <ControlTemplate.Triggers>
                        <Trigger Property="IsMouseOver" Value="True">
                            <Setter TargetName="imgButton" Property="Source"
                                   Value="/Imagens/Botoes/BotaoVoltarSelecionado.png"/>
                        </Trigger>
                    </ControlTemplate.Triggers>
                </ControlTemplate>
            </Button.Template>
        </Button>

    </Grid>
</Page>

```

APÊNDICE AQ – CODIGO TOTEM – EXPOSICAO – C#

```

using System.Windows;
using System.Windows.Controls;
using System.Windows.Media.Imaging;
using System.Windows.Threading;

namespace ExplorandoMarteComTecnologia_WPF.Views
{
    /// <summary>
    /// Interação lógica para Exposicao.xaml
    /// </summary>
    public partial class Exposicao : Page
    {

        DispatcherTimer timer = new DispatcherTimer();
        public Exposicao(int obraIndex)
        {
            InitializeComponent();
            MudarExposicao(obraIndex);

            ConfigurarTimer();
        }

        private void ConfigurarTimer()
        {
            timer.Interval = TimeSpan.FromMinutes(1.30); // 1 minuto
            timer.Tick += Timer_Tick;
            timer.Start();
        }

        private void Timer_Tick(object sender, EventArgs e)
        {
            // Código para retornar ao menu

            // Parar o timer após o tick
            timer.Stop();
            var mainWindow = (MainWindow)Application.Current.MainWindow;
            mainWindow.MainFrame.Content = null;
        }

        private void ResetarTimer()
        {
            timer.Stop(); // Para o timer
            timer.Start(); // Reinicia o timer, voltando a contar do zero
        }

        private void OnPageUnloaded(object sender, RoutedEventArgs e)
        {
            // Para o timer quando a página é descarregada
            timer.Stop();
        }

        private void btnVoltar_Click(object sender, RoutedEventArgs e)
        {
            // Acessar a janela principal
            var mainWindow = (MainWindow)Application.Current.MainWindow;

            // Carregar a página Exposicoes no MainFrame
            mainWindow.MainFrame.Content = new Exposicoes();
        }

        private void MudarExposicao(int obraIndex)
    }
}

```

```

{
    switch (obraIndex)
    {
        case 0:
            lblObraTitulo.Content = "O Planeta Vermelho";
            lblDescricao.FontSize = 24;
            lblDescricao.Content = "Marte é conhecido por sua
característica\ncor vermelha, um traço distintivo que fascina\ncientistas e
astrônomos.\nEssa tonalidade é resultado da presença de\nóxido de ferro em sua
superfície, um produto\nda oxidação intensa do ferro presente na\nterra
marciana.\nEm comparação com a Terra, a oxidação\nem Marte é muito mais
acentuada, conferindo\nao planeta sua aparência única\ne deslumbrante.";
            imgObra.Source = new BitmapImage(new
Uri("/Imagens/Exposicoes/0_Planeta_Vermelho.jpg", UriKind.Relative));
            break;

        case 1:
            lblObraTitulo.Content = "Exploração e Potencial para Vida";
            lblDescricao.FontSize = 24;
            lblDescricao.Content = "A exploração de Marte é
fundamental\npara desvendar os segredos do sistema\nsolar.\nEste planeta
vermelho, com sua superfície\nárida e paisagens deslumbrantes,\nné um alvo
fascinante para cientistas\ne engenheiros.\nAtualmente, robôs pioneiros como
o\nCuriosity Rover, Perseverance Rover,\nInSight Lander e ExoMars Rover\nestão
explorando Marte, coletando dados\nsobre sua geologia, composição do
solo,\nclima, atmosfera e potencial\npara água líquida.";
            imgObra.Source = new BitmapImage(new
Uri("/Imagens/Exposicoes/Exploracao_e_Potencial_para_Vida.png",
UriKind.Relative));
            break;

        case 2:
            lblObraTitulo.Content = "Terreno Marciano";
            lblDescricao.FontSize = 24;
            lblDescricao.Content = "Marte apresenta uma
paisagem\ndiversificada, marcada por vales profundos,\ncrateras gigantescas e
vulcões imponentes.\nO mais notável deles é o Olympus Mons,\no maior vulcão do
sistema solar, com uma\naltura impressionante de 27 km.\nA superfície de Marte é
dividida em\nduas regiões principais:\na planície setentrional, caracterizada
por\nsuas amplas extensões planas,\ne a região montanhosa meridional,\nrepleta de
montanhas e vales.";
            imgObra.Source = new BitmapImage(new
Uri("/Imagens/Exposicoes/Terreno_Marciano.jpg", UriKind.Relative));
            break;

        case 3:
            lblObraTitulo.Content = "Água em Marte";
            lblDescricao.FontSize = 24;
            lblDescricao.Content = "A possibilidade de vida em Marte é um
tema\nintrigante e complexo.\nA água é essencial para a vida como\nconhecemos, e
sua presença é um fator\nkritico para o desenvolvimento de organismos\nvivos.\nA
NASA e outras agências espaciais já\nnencontraram evidências de que Marte
teve\nágua líquida no passado, como rios e lagos\nsecos, deltas de rios e
minerais hidratados.\nEssas descobertas sugerem que o planeta\npode ter tido
condições favoráveis\nà vida no passado.";
            imgObra.Source = new BitmapImage(new
Uri("/Imagens/Exposicoes/Aqua_em_Marte.jpg", UriKind.Relative));
            break;

        case 4:
            lblObraTitulo.Content = "Valles Marineris";
            lblDescricao.FontSize = 24;

```

```

        lblDescricao.Content = "Valles Marineris, o maior canyon
do\nsistema solar, é um verdadeiro marco natural\nem Marte, estendendo-se por
uma\nimpressionante distância de 4.000 km.\nPara colocar essa magnitude em
perspectiva,\né quatro vezes mais longo do que o\nGrand Canyon, um dos mais
icônicos\nmonumentos naturais da Terra.\nSua profundidade e vastidão são
um\ntestemunho da força erosiva do vento e\nda água que, no passado, fluíram em
Marte.";
        imgObra.Source = new BitmapImage(new
Uri("/Imagens/Exposicoes/Valles_Marineris.jpg", UriKind.Relative));
        break;

    case 5:
        lblObraTitulo.Content = "O Monte Olimpo";
        lblDescricao.FontSize = 24;
        lblDescricao.Content = "O Monte Olimpo, localizado em
Marte,\né um verdadeiro gigante, com seus\nimpressionantes 27 km de altura,
superando o\nMonte Everest em cerca de três vezes.\nExplorar esse vulcão seria um
desafio\nemocionante e complexo.\nPara explorar o Monte Olimpo, seria\nnecessário
desenvolver tecnologias\navançadas e estratégias inovadoras.\nEm primeiro lugar,
missões robóticas seriam\nessenciais para mapear a superfície e coletar\ndados
sobre a geologia e composição\ndo vulcão.\nIsso ajudaria a identificar áreas
de\ninteresse e riscos potenciais.";
        imgObra.Source = new BitmapImage(new
Uri("/Imagens/Exposicoes/0_Monte_Olimpo.jpg", UriKind.Relative));
        break;

    case 6:
        lblObraTitulo.Content = "Impacto de Asteroides";
        lblDescricao.FontSize = 24;
        lblDescricao.Content = "Marte, o Planeta Vermelho, é um
testemunho\nvivo da violência cósmica que\nmarca a história do sistema
solar.\nSua superfície é pontuada por inúmeras\n crateras de impacto, resultado da
colisão\ncom asteroides e cometas ao longo de\nmilhões de anos, variando em
tamanho desde\npequenas depressões até gigantescas\ncavidades como a Cratera
Hellas, com mais\nde 2.200 km de diâmetro.\nEsses impactos fornecem
informações\nvaliosas sobre a composição e estrutura\ngeológica de Marte, além de
sugerir que a\nágua pode ter sido trazida por cometas\nou asteroides.";
        imgObra.Source = new BitmapImage(new
Uri("/Imagens/Exposicoes/Impacto_de_Asteroides.jpg", UriKind.Relative));
        break;

    case 7:
        lblObraTitulo.Content = "Colonização de Marte";
        lblDescricao.FontSize = 24;
        lblDescricao.Content = "A colonização de Marte é um
objetivo\nambicioso que tem capturado a imaginação\nde científicos, engenheiros e
visionários\npor décadas.\nA NASA e outras agências espaciais já têm\nplanos
concretos para enviar missões\ntripuladas a Marte na década de 2030, com
a\nSpaceX, fundada por Elon Musk,\ntrabalhando arduamente para
desenvolver\ntecnologias necessárias para uma\ncolonização sustentável.\nCom a
tecnologia avançando rapidamente,\na possibilidade de estabelecer uma\ncomunidade
humana no Planeta Vermelho\nestá se tornando cada vez mais realista.\nNo entanto,
a colonização de Marte\nnão será fácil.";
        imgObra.Source = new BitmapImage(new
Uri("/Imagens/Exposicoes/Colonizacao_de_Marte.jpg", UriKind.Relative));
        break;

    default:
        //Se uma obra não existente for selecionada, voltar para o
menu
        var mainWindow = (MainWindow)Application.Current.MainWindow;
        mainWindow.MainFrame.Content = new Exposicoes();
        break;

```

```
        }  
    }  
}
```

APÊNDICE AR – CODIGO TOTEM – MAPA – XAML

```

<Page x:Class="ExplorandoMarteComTecnologia_WPF.Views.Mapa"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:local="clr-namespace:ExplorandoMarteComTecnologia_WPF.Views"
    Unloaded="OnPageUnloaded"
    mc:Ignorable="d"
    d:DesignHeight="768" d:DesignWidth="1366"
    Title="Mapa">

    <Grid>
        <!-- Fundo do grid -->
        <Grid.Background>
            <ImageBrush ImageSource="/Imagens/Fundos/Fundo1.png"/>
        </Grid.Background>

        <Image x:Name="pcbFundoElementos" Margin="26,20,26,20"
            Source="/Imagens/Fundos/FundoDeElementos1.png" Stretch="Fill"/>

        <!-- Imagem do mapa centralizada e ajustada -->
        <Image x:Name="imgMapa" Source="/Imagens/Mapa.png" Stretch="Uniform"
            Margin="227,131,80,73"/>

        <!-- Botão do menu, com ajuste na margem e tamanho -->
        <Button x:Name="btnMenu" HorizontalAlignment="Left" Height="88"
            Margin="46,38,0,0" VerticalAlignment="Top" Width="247" Click="btnMenu_Click">
            <Button.Template>
                <ControlTemplate TargetType="Button">
                    <Grid>
                        <Image x:Name="imgButton"
                            Source="/Imagens/Botoes/BotaoMenu.png"/>
                    </Grid>
                    <ControlTemplate.Triggers>
                        <Trigger Property="IsMouseOver" Value="True">
                            <Setter TargetName="imgButton" Property="Source"
                                Value="/Imagens/Botoes/BotaoMenuSelecionado.png"/>
                        </Trigger>
                    </ControlTemplate.Triggers>
                </ControlTemplate>
            </Button.Template>
        </Button>

    </Grid>
</Page>

```

APÊNDICE AS – CODIGO TOTEM – MAPA – C#

```

using System.Windows;
using System.Windows.Controls;
using System.Windows.Threading;

namespace ExplorandoMarteComTecnologia_WPF.Views
{
    /// <summary>
    /// Interação lógica para Mapa.xaml
    /// </summary>
    public partial class Mapa : Page
    {

        DispatcherTimer timer = new DispatcherTimer();
        public Mapa()
        {
            InitializeComponent();
            ConfigurarTimer();
        }

        private void ResetarTimer()
        {
            timer.Stop(); // Para o timer
            timer.Start(); // Reinicia o timer, voltando a contar do zero
        }

        private void ConfigurarTimer()
        {
            timer.Interval = TimeSpan.FromMinutes(1); // 1 minuto
            timer.Tick += Timer_Tick;
            timer.Start();
        }

        private void Timer_Tick(object sender, EventArgs e)
        {
            // Código para retornar ao menu

            // Parar o timer após o tick
            timer.Stop();
            var mainWindow = (MainWindow)Application.Current.MainWindow;
            mainWindow.MainFrame.Content = null;
        }

        private void OnPageUnloaded(object sender, RoutedEventArgs e)
        {
            // Para o timer quando a página é descarregada
            timer.Stop();
        }

        private void btnMenu_Click(object sender, RoutedEventArgs e)
        {
            // Acessar a janela principal
            var mainWindow = (MainWindow)Application.Current.MainWindow;

            // Limpar o conteúdo do Frame, removendo a Page
            mainWindow.MainFrame.Content = null;
        }
    }
}

```

}

APÊNDICE AT – CODIGO TOTEM – QUESTIONARIO– XAML

```

<Window x:Class="ExplorandoMarteComTecnologia_WPF.Views.Questionario"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:ExplorandoMarteComTecnologia_WPF.Views"
    mc:Ignorable="d"
    Title="Questionario"
    Height="768" Width="1366"
    WindowStyle="None"
    ResizeMode="NoResize"
    WindowStartupLocation="CenterScreen">
    <Grid>
        <Grid.Background>
            <ImageBrush ImageSource="/Imagens/Fundos/Fundo3.png" Stretch="Fill"
TileMode="None"/>
        </Grid.Background>

        <Image x:Name="pcbFundoElementos" HorizontalAlignment="Left"
Margin="78,70,0,70" Width="982" Source="/Imagens/Fundos/FundoDeElementos1.png"
Stretch="Fill"/>

        <Label x:Name="lblPergunta" Content="lblPergunta"
HorizontalAlignment="Left" Margin="276,96,0,0" VerticalAlignment="Top"
FontFamily="Arial" FontSize="24" Foreground="White" FontWeight="Bold"/>

        <Label x:Name="lblAlternativaA" Content="lblAlternativaA"
HorizontalAlignment="Left" Margin="276,233,0,0" VerticalAlignment="Top"
MouseDown="lblAlternativaA_MouseDown" FontFamily="Arial" FontSize="24"
Foreground="White" FontWeight="Bold"/>
        <Label x:Name="lblAlternativaB" Content="lblAlternativaB"
HorizontalAlignment="Left" Margin="276,338,0,0" VerticalAlignment="Top"
MouseDown="lblAlternativaB_MouseDown" FontFamily="Arial" FontSize="24"
Foreground="White" FontWeight="Bold"/>
        <Label x:Name="lblAlternativaC" Content="lblAlternativaC"
HorizontalAlignment="Left" Margin="276,443,0,0" VerticalAlignment="Top"
MouseDown="lblAlternativaC_MouseDown" FontFamily="Arial" FontSize="24"
Foreground="White" FontWeight="Bold"/>
        <Label x:Name="lblAlternativaD" Content="lblAlternativaD"
HorizontalAlignment="Left" Margin="276,548,0,0" VerticalAlignment="Top"
MouseDown="lblAlternativaD_MouseDown" FontFamily="Arial" FontSize="24"
Foreground="White" FontWeight="Bold" RenderTransformOrigin="0.494,0.542"/>

        <Label x:Name="lblMensagem" Content="lblMensagem"
HorizontalAlignment="Left" Margin="140,0,0,79" FontFamily="Arial" FontSize="24"
Foreground="#FFFF3E3E" FontWeight="Bold" Height="38" VerticalAlignment="Bottom"/>
        <Image x:Name="pcbAlternativaA" HorizontalAlignment="Left" Height="100"
Margin="140,202,0,0" VerticalAlignment="Top" Width="100"
Source="/Imagens/Botoes/Questionario/BotaoA.png" Stretch="Fill"
MouseDown="pcbAlternativaA_MouseDown"/>
        <Image x:Name="pcbAlternativaB" HorizontalAlignment="Left" Height="100"
Margin="140,307,0,0" VerticalAlignment="Top" Width="100"
Source="/Imagens/Botoes/Questionario/BotaoB.png" Stretch="Fill"
MouseDown="pcbAlternativaB_MouseDown"/>

```

```
<Image x:Name="pcbAlternativaC" HorizontalAlignment="Left" Height="100"
Margin="140,412,0,0" VerticalAlignment="Top" Width="100"
Source="/Imagens/Botoes/Questionario/BotaoC.png" Stretch="Fill"
MouseDown="pcbAlternativaC_MouseDown"/>
<Image x:Name="pcbAlternativaD" HorizontalAlignment="Left" Height="100"
Margin="140,517,0,0" VerticalAlignment="Top" Width="100"
Source="/Imagens/Botoes/Questionario/BotaoD.png" Stretch="Fill"
MouseDown="pcbAlternativaD_MouseDown"/>

<Image x:Name="pcbContinuar" HorizontalAlignment="Left" Height="94"
Margin="1065,579,0,0" VerticalAlignment="Top" Width="237"
Source="/Imagens/Botoes/BotaoContinuar.png" MouseDown="pcbContinuar_MouseDown"
Stretch="Fill"/>

<Button x:Name="btnMenu" HorizontalAlignment="Left" Height="50"
Margin="99,90,0,0" VerticalAlignment="Top" Width="141" Click="btnMenu_Click">
    <Button.Template>
        <ControlTemplate TargetType="{x:Type Button}">
            <Grid>
                <Image x:Name="imgButton"
Source="/Imagens/Botoes/BotaoMenu.png" />
            </Grid>
            <ControlTemplate.Triggers>
                <Trigger Property="IsMouseOver" Value="True">
                    <Setter TargetName="imgButton" Property="Source"
Value="/Imagens/Botoes/BotaoMenuSelecionado.png"/>
                </Trigger>
            </ControlTemplate.Triggers>
        </ControlTemplate>
    </Button.Template>
</Button>

<Frame x:Name="QuestionarioFrame" />

</Grid>
</Window>
```

APÊNDICE AU – CODIGO TOTEM – QUESTIONARIO– C#

```

using ExplorandoMarteComTecnologia_WPF.Controllers;
using System.Windows;
using System.Windows.Input;
using System.Windows.Media.Imaging;
using System.Windows.Threading;

namespace ExplorandoMarteComTecnologia_WPF.Views
{
    /// <summary>
    /// Lógica interna para Questionario.xaml
    /// </summary>
    public partial class Questionario : Window
    {

        DispatcherTimer timer = new DispatcherTimer();
        public Questionario()
        {
            InitializeComponent();
            Estatico.questionarioRespostas.Clear();
            Estatico.avaliacaoRespostas.Clear();
            Estatico.COMENTARIO = "";
            ProximaPergunta(1);

            ConfigurarTimer();
        }

        private void ResetarTimer()
        {
            timer.Stop(); // Para o timer
            timer.Start(); // Reinicia o timer, voltando a contar do zero
        }

        private void ConfigurarTimer()
        {
            timer.Interval = TimeSpan.FromMinutes(2); // 2 minutos
            timer.Tick += Timer_Tick;
            timer.Start();
        }

        private void Timer_Tick(object sender, EventArgs e)
        {
            // Código para retornar ao menu

            // Parar o timer após o tick
            timer.Stop();
            this.Close();
        }

        int respostaSelecionada = 0;
        int idPergunta = 1;
        string alternativaEscolhida = "";

        private void AtualizarBotoes(string alternativa)
        {
            string caminhoBotaoA = "/Imagens/Botoes/Questionario/BotaoA.png";
            string caminhoBotaoB = "/Imagens/Botoes/Questionario/BotaoB.png";

```

```

        string caminhoBotaoC = "/Imagens/Botoes/Questionario/BotaoC.png";
        string caminhoBotaoD = "/Imagens/Botoes/Questionario/BotaoD.png";

        pcbAlternativaA.Source = new BitmapImage(new Uri(caminhoBotaoA,
UriKind.Relative));
        pcbAlternativaB.Source = new BitmapImage(new Uri(caminhoBotaoB,
UriKind.Relative));
        pcbAlternativaC.Source = new BitmapImage(new Uri(caminhoBotaoC,
UriKind.Relative));
        pcbAlternativaD.Source = new BitmapImage(new Uri(caminhoBotaoD,
UriKind.Relative));

        switch (alternativa)
        {
            case "A":
                string caminhoBotaoASelecionado =
"/Imagens/Botoes/Questionario/BotaoASelecionado.png";
                pcbAlternativaA.Source = new BitmapImage(new
Uri(caminhoBotaoASelecionado, UriKind.Relative));
                break;
            case "B":
                string caminhoBotaoBSelecionado =
"/Imagens/Botoes/Questionario/BotaoBSelecionado.png";
                pcbAlternativaB.Source = new BitmapImage(new
Uri(caminhoBotaoBSelecionado, UriKind.Relative));
                break;
            case "C":
                string caminhoBotaoCSelecionado =
"/Imagens/Botoes/Questionario/BotaoCSelecionado.png";
                pcbAlternativaC.Source = new BitmapImage(new
Uri(caminhoBotaoCSelecionado, UriKind.Relative));
                break;
            case "D":
                string caminhoBotaoDSelecionado =
"/Imagens/Botoes/Questionario/BotaoDSelecionado.png";
                pcbAlternativaD.Source = new BitmapImage(new
Uri(caminhoBotaoDSelecionado, UriKind.Relative));
                break;
            default:
                break;
        }
    }

    public void ProximaPergunta(int idPergunta)
    {

        switch (idPergunta)
        {
            case 1:

                alternativaEscolhida = "";
                respostaSelecionada = 0;

                AtualizarBotoes("CAIR OPCAO DEFAULT");

                lblPergunta.Content = "1 - Qual é o maior vulcão do sistema
solar?";
                lblAlternativaA.Content = "A) Monte Everest, com 8.848 metros
de altura";
                lblAlternativaB.Content = "B) Mauna Kea, com 10.203 metros de
altura";
        }
    }
}

```

```

metros de altura";
lblAlternativaC.Content = "C) Monte Kilimanjaro, com 5.895
lblAlternativaD.Content = "D) Monte Olimpo, com 27 km de
altura";

lblMensagem.Visibility = Visibility.Hidden;
lblMensagem.Content = "";
break;

case 2:

alternativaEscolhida = "";
respostaSelecionada = 0;

AtualizarBotoes("CAIR OPCAO DEFAULT");

lblPergunta.Content = "2 - A NASA tem planos concretos para
enviar missões\ncripuladas para Marte na década de 2030\ncom a SpaceX. Quem
fundou a SpaceX?";
lblAlternativaA.Content = "A) Michael “Lorde” Jackson";
lblAlternativaB.Content = "B) Neil Armstrong";
lblAlternativaC.Content = "C) Elon Musk";
lblAlternativaD.Content = "D) Sally Ride";

lblMensagem.Visibility = Visibility.Hidden;
lblMensagem.Content = "";
break;

case 3:

alternativaEscolhida = "";
respostaSelecionada = 0;

AtualizarBotoes("CAIR OPCAO DEFAULT");

lblPergunta.Content = "3 - Qual é o fenômeno que deixa o
planeta Marte com a sua\ncor vermelha?";
lblAlternativaA.Content = "A) O alto teor de carbono na
atmosfera";
lblAlternativaB.Content = "B) A presença de óxido de ferro em
sua superfície";
lblAlternativaC.Content = "C) A proximidade do Sol, causando
oxidação intensa";
lblAlternativaD.Content = "D) A presença de muito planeta
deixou com vergonha";

lblMensagem.Visibility = Visibility.Hidden;
lblMensagem.Content = "";
break;

case 4:

alternativaEscolhida = "";
respostaSelecionada = 0;

AtualizarBotoes("CAIR OPCAO DEFAULT");

lblPergunta.Content = "4 - Qual descoberta sobre Marte pode
indicar a possibilidade\nde vida no passado?";
lblAlternativaA.Content = "A) Evidências de água líquida em
rios e lagos secos.";
lblAlternativaB.Content = "B) A presença de metano na
atmosfera.";

```

```

        lblAlternativaC.Content = "C) Traços de fósseis de pequenos
organismos.";
        lblAlternativaD.Content = "D) A descoberta de grandes
depósitos de gelo.";

        lblMensagem.Visibility = Visibility.Hidden;
        lblMensagem.Content = "";
        break;

    case 5:
        alternativaEscolhida = "";
        respostaSelecionada = 0;

        AtualizarBotoes("CAIR OPCAO DEFAULT");

        lblPergunta.Content = "5 - Qual das seguintes tecnologias
será necessária para explorar\no Monte Olimpo em Marte?";
        lblAlternativaA.Content = "A) Foguetes de propulsão solar.";
        lblAlternativaB.Content = "B) Módulos infláveis para habitats
humanos.";
        lblAlternativaC.Content = "C) Missões robóticas para mapear a
superfície.";
        lblAlternativaD.Content = "D) Satélites com câmeras de alta
resolução para\nsobrevoos contínuos.";

        lblMensagem.Visibility = Visibility.Hidden;
        lblMensagem.Content = "";
        break;

    case 6:
        //tmrTempoAusencia.Enabled = false;

        Avaliacao avaliacao = new Avaliacao();
        QuestionarioFrame.NavigationService.Navigate(avaliacao);

        break;

    default:
        this.Close();
        break;
    }

}

private void pcbAlternativaA_MouseDown(object sender,
MouseEventArgs e)
{
    ResetarTimer();
    respostaSelecionada = 1;      //Feito para não ocorrer um clique duplo
    alternativaEscolhida = lblAlternativaA.Content.ToString();
    AtualizarBotoes("A");
}

private void pcbAlternativaB_MouseDown(object sender,
MouseEventArgs e)
{
    ResetarTimer();
    respostaSelecionada = 1;      //Feito para não ocorrer um clique duplo
    alternativaEscolhida = lblAlternativaB.Content.ToString();
    AtualizarBotoes("B");
}

```

```
}

    private void pcbAlternativaC_MouseDown(object sender,
MouseEventArgs e)
{
    ResetarTimer();
    respostaSelecionada = 1;      //Feito para não ocorrer um clique duplo
alternativaEscolhida = lblAlternativaC.Content.ToString();
AtualizarBotoes("C");
}

    private void pcbAlternativaD_MouseDown(object sender,
MouseEventArgs e)
{
    ResetarTimer();
    respostaSelecionada = 1;      //Feito para não ocorrer um clique duplo
alternativaEscolhida = lblAlternativaD.Content.ToString();
AtualizarBotoes("D");
}

    private void lblAlternativaA_MouseDown(object sender,
MouseEventArgs e)
{
    ResetarTimer();
    respostaSelecionada = 1;      //Feito para não ocorrer um clique duplo
alternativaEscolhida = lblAlternativaA.Content.ToString();
AtualizarBotoes("A");
}

    private void lblAlternativaB_MouseDown(object sender,
MouseEventArgs e)
{
    ResetarTimer();
    respostaSelecionada = 1;      //Feito para não ocorrer um clique duplo
alternativaEscolhida = lblAlternativaB.Content.ToString();
AtualizarBotoes("B");
}

    private void lblAlternativaC_MouseDown(object sender,
MouseEventArgs e)
{
    ResetarTimer();
    respostaSelecionada = 1;      //Feito para não ocorrer um clique duplo
alternativaEscolhida = lblAlternativaC.Content.ToString();
AtualizarBotoes("C");
}

    private void lblAlternativaD_MouseDown(object sender,
MouseEventArgs e)
{
    ResetarTimer();
    respostaSelecionada = 1;      //Feito para não ocorrer um clique duplo
alternativaEscolhida = lblAlternativaD.Content.ToString();
AtualizarBotoes("D");
}
```

```
e)    private void pcbContinuar_MouseDown(object sender, MouseButtonEventArgs e)
        {
            ResetarTimer();
            if (!respostaSelecionada.Equals(0))
            {
                Controle controle = new Controle();
                controle.ValidarResposta(idPergunta, alternativaEscolhida);
                idPergunta += 1;
                ProximaPergunta(idPergunta);
            }
            else
            {
                lblMensagem.Visibility = Visibility.Visible;
                lblMensagem.Content = "Selecione uma resposta!";
            }
        }

        private void btnMenu_Click(object sender, RoutedEventArgs e)
        {
            this.Close();
        }
    }
}
```

APÊNDICE AV – CODIGO TOTEM – AVALIACAO – XAML

```

<Page x:Class="ExplorandoMarteComTecnologia_WPF.Views.Avaliacao"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:local="clr-namespace:ExplorandoMarteComTecnologia_WPF.Views"
      mc:Ignorable="d"
      d:DesignHeight="768" d:DesignWidth="1366"
      Title="Avaliacao"
      Height="768" Width="1366">

    <Grid>
        <Grid.Background>
            <ImageBrush ImageSource="/Imagens/Fundos/Fundo5.jpg"/>
        </Grid.Background>

        <Label x:Name="lblPergunta" Content="lblPergunta" Margin="0,26,0,0"
               VerticalAlignment="Top" FontFamily="Arial" FontSize="30" FontWeight="Bold"
               Height="87" HorizontalAlignment="Center" Background="#FFE6E6E6"/>
        <Label x:Name="lblMensagem" Content="lblMensagem"
               HorizontalAlignment="Left" Margin="59,678,0,0" VerticalAlignment="Top"
               FontFamily="Arial" FontWeight="Bold" FontSize="24" Foreground="#FF900000"/>

        <Image x:Name="pcbExcelente" Height="100" Margin="941,218,280,0"
               VerticalAlignment="Top" Source="/Imagens/Botoes/Avaliacao/BotaoExcelente.png"
               MouseDown="pcbExcelente_MouseDown" Stretch="Fill"/>
        <Image x:Name="pcbBom" Height="100" Margin="776,218,445,0"
               VerticalAlignment="Top" Source="/Imagens/Botoes/Avaliacao/BotaoBom.png"
               MouseDown="pcbBom_MouseDown" Stretch="Fill"/>
        <Image x:Name="pcbRegular" HorizontalAlignment="Center" Height="100"
               Margin="0,218,0,0" VerticalAlignment="Top" Width="146"
               Source="/Imagens/Botoes/Avaliacao/BotaoRegular.png" Stretch="Fill"
               MouseDown="pcbRegular_MouseDown"/>
        <Image x:Name="pcbRuim" Height="100" Margin="445,218,776,0"
               VerticalAlignment="Top" Source="/Imagens/Botoes/Avaliacao/BotaoRuim.png"
               Stretch="Fill" MouseDown="pcbRuim_MouseDown" RenderTransformOrigin="0.508,0.52"/>
        <Image x:Name="pcbPessimo" Height="100" Margin="280,218,941,0"
               VerticalAlignment="Top" Source="/Imagens/Botoes/Avaliacao/BotaoPessimo.png"
               MouseDown="pcbPessimo_MouseDown" Stretch="Fill"/>

        <Image x:Name="pcbContinuar" HorizontalAlignment="Left" Height="94"
               Margin="1065,579,0,0" VerticalAlignment="Top" Width="237"
               Source="/Imagens/Botoes/BotaoContinuar.png" MouseDown="pcbContinuar_MouseDown"
               Stretch="Fill"/>
        <TextBox x:Name="txbComentario" HorizontalAlignment="Center" Height="186"
                  Margin="0,132,0,0" MaxLength="180" TextWrapping="Wrap" VerticalAlignment="Top"
                  Width="898" FontSize="28" FontWeight="Bold" FontFamily="Arial"/>

        <Image x:Name="pcbFundoElementos" HorizontalAlignment="Left" Height="346"
               Margin="78,327,0,0" VerticalAlignment="Top" Width="982"
               Source="/Imagens/Fundos/FundoDeElementos1.png" Stretch="Fill"/>

        <Image x:Name="pcbKeyA" HorizontalAlignment="Left" Height="82"
               Margin="120,430,0,0" VerticalAlignment="Top" Width="85"
               Source="/Imagens/Botoes/Teclado/Maiusculas/A.png" MouseDown="pcbKeyA_MouseDown"/>
        <Image x:Name="pcbKeyB" HorizontalAlignment="Left" Height="82"
               Margin="534,517,0,0" VerticalAlignment="Top" Width="84"
               Source="/Imagens/Botoes/Teclado/Maiusculas/B.png" MouseDown="pcbKeyB_MouseDown"/>

```

```

<Image x:Name="pcbKeyC" HorizontalAlignment="Left" Height="82"
Margin="360,517,0,0" VerticalAlignment="Top" Width="84"
Source="/Imagens/Botoes/Teclado/Maiuscidas/C.png" MouseDown="pcbKeyC_MouseDown"/>
<Image x:Name="pcbKeyD" HorizontalAlignment="Left" Height="82"
Margin="294,430,0,0" VerticalAlignment="Top" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/D.png" MouseDown="pcbKeyD_MouseDown"/>
<Image x:Name="pcbKeyE" HorizontalAlignment="Left" Height="82"
Margin="259,0,0,0" VerticalAlignment="Center" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/E.png" MouseDown="pcbKeyE_MouseDown"/>
<Image x:Name="pcbKeyF" HorizontalAlignment="Left" Height="82"
Margin="381,430,0,0" VerticalAlignment="Top" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/F.png" MouseDown="pcbKeyF_MouseDown"/>
<Image x:Name="pcbKeyG" HorizontalAlignment="Left" Height="82"
Margin="468,430,0,0" VerticalAlignment="Top" Width="84"
Source="/Imagens/Botoes/Teclado/Maiuscidas/G.png" MouseDown="pcbKeyG_MouseDown"/>
<Image x:Name="pcbKeyH" HorizontalAlignment="Left" Height="82"
Margin="555,430,0,0" VerticalAlignment="Top" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/H.png" MouseDown="pcbKeyH_MouseDown"/>
<Image x:Name="pcbKeyI" HorizontalAlignment="Left" Height="82"
Margin="694,0,0,0" VerticalAlignment="Center" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/I.png" MouseDown="pcbKeyI_MouseDown"/>
<Image x:Name="pcbKeyJ" HorizontalAlignment="Center" Height="82"
Margin="0,430,0,0" VerticalAlignment="Top" Width="84"
Source="/Imagens/Botoes/Teclado/Maiuscidas/J.png" MouseDown="pcbKeyJ_MouseDown"/>
<Image x:Name="pcbKeyK" HorizontalAlignment="Left" Height="82"
Margin="729,430,0,0" VerticalAlignment="Top" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/K.png" MouseDown="pcbKeyK_MouseDown"/>
<Image x:Name="pcbKeyL" HorizontalAlignment="Left" Height="82"
Margin="816,430,0,0" VerticalAlignment="Top" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/L.png" MouseDown="pcbKeyL_MouseDown"/>
<Image x:Name="pcbKeyM" HorizontalAlignment="Left" Height="82"
Margin="708,517,0,0" VerticalAlignment="Top" Width="84"
Source="/Imagens/Botoes/Teclado/Maiuscidas/M.png" MouseDown="pcbKeyM_MouseDown"/>
<Image x:Name="pcbKeyN" HorizontalAlignment="Left" Height="82"
Margin="620,517,0,0" VerticalAlignment="Top" Width="86"
Source="/Imagens/Botoes/Teclado/Maiuscidas/N.png" MouseDown="pcbKeyN_MouseDown"/>
<Image x:Name="pcbKeyO" HorizontalAlignment="Left" Height="82"
Margin="781,0,0,0" VerticalAlignment="Center" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/O.png" MouseDown="pcbKeyO_MouseDown"/>
<Image x:Name="pcbKeyP" HorizontalAlignment="Left" Height="82"
Margin="868,0,0,0" VerticalAlignment="Center" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/P.png" MouseDown="pcbKeyP_MouseDown"/>
<Image x:Name="pcbKeyQ" HorizontalAlignment="Left" Height="82"
Margin="85,0,0,0" VerticalAlignment="Center" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/Q.png" MouseDown="pcbKeyQ_MouseDown"/>
<Image x:Name="pcbKeyR" HorizontalAlignment="Left" Height="82"
Margin="346,0,0,0" VerticalAlignment="Center" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/R.png" MouseDown="pcbKeyR_MouseDown"/>
<Image x:Name="pcbKeyS" HorizontalAlignment="Left" Height="82"
Margin="207,430,0,0" VerticalAlignment="Top" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/S.png" MouseDown="pcbKeyS_MouseDown"/>
<Image x:Name="pcbKeyT" HorizontalAlignment="Left" Height="82"
Margin="433,0,0,0" VerticalAlignment="Center" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/T.png" MouseDown="pcbKeyT_MouseDown"/>
<Image x:Name="pcbKeyU" HorizontalAlignment="Left" Height="82"
Margin="607,0,0,0" VerticalAlignment="Center" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/U.png" MouseDown="pcbKeyU_MouseDown"/>
<Image x:Name="pcbKeyV" HorizontalAlignment="Left" Height="82"
Margin="446,517,0,0" VerticalAlignment="Top" Width="86"
Source="/Imagens/Botoes/Teclado/Maiuscidas/V.png" MouseDown="pcbKeyV_MouseDown"/>
<Image x:Name="pcbKeyW" HorizontalAlignment="Left" Height="82"
Margin="172,0,0,0" VerticalAlignment="Center" Width="85"
Source="/Imagens/Botoes/Teclado/Maiuscidas/W.png" MouseDown="pcbKeyW_MouseDown"/>

```

```
<Image x:Name="pcbKeyX" HorizontalAlignment="Left" Height="82"
Margin="272,517,0,0" VerticalAlignment="Top" Width="86"
Source="/Imagens/Botoes/Teclado/Maiusculas/X.png" MouseDown="pcbKeyX_MouseDown"/>
<Image x:Name="pcbKeyY" HorizontalAlignment="Left" Height="82"
Margin="520,0,0,0" VerticalAlignment="Center" Width="85"
Source="/Imagens/Botoes/Teclado/Maiusculas/Y.png" MouseDown="pcbKeyY_MouseDown"/>
<Image x:Name="pcbKeyZ" HorizontalAlignment="Left" Height="82"
Margin="186,517,0,0" VerticalAlignment="Top" Width="84"
Source="/Imagens/Botoes/Teclado/Maiusculas/Z.png" MouseDown="pcbKeyZ_MouseDown"/>

<Image x:Name="pcbKeyÇ" HorizontalAlignment="Left" Height="82"
Margin="903,430,0,0" VerticalAlignment="Top" Width="84"
Source="/Imagens/Botoes/Teclado/Maiusculas/Ç.png" MouseDown="pcbKeyÇ_MouseDown"/>
<Image x:Name="pcbKeySpace" HorizontalAlignment="Left" Height="54"
Margin="273,604,0,0" VerticalAlignment="Top" Width="576"
Source="/Imagens/Botoes/Teclado/Space.png" Stretch="Fill"
MouseDown="pcbKeySpace_MouseDown"/>

<Image x:Name="pcbKeyComma" HorizontalAlignment="Left" Height="82"
Margin="795,517,0,0" VerticalAlignment="Top" Width="84"
Source="/Imagens/Botoes/Teclado/Virgula.png" MouseDown="pcbKeyComma_MouseDown"/>
<Image x:Name="pcbKeyDot" HorizontalAlignment="Left" Height="82"
Margin="882,517,0,0" VerticalAlignment="Top" Width="84"
Source="/Imagens/Botoes/Teclado/Ponto.png" MouseDown="pcbKeyDot_MouseDown"/>
<Image x:Name="pcbKeyShift" HorizontalAlignment="Left" Height="82"
Margin="99,517,0,0" VerticalAlignment="Top" Width="84"
Source="/Imagens/Botoes/Teclado/ShiftAtivo.png"
MouseDown="pcbKeyShift_MouseDown"/>

<Image x:Name="pcbKeyBackspace" HorizontalAlignment="Left" Height="82"
Margin="955,0,0,0" VerticalAlignment="Center" Width="84"
Source="/Imagens/Botoes/Teclado/Backspace.png"
MouseDown="pcbKeyBackspace_MouseDown"/>

</Grid>
</Page>
```

APÊNDICE AW – CODIGO TOTEM – AVALIACAO – C#

```

using ExplorandoMarteComTecnologia_WPF.Controllers;
using ExplorandoMarteComTecnologia_WPF.DTO;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Input;
using System.Windows.Media.Imaging;
using System.Windows.Threading;

namespace ExplorandoMarteComTecnologia_WPF.Views
{
    /// <summary>
    /// Interação lógica para Avaliacao.xaml
    /// </summary>
    public partial class Avaliacao : Page
    {

        DispatcherTimer timer = new DispatcherTimer();
        public Avaliacao()
        {
            InitializeComponent();
            pcbFundoElementos.Visibility = Visibility.Hidden;
            TecladoVisivel(false);
            AvancarPergunta(perguntaId);
            ConfigurarTimer();
        }

        public string mensagemMainFrame = "";
        int perguntaId = 1;
        int respostaRespondida = 0;
        string respostaSelecionada = "";
        string notaQualidadeExposicoes = "";
        string notaInteracaoTotemSite = "";
        string notaInformacoesClaras = "";
        int shiftOpcão = 1;

        private void ConfigurarTimer()
        {
            timer.Interval = TimeSpan.FromMinutes(1); // 1 minuto
            timer.Tick += Timer_Tick;
            timer.Start();
        }

        private void Timer_Tick(object sender, EventArgs e)
        {
            // Código para retornar ao menu

            // Parar o timer após o tick
            timer.Stop();
            Window janela = Window.GetWindow(this);
            if (janela != null)
            {
                janela.Close();
            }
        }

        private void ResetarTimer()
        {
            timer.Stop(); // Para o timer
            timer.Start(); // Reinicia o timer, voltando a contar do zero
        }
    }
}

```

```

    }

    private void BotoesAvaliacaoVisivel(bool visivel)
    {
        switch (visivel)
        {
            case true:
                pcbPessimo.Visibility = Visibility.Visible;
                pcbRuim.Visibility = Visibility.Visible;
                pcbRegular.Visibility = Visibility.Visible;
                pcbBom.Visibility = Visibility.Visible;
                pcbExcelente.Visibility = Visibility.Visible;
                break;

            case false:
                pcbPessimo.Visibility = Visibility.Hidden;
                pcbRuim.Visibility = Visibility.Hidden;
                pcbRegular.Visibility = Visibility.Hidden;
                pcbBom.Visibility = Visibility.Hidden;
                pcbExcelente.Visibility = Visibility.Hidden;
                break;
        }
    }

    private void AtualizarBotoes(string alternativa)
    {
        string caminhoBotaoExcelente =
"/Imagens/Botoes/Avaliacao/BotaoExcelente.png";
        string caminhoBotaoBom = "/Imagens/Botoes/Avaliacao/BotaoBom.png";
        string caminhoBotaoRegular =
"/Imagens/Botoes/Avaliacao/BotaoRegular.png";
        string caminhoBotaoRuim = "/Imagens/Botoes/Avaliacao/BotaoRuim.png";
        string caminhoBotaoPessimo =
"/Imagens/Botoes/Avaliacao/BotaoPessimo.png";

        pcbExcelente.Source = new BitmapImage(new Uri(caminhoBotaoExcelente,
UriKind.Relative));
        pcbBom.Source = new BitmapImage(new Uri(caminhoBotaoBom,
UriKind.Relative));
        pcbRegular.Source = new BitmapImage(new Uri(caminhoBotaoRegular,
UriKind.Relative));
        pcbRuim.Source = new BitmapImage(new Uri(caminhoBotaoRuim,
UriKind.Relative));
        pcbPessimo.Source = new BitmapImage(new Uri(caminhoBotaoPessimo,
UriKind.Relative));

        switch (alternativa)
        {
            case "Excelente":
                string caminhoBotaoExcelenteSelecionado =
"/Imagens/Botoes/Avaliacao/BotaoExcelenteSelecionado.png";
                pcbExcelente.Source = new BitmapImage(new
Uri(caminhoBotaoExcelenteSelecionado, UriKind.Relative));
                break;
            case "Bom":
                string caminhoBotaoBomSelecionado =
"/Imagens/Botoes/Avaliacao/BotaoBomSelecionado.png";
                pcbBom.Source = new BitmapImage(new
Uri(caminhoBotaoBomSelecionado, UriKind.Relative));
                break;
            case "Regular":

```

```

        string caminhoBotaoRegularSelecionado =
"/Imagens/Botoes/Avaliacao/BotaoRegularSelecionado.png";
        pcbRegular.Source = new BitmapImage(new
Uri(caminhoBotaoRegularSelecionado, UriKind.Relative));
        break;
    case "Ruim":
        string caminhoBotaoRuimSelecionado =
"/Imagens/Botoes/Avaliacao/BotaoRuimSelecionado.png";
        pcbRuim.Source = new BitmapImage(new
Uri(caminhoBotaoRuimSelecionado, UriKind.Relative));
        break;
    case "Pessimo":
        string caminhoBotaoPessimoSelecionado =
"/Imagens/Botoes/Avaliacao/BotaoPessimoSelecionado.png";
        pcbPessimo.Source = new BitmapImage(new
Uri(caminhoBotaoPessimoSelecionado, UriKind.Relative));
        break;
    default:
        break;
    }
}

private void BotoesVisiveis(bool visivel)
{
    switch (visivel)
    {
        case true:
            pcbPessimo.Visibility = Visibility.Visible;
            pcbRuim.Visibility = Visibility.Visible;
            pcbRegular.Visibility = Visibility.Visible;
            pcbBom.Visibility = Visibility.Visible;
            pcbExcelente.Visibility = Visibility.Visible;
            break;

        case false:
            pcbPessimo.Visibility = Visibility.Hidden;
            pcbRuim.Visibility = Visibility.Hidden;
            pcbRegular.Visibility = Visibility.Hidden;
            pcbBom.Visibility = Visibility.Hidden;
            pcbExcelente.Visibility = Visibility.Hidden;
            break;
    }
}

private void AvancarPergunta(int perguntaId)
{
    switch (perguntaId)
    {
        case 1:
            TecladoVisivel(false);
            BotoesAvaliacaoVisivel(true);
            AtualizarBotoes("Cair num Default");
            respostaRespondida = 0;
            respostaSelecionada = "";
            lblPergunta.Content = "Como você avalia a qualidade
das\nexposições apresentadas no museu?";

            txbComentario.Visibility = Visibility.Hidden;
            BotoesVisiveis(true);

            lblMensagem.Visibility = Visibility.Hidden;
            lblMensagem.Content = "";
    }
}

```

```

        break;

    case 2:
        TecladoVisivel(false);
        BotoesAvaliacaoVisivel(true);
        AtualizarBotoes("Cair num Default");
        respostaRespondida = 0;
        respostaSelecionada = "";
        lblPergunta.Content = "Como você classifica a
experiência\ngeral de interação com o totém/site?";

        txbComentario.Visibility = Visibility.Hidden;
        BotoesVisiveis(true);

        lblMensagem.Visibility = Visibility.Hidden;
        lblMensagem.Content = "";

        break;

    case 3:
        TecladoVisivel(false);
        BotoesAvaliacaoVisivel(true);
        AtualizarBotoes("Cair num Default");
        respostaRespondida = 0;
        respostaSelecionada = "";
        lblPergunta.Content = "Como você avalia a clareza das
informações\nfornecidas nas descrições das obras?";

        txbComentario.Visibility = Visibility.Hidden;
        BotoesVisiveis(true);

        lblMensagem.Visibility = Visibility.Hidden;
        lblMensagem.Content = "";

        AtualizarTeclas(shiftOpcao);

        break;

    case 4:
        pcbFundoElementos.Visibility = Visibility.Visible;
        TecladoVisivel(true);
        AtualizarTeclas(shiftOpcao);
        BotoesAvaliacaoVisivel(false);
        AtualizarBotoes("Cair num Default");
        lblPergunta.Content = "Sua opinião é importante para
nós.\nDeixe seu comentário! (Opcional)";

        txbComentario.Visibility = Visibility.Visible;
        BotoesVisiveis(false);

        lblMensagem.Visibility = Visibility.Hidden;
        lblMensagem.Content = "";

        break;

    case 5:
        Estatico.avaliacaoRespostas.Add(new AvaliacaoRespostasDTO
{

```

```

        Id = 1,
        Pergunta = "QualidadeExposicoes",
        Excelente = notaQualidadeExposicoes == "Excelente" ? 1 :
0,
        Bom = notaQualidadeExposicoes == "Bom" ? 1 : 0,
        Regular = notaQualidadeExposicoes == "Regular" ? 1 : 0,
        Ruim = notaQualidadeExposicoes == "Ruim" ? 1 : 0,
        Pessimo = notaQualidadeExposicoes == "Pessimo" ? 1 : 0
    });

Estatico.avaliacaoRespostas.Add(new AvaliacaoRespostasDTO
{
    Id = 2,
    Pergunta = "InteracaoTotemSite",
    Excelente = notaInteracaoTotemSite == "Excelente" ? 1 :
0,
    Bom = notaInteracaoTotemSite == "Bom" ? 1 : 0,
    Regular = notaInteracaoTotemSite == "Regular" ? 1 : 0,
    Ruim = notaInteracaoTotemSite == "Ruim" ? 1 : 0,
    Pessimo = notaInteracaoTotemSite == "Pessimo" ? 1 : 0
});

Estatico.avaliacaoRespostas.Add(new AvaliacaoRespostasDTO
{
    Id = 3,
    Pergunta = "InformacoesClaras",
    Excelente = notaInformacoesClaras == "Excelente" ? 1 : 0,
    Bom = notaInformacoesClaras == "Bom" ? 1 : 0,
    Regular = notaInformacoesClaras == "Regular" ? 1 : 0,
    Ruim = notaInformacoesClaras == "Ruim" ? 1 : 0,
    Pessimo = notaInformacoesClaras == "Pessimo" ? 1 : 0
});

if (!string.IsNullOrEmpty(txbComentario.Text))
{
    Estatico.COMENTARIO = txbComentario.Text;
}

//tmrTempoAusencia.Enabled = false;

Window janela = Window.GetWindow(this);

Estatico.MENSAGEMCOMUNICACAO = "ARMAZENAR";

if (janela != null)
{
    janela.Close();
}

break;

default:
    perguntaId = 1;
    AvancarPergunta(1);
    break;
}

```

```

        }

    }

private void pcbPessimo_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    AtualizarBotoes("Pessimo");
    respostaRespondida = 1;
    respostaSelecionada = "Pessimo";
}

private void pcbRuim_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    AtualizarBotoes("Ruim");
    respostaRespondida = 1;
    respostaSelecionada = "Ruim";
}

private void pcbRegular_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    AtualizarBotoes("Regular");
    respostaRespondida = 1;
    respostaSelecionada = "Regular";
}

private void pcbBom_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    AtualizarBotoes("Bom");
    respostaRespondida = 1;
    respostaSelecionada = "Bom";
}

private void pcbExcelente_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    AtualizarBotoes("Excelente");
    respostaRespondida = 1;
    respostaSelecionada = "Excelente";
}

private void pcbContinuar_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (!string.IsNullOrEmpty(respostaSelecionada))
    {

        switch (perguntaId)
        {
            case 1:
                notaQualidadeExposicoes = respostaSelecionada;
                break;

            case 2:
                notaInteracaoTotemSite = respostaSelecionada;
                break;
        }
    }
}

```

```

        case 3:
            notaInformacoesClaras = respostaSelecionada;
            break;

        default:
            break;
    }

    perguntaId += 1;
    AvancarPergunta(perguntaId);
}
else
{
    lblMensagem.Visibility = Visibility.Visible;
    lblMensagem.Content = "Por favor, escolha uma opção.";
}
}

#endregion Teclado

private void TecladoVisivel(bool visivel)
{
    switch (visivel)
    {
        case true:
            pcbKeyA.Visibility = Visibility.Visible;
            pcbKeyB.Visibility = Visibility.Visible;
            pcbKeyC.Visibility = Visibility.Visible;
            pcbKeyD.Visibility = Visibility.Visible;
            pcbKeyE.Visibility = Visibility.Visible;
            pcbKeyF.Visibility = Visibility.Visible;
            pcbKeyG.Visibility = Visibility.Visible;
            pcbKeyH.Visibility = Visibility.Visible;
            pcbKeyI.Visibility = Visibility.Visible;
            pcbKeyJ.Visibility = Visibility.Visible;
            pcbKeyK.Visibility = Visibility.Visible;
            pcbKeyL.Visibility = Visibility.Visible;
            pcbKeyM.Visibility = Visibility.Visible;
            pcbKeyN.Visibility = Visibility.Visible;
            pcbKeyO.Visibility = Visibility.Visible;
            pcbKeyP.Visibility = Visibility.Visible;
            pcbKeyQ.Visibility = Visibility.Visible;
            pcbKeyR.Visibility = Visibility.Visible;
            pcbKeyS.Visibility = Visibility.Visible;
            pcbKeyT.Visibility = Visibility.Visible;
            pcbKeyU.Visibility = Visibility.Visible;
            pcbKeyV.Visibility = Visibility.Visible;
            pcbKeyW.Visibility = Visibility.Visible;
            pcbKeyX.Visibility = Visibility.Visible;
            pcbKeyY.Visibility = Visibility.Visible;
            pcbKeyZ.Visibility = Visibility.Visible;
            pcbKeyÇ.Visibility = Visibility.Visible;
            pcbKeyShift.Visibility = Visibility.Visible;
            pcbKeySpace.Visibility = Visibility.Visible;
            pcbKeyBackspace.Visibility = Visibility.Visible;
            pcbKeyComma.Visibility = Visibility.Visible;
            pcbKeyDot.Visibility = Visibility.Visible;
            break;
    }
}

```

```

        case false:
            pcbKeyA.Visibility = Visibility.Hidden;
            pcbKeyB.Visibility = Visibility.Hidden;
            pcbKeyC.Visibility = Visibility.Hidden;
            pcbKeyD.Visibility = Visibility.Hidden;
            pcbKeyE.Visibility = Visibility.Hidden;
            pcbKeyF.Visibility = Visibility.Hidden;
            pcbKeyG.Visibility = Visibility.Hidden;
            pcbKeyH.Visibility = Visibility.Hidden;
            pcbKeyI.Visibility = Visibility.Hidden;
            pcbKeyJ.Visibility = Visibility.Hidden;
            pcbKeyK.Visibility = Visibility.Hidden;
            pcbKeyL.Visibility = Visibility.Hidden;
            pcbKeyM.Visibility = Visibility.Hidden;
            pcbKeyN.Visibility = Visibility.Hidden;
            pcbKeyO.Visibility = Visibility.Hidden;
            pcbKeyP.Visibility = Visibility.Hidden;
            pcbKeyQ.Visibility = Visibility.Hidden;
            pcbKeyR.Visibility = Visibility.Hidden;
            pcbKeyS.Visibility = Visibility.Hidden;
            pcbKeyT.Visibility = Visibility.Hidden;
            pcbKeyU.Visibility = Visibility.Hidden;
            pcbKeyV.Visibility = Visibility.Hidden;
            pcbKeyW.Visibility = Visibility.Hidden;
            pcbKeyX.Visibility = Visibility.Hidden;
            pcbKeyY.Visibility = Visibility.Hidden;
            pcbKeyZ.Visibility = Visibility.Hidden;
            pcbKeyÇ.Visibility = Visibility.Hidden;
            pcbKeyShift.Visibility = Visibility.Hidden;
            pcbKeySpace.Visibility = Visibility.Hidden;
            pcbKeyBackspace.Visibility = Visibility.Hidden;
            pcbKeyComma.Visibility = Visibility.Hidden;
            pcbKeyDot.Visibility = Visibility.Hidden;

            break;
        }
    }

    public void AtualizarTeclas(int shiftOpcão)
    {
        switch (shiftOpcão)
        {
            case 0:
                pcbKeyA.Source = new BitmapImage(new
                    Uri("/Imagens/Botoes/Teclado/Minusculo/a.png", UriKind.Relative));
                pcbKeyB.Source = new BitmapImage(new
                    Uri("/Imagens/Botoes/Teclado/Minusculo/b.png", UriKind.Relative));
                pcbKeyC.Source = new BitmapImage(new
                    Uri("/Imagens/Botoes/Teclado/Minusculo/c.png", UriKind.Relative));
                pcbKeyD.Source = new BitmapImage(new
                    Uri("/Imagens/Botoes/Teclado/Minusculo/d.png", UriKind.Relative));
                pcbKeyE.Source = new BitmapImage(new
                    Uri("/Imagens/Botoes/Teclado/Minusculo/e.png", UriKind.Relative));
                pcbKeyF.Source = new BitmapImage(new
                    Uri("/Imagens/Botoes/Teclado/Minusculo/f.png", UriKind.Relative));
                pcbKeyG.Source = new BitmapImage(new
                    Uri("/Imagens/Botoes/Teclado/Minusculo/g.png", UriKind.Relative));
                pcbKeyH.Source = new BitmapImage(new
                    Uri("/Imagens/Botoes/Teclado/Minusculo/h.png", UriKind.Relative));
                pcbKeyI.Source = new BitmapImage(new
                    Uri("/Imagens/Botoes/Teclado/Minusculo/i.png", UriKind.Relative));
                pcbKeyJ.Source = new BitmapImage(new
                    Uri("/Imagens/Botoes/Teclado/Minusculo/j.png", UriKind.Relative));

```

```

        pcbKeyK.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/k.png", UriKind.Relative));
        pcbKeyL.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/l.png", UriKind.Relative));
        pcbKeyM.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/m.png", UriKind.Relative));
        pcbKeyN.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/n.png", UriKind.Relative));
        pcbKeyO.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/o.png", UriKind.Relative));
        pcbKeyP.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/p.png", UriKind.Relative));
        pcbKeyQ.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/q.png", UriKind.Relative));
        pcbKeyR.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/r.png", UriKind.Relative));
        pcbKeyS.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/s.png", UriKind.Relative));
        pcbKeyT.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/t.png", UriKind.Relative));
        pcbKeyU.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/u.png", UriKind.Relative));
        pcbKeyV.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/v.png", UriKind.Relative));
        pcbKeyW.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/w.png", UriKind.Relative));
        pcbKeyX.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/x.png", UriKind.Relative));
        pcbKeyY.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/y.png", UriKind.Relative));
        pcbKeyZ.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/z.png", UriKind.Relative));
        pcbKeyÇ.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Minusculo/ç.png", UriKind.Relative));
        pcbKeyShift.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/ShiftDesligado.png", UriKind.Relative));
        break;

    case 1:
        pcbKeyA.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/A.png", UriKind.Relative));
        pcbKeyB.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/B.png", UriKind.Relative));
        pcbKeyC.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/C.png", UriKind.Relative));
        pcbKeyD.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/D.png", UriKind.Relative));
        pcbKeyE.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/E.png", UriKind.Relative));
        pcbKeyF.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/F.png", UriKind.Relative));
        pcbKeyG.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/G.png", UriKind.Relative));
        pcbKeyH.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/H.png", UriKind.Relative));
        pcbKeyI.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/I.png", UriKind.Relative));
        pcbKeyJ.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/J.png", UriKind.Relative));
        pcbKeyK.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/K.png", UriKind.Relative));
        pcbKeyL.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/L.png", UriKind.Relative));

```

```

        pcbKeyM.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/M.png", UriKind.Relative));
        pcbKeyN.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/N.png", UriKind.Relative));
        pcbKeyO.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/O.png", UriKind.Relative));
        pcbKeyP.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/P.png", UriKind.Relative));
        pcbKeyQ.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/Q.png", UriKind.Relative));
        pcbKeyR.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/R.png", UriKind.Relative));
        pcbKeyS.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/S.png", UriKind.Relative));
        pcbKeyT.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/T.png", UriKind.Relative));
        pcbKeyU.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/U.png", UriKind.Relative));
        pcbKeyV.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/V.png", UriKind.Relative));
        pcbKeyW.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/W.png", UriKind.Relative));
        pcbKeyX.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/X.png", UriKind.Relative));
        pcbKeyY.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/Y.png", UriKind.Relative));
        pcbKeyZ.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/Z.png", UriKind.Relative));
        pcbKeyÇ.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/Maiusculas/Ç.png", UriKind.Relative));
        pcbKeyShift.Source = new BitmapImage(new
Uri("/Imagenes/Botoes/Teclado/ShiftAtivo.png", UriKind.Relative));
        break;
    }
}

private void pcbKeyBackspace_MouseDown(object sender,
MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length > 0)
    {
        txbComentario.Text = txbComentario.Text.Substring(0,
txbComentario.Text.Length - 1);
    }
}

private void pcbKeyA_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcao)
        {
            case 0:
                txbComentario.Text += controle.Teclado(27);
                break;
            case 1:
                txbComentario.Text += controle.Teclado(1);
                break;
        }
    }
}

```

```

        }
    }

private void pcbKeyB_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(28);
                break;
            case 1:
                txbComentario.Text += controle.Teclado(2);
                break;
        }
    }
}

private void pcbKeyC_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(29);
                break;
            case 1:
                txbComentario.Text += controle.Teclado(3);
                break;
        }
    }
}

private void pcbKeyD_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(30);
                break;
            case 1:
                txbComentario.Text += controle.Teclado(4);
                break;
        }
    }
}

private void pcbKeyE_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {

```

```

        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentário.Text += controle.Teclado(31);
                break;
            case 1:
                txbComentário.Text += controle.Teclado(5);
                break;
        }
    }

private void pcbKeyF_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentário.Text.Length < txbComentário.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentário.Text += controle.Teclado(32);
                break;
            case 1:
                txbComentário.Text += controle.Teclado(6);
                break;
        }
    }
}

private void pcbKeyG_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentário.Text.Length < txbComentário.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentário.Text += controle.Teclado(33);
                break;
            case 1:
                txbComentário.Text += controle.Teclado(7);
                break;
        }
    }
}

private void pcbKeyH_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentário.Text.Length < txbComentário.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentário.Text += controle.Teclado(34);
                break;
            case 1:
                txbComentário.Text += controle.Teclado(8);
                break;
        }
    }
}

```

```
        }
    }

private void pcbKeyI_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(35);
                break;
            case 1:
                txbComentario.Text += controle.Teclado(9);
                break;
        }
    }
}

private void pcbKeyJ_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(36);
                break;
            case 1:
                txbComentario.Text += controle.Teclado(10);
                break;
        }
    }
}

private void pcbKeyK_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(37);
                break;
            case 1:
                txbComentario.Text += controle.Teclado(11);
                break;
        }
    }
}

private void pcbKeyL_MouseDown(object sender, MouseButtonEventArgs e)
{
```

```

ResetarTimer();
if (txbComentario.Text.Length < txbComentario.MaxLength)
{
    Controle controle = new Controle();
    switch (shiftOpcão)
    {
        case 0:
            txbComentario.Text += controle.Teclado(38);
            break;

        case 1:
            txbComentario.Text += controle.Teclado(12);
            break;
    }
}

private void pcbKeyM_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(39);
                break;

            case 1:
                txbComentario.Text += controle.Teclado(13);
                break;
        }
    }
}

private void pcbKeyN_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(40);
                break;

            case 1:
                txbComentario.Text += controle.Teclado(14);
                break;
        }
    }
}

private void pcbKey0_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {

```

```

        case 0:
            txbComentario.Text += controle.Teclado(41);
            break;

        case 1:
            txbComentario.Text += controle.Teclado(15);
            break;
    }
}

private void pcbKeyP_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(42);
                break;

            case 1:
                txbComentario.Text += controle.Teclado(16);
                break;
        }
    }
}

private void pcbKeyQ_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(43);
                break;

            case 1:
                txbComentario.Text += controle.Teclado(17);
                break;
        }
    }
}

private void pcbKeyR_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(44);
                break;

            case 1:
                txbComentario.Text += controle.Teclado(18);
        }
    }
}

```

```
                break;
            }
        }

private void pcbKeyS_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(45);
                break;

            case 1:
                txbComentario.Text += controle.Teclado(19);
                break;
        }
    }
}

private void pcbKeyT_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(46);
                break;

            case 1:
                txbComentario.Text += controle.Teclado(20);
                break;
        }
    }
}

private void pcbKeyU_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(47);
                break;

            case 1:
                txbComentario.Text += controle.Teclado(21);
                break;
        }
    }
}

private void pcbKeyV_MouseDown(object sender, MouseButtonEventArgs e)
```

```

{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(48);
                break;

            case 1:
                txbComentario.Text += controle.Teclado(22);
                break;
        }
    }
}

private void pcbKeyW_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(49);
                break;

            case 1:
                txbComentario.Text += controle.Teclado(23);
                break;
        }
    }
}

private void pcbKeyX_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
        {
            case 0:
                txbComentario.Text += controle.Teclado(50);
                break;

            case 1:
                txbComentario.Text += controle.Teclado(24);
                break;
        }
    }
}

private void pcbKeyY_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        switch (shiftOpcão)
    }
}

```

```
{  
    case 0:  
        txbComentario.Text += controle.Teclado(51);  
        break;  
  
    case 1:  
        txbComentario.Text += controle.Teclado(25);  
        break;  
    }  
}  
  
private void pcbKeyZ_MouseDown(object sender, MouseButtonEventArgs e)  
{  
    ResetarTimer();  
    if (txbComentario.Text.Length < txbComentario.MaxLength)  
    {  
        Controle controle = new Controle();  
        switch (shiftOpcão)  
        {  
            case 0:  
                txbComentario.Text += controle.Teclado(52);  
                break;  
  
            case 1:  
                txbComentario.Text += controle.Teclado(26);  
                break;  
        }  
    }  
}  
  
private void pcbKeyÇ_MouseDown(object sender, MouseButtonEventArgs e)  
{  
    ResetarTimer();  
    if (txbComentario.Text.Length < txbComentario.MaxLength)  
    {  
        Controle controle = new Controle();  
        switch (shiftOpcão)  
        {  
            case 0:  
                txbComentario.Text += controle.Teclado(57);  
                break;  
  
            case 1:  
                txbComentario.Text += controle.Teclado(56);  
                break;  
        }  
    }  
}  
  
private void pcbKeyComma_MouseDown(object sender, MouseButtonEventArgs e)  
{  
    ResetarTimer();  
    if (txbComentario.Text.Length < txbComentario.MaxLength)  
    {  
        Controle controle = new Controle();  
        txbComentario.Text += controle.Teclado(53);  
    }  
}  
  
private void pcbKeyDot_MouseDown(object sender, MouseButtonEventArgs e)  
{  
    ResetarTimer();  
}
```

```
if (txbComentario.Text.Length < txbComentario.MaxLength)
{
    Controle controle = new Controle();
    txbComentario.Text += controle.Teclado(54);
}
}

private void pcbKeySpace_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    if (txbComentario.Text.Length < txbComentario.MaxLength)
    {
        Controle controle = new Controle();
        txbComentario.Text += controle.Teclado(55);
    }
}

private void pcbKeyShift_MouseDown(object sender, MouseButtonEventArgs e)
{
    ResetarTimer();
    switch (shiftOpcão)
    {
        case 0:
            shiftOpcão = 1;
            break;

        case 1:
            shiftOpcão = 0;
            break;

        default:
            shiftOpcão = 1;
            break;
    }
    AtualizarTeclas(shiftOpcão);
}

#endregion

}
```

APÊNDICE AX – CODIGO TOTEM – RELATORIORESPOSTAS – XAML

```

<Page x:Class="ExplorandoMarteComTecnologia_WPF.Views.RelatorioRespostas"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:local="clr-namespace:ExplorandoMarteComTecnologia_WPF.Views"
    Unloaded="OnPageUnloaded"
    mc:Ignorable="d"
    d:DesignHeight="768" d:DesignWidth="1366"
    Title="RelatorioRespostas">

    <Grid>
        <Grid.Background>
            <ImageBrush ImageSource="/Imagens/Fundos/Fundo4.png"/>
        </Grid.Background>

        <Label x:Name="lblRespostaSelecionada1" Content="Resposta"
            HorizontalAlignment="Left" Margin="97,148,0,0" VerticalAlignment="Top"
            Foreground="#FF37F703" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
            <Label x:Name="lblRespostaCerta1" Content="D) Monte Olimpo, com 27 km de
            altura" HorizontalAlignment="Left" Margin="97,188,0,0" VerticalAlignment="Top"
            Foreground="#FF55F110" Visibility="Hidden" FontFamily="Arial" FontSize="26"
            Background="#FF1B1B1B"/>

            <Image x:Name="imgRC1" HorizontalAlignment="Left" Height="27"
            Margin="97,228,0,0" VerticalAlignment="Top" Width="136"
            Source="/Imagens/Relatorio/RespostaCerta.png" Visibility="Hidden"/>

            <Label x:Name="lblRespostaSelecionada2" Content="Resposta"
            HorizontalAlignment="Left" Margin="97,315,0,0" VerticalAlignment="Top"
            Foreground="#FF37F703" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
            <Label x:Name="lblRespostaCerta2" Content="C) Elon Musk"
            HorizontalAlignment="Left" Margin="97,355,0,0" VerticalAlignment="Top"
            Foreground="#FF55F110" Visibility="Hidden" FontFamily="Arial" FontSize="26"
            Background="#FF1B1B1B"/>

            <Image x:Name="imgRC2" HorizontalAlignment="Left" Height="27"
            Margin="97,395,0,0" VerticalAlignment="Top" Width="136"
            Source="/Imagens/Relatorio/RespostaCerta.png" Visibility="Hidden"/>

            <Label x:Name="lblRespostaSelecionada3" Content="Resposta"
            HorizontalAlignment="Left" Margin="97,481,0,0" VerticalAlignment="Top"
            Foreground="#FF37F703" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
            <Label x:Name="lblRespostaCerta3" Content="B) A presença de óxido de
            ferro em sua superfície" HorizontalAlignment="Left" Margin="97,521,0,0"
            VerticalAlignment="Top" Foreground="#FF55F110" Visibility="Hidden"
            FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>

            <Image x:Name="imgRC3" HorizontalAlignment="Left" Height="27"
            Margin="97,561,0,0" VerticalAlignment="Top" Width="136"
            Source="/Imagens/Relatorio/RespostaCerta.png" Visibility="Hidden"/>

            <Label x:Name="lblRespostaSelecionada4" Content="Resposta"
            HorizontalAlignment="Left" Margin="97,644,0,0" VerticalAlignment="Top"
            Foreground="#FF37F703" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
            <Label x:Name="lblRespostaCerta4" Content="A) Evidências de água líquida
            em rios e lagos secos." HorizontalAlignment="Left" Margin="97,684,0,0"
            
```

```

VerticalAlignment="Top" Foreground="#FF55F110" Visibility="Hidden"
FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>

    <Image x:Name="imgRC4" HorizontalAlignment="Left" Height="27"
Margin="97,724,0,0" VerticalAlignment="Top" Width="136"
Source="/Imagens/Relatorio/RespostaCerta.png" Visibility="Hidden"/>

    <Label x:Name="lblRespostaSelecionada5" Content="Resposta"
HorizontalAlignment="Left" Margin="684,155,0,0" VerticalAlignment="Top"
Foreground="#FF37F703" FontFamily="Arial" FontSize="20" Background="#FF1B1B1B"/>
        <Label x:Name="lblRespostaCertas5" Content="C) Missões robóticas para
mapear a superfície." HorizontalAlignment="Left" Margin="684,188,0,0"
VerticalAlignment="Top" Foreground="#FF55F110" Visibility="Hidden"
FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>

    <Image x:Name="imgRC5" HorizontalAlignment="Left" Height="27"
Margin="683,228,0,0" VerticalAlignment="Top" Width="136"
Source="/Imagens/Relatorio/RespostaCerta.png" Visibility="Hidden"/>




    <Label x:Name="lblPessoasTotalQuestao1" Content="Quantidade"
HorizontalAlignment="Left" Margin="846,108,0,0" VerticalAlignment="Top"
Foreground="White" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
        <Label x:Name="lblPessoasTotalQuestao2" Content="Quantidade"
HorizontalAlignment="Left" Margin="846,224,0,0" VerticalAlignment="Top"
Foreground="White" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
        <Label x:Name="lblPessoasTotalQuestao3" Content="Quantidade"
HorizontalAlignment="Left" Margin="846,335,0,0" VerticalAlignment="Top"
Foreground="White" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
        <Label x:Name="lblPessoasTotalQuestao4" Content="Quantidade"
HorizontalAlignment="Left" Margin="846,450,0,0" VerticalAlignment="Top"
Foreground="White" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
        <Label x:Name="lblPessoasTotalQuestao5" Content="Quantidade"
HorizontalAlignment="Left" Margin="846,564,0,0" VerticalAlignment="Top"
Foreground="White" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>

    <Label x:Name="lblSuasRespostas" Content="Suas Respostas"
HorizontalAlignment="Left" Margin="14,12,0,0" VerticalAlignment="Top"
Foreground="White" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
        <Label x:Name="lblPerguntaNumeracao1" Content="1)"
HorizontalAlignment="Left" Margin="97,108,0,0" VerticalAlignment="Top"
Foreground="White" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
        <Label x:Name="lblPerguntaNumeracao2" Content="2)"
HorizontalAlignment="Left" Margin="97,275,0,0" VerticalAlignment="Top"
Foreground="White" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
        <Label x:Name="lblPerguntaNumeracao3" Content="3)"
HorizontalAlignment="Left" Margin="97,441,0,0" VerticalAlignment="Top"
Foreground="White" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
        <Label x:Name="lblPerguntaNumeracao4" Content="4)"
HorizontalAlignment="Left" Margin="97,604,0,0" VerticalAlignment="Top"
Foreground="White" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
        <Label x:Name="lblPerguntaNumeracao5" Content="5)"
HorizontalAlignment="Left" Margin="684,116,0,0" VerticalAlignment="Top"
Foreground="White" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>
        <Label x:Name="lblEstatisticas" Content="Estatísticas"
HorizontalAlignment="Left" Margin="14,12,0,0" VerticalAlignment="Top"
Foreground="White" FontFamily="Arial" FontSize="26" Background="#FF1B1B1B"/>

```

```
<Image x:Name="pcbContinuar" Margin="0,0,49,50"
Source="/Imagens/Botoes/BotaoContinuar.png" MouseDown="pcbContinuar_MouseDown"
Height="120" VerticalAlignment="Bottom" HorizontalAlignment="Right" Width="316"/>
<Image x:Name="imgAgradecimento" Margin="0,0,0,0" Stretch="Fill"
Source="/Imagens/Fundos/Fundo7.png" Visibility="Hidden"
MouseDown="imgAgradecimento_MouseDown"/>

<Image x:Name="imgStatusPergunta1" HorizontalAlignment="Left" Height="40"
Margin="52,108,0,0" VerticalAlignment="Top" Width="40"
Source="/Imagens/Relatorio/Certo.png"/>
<Image x:Name="imgStatusPergunta2" HorizontalAlignment="Left" Height="40"
Margin="52,275,0,0" VerticalAlignment="Top" Width="40"
Source="/Imagens/Relatorio/Certo.png"/>
<Image x:Name="imgStatusPergunta3" HorizontalAlignment="Left" Height="40"
Margin="52,441,0,0" VerticalAlignment="Top" Width="40"
Source="/Imagens/Relatorio/Certo.png"/>
<Image x:Name="imgStatusPergunta4" HorizontalAlignment="Left" Height="40"
Margin="52,604,0,0" VerticalAlignment="Top" Width="40"
Source="/Imagens/Relatorio/Certo.png"/>
<Image x:Name="imgStatusPergunta5" HorizontalAlignment="Left" Height="40"
Margin="639,116,0,0" VerticalAlignment="Top" Width="40"
Source="/Imagens/Relatorio/Certo.png"/>

</Grid>
</Page>
```

APÊNDICE AY – CODIGO TOTEM – RELATORIORESPOSTAS – C#

```

using ExplorandoMarteComTecnologia_WPF.Controllers;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Threading;

namespace ExplorandoMarteComTecnologia_WPF.Views
{
    /// <summary>
    /// Interação lógica para RelatorioRespostas.xaml
    /// </summary>
    public partial class RelatorioRespostas : Page
    {
        private List<Label> lblPerguntaNumeracao;
        DispatcherTimer timer = new DispatcherTimer();

        public RelatorioRespostas()
        {
            InitializeComponent();

            lblRespostaSelecionada1.Content = Estatico.TEMPRESPOSTAQUEST1;
            lblRespostaSelecionada2.Content = Estatico.TEMPRESPOSTAQUEST2;
            lblRespostaSelecionada3.Content = Estatico.TEMPRESPOSTAQUEST3;
            lblRespostaSelecionada4.Content = Estatico.TEMPRESPOSTAQUEST4;
            lblRespostaSelecionada5.Content = Estatico.TEMPRESPOSTAQUEST5;

            Estatico.RELATORIOPAGINA = 1;

            Window janela = Window.GetWindow(this);
            Controle controle = new Controle();

            controle.AtualizarValoresAcertosErrosPerguntas("/api/Master/QuestionarioAvaliacao/q");
            ProximaPagina(Estatico.RELATORIOPAGINA);
            ConfigurarTimer();
        }

        string questao1 = "Acertou";
        string questao2 = "Acertou";
        string questao3 = "Acertou";
        string questao4 = "Acertou";
        string questao5 = "Acertou";

        private bool isTicking = false;

        private void ResetarTimer()
        {
            timer.Stop(); // Para o timer
            timer.Start(); // Reinicia o timer, voltando a contar do zero
        }

        private void ConfigurarTimer()
        {
            Estatico.RELATORIOPAGINA = 1;
        }
    }
}

```

```

        timer.Interval = TimeSpan.FromMinutes(1); // 1 minuto
        timer.Tick += Timer_Tick;
        timer.Start();
    }

private void Timer_Tick(object sender, EventArgs e)
{
    // Se o timer já estiver "rodando", não faça nada
    if (isTicking)
        return;

    // Marca o início do tick
    isTicking = true;

    // Exibe o pop-up de debug

    // A lógica para incrementar a página e chamar a próxima
    Estatico.RELATORIOPAGINA += 1;

    // Chama a função para exibir a próxima página
    ProximaPagina(Estatico.RELATORIOPAGINA);

    // Se a página atingir 3, interrompe o timer
    if (Estatico.RELATORIOPAGINA >= 4)
    {
        timer.Stop();
    }

    // Marca o fim do tick
    isTicking = false;
}

private void OnPageUnloaded(object sender, RoutedEventArgs e)
{
    // Para o timer quando a página é descarregada
    timer.Stop();
}

private void ChecarRespostas()
{
    // Comparar o conteúdo como string
    if (lblRespostaSelecionada1.Content.ToString() !=
    lblRespostaCerta1.Content.ToString())
    {
        questao1 = "Errou";
        lblRespostaSelecionada1.Foreground = new
        SolidColorBrush(Colors.Red);
        lblRespostaCerta1.Visibility = Visibility.Visible;
        imgRC1.Visibility = Visibility.Visible;
        imgStatusPergunta1.Source = new BitmapImage(new
        Uri("/Imagens/Relatorio/Errado.png", UriKind.Relative));
    }

    if (lblRespostaSelecionada2.Content.ToString() !=
    lblRespostaCerta2.Content.ToString())
    {
        questao2 = "Errou";
        lblRespostaSelecionada2.Foreground = new
        SolidColorBrush(Colors.Red);
        lblRespostaCerta2.Visibility = Visibility.Visible;
        imgRC2.Visibility = Visibility.Visible;
        imgStatusPergunta2.Source = new BitmapImage(new
        Uri("/Imagens/Relatorio/Errado.png", UriKind.Relative));
    }
}

```

```

        }

        if (lblRespostaSelecionada3.Content.ToString() != 
lblRespostaCerta3.Content.ToString())
{
    questao3 = "Errou";
    lblRespostaSelecionada3.Foreground = new
SolidColorBrush(Colors.Red);
    lblRespostaCerta3.Visibility = Visibility.Visible;
    imgRC3.Visibility = Visibility.Visible;
    imgStatusPergunta3.Source = new BitmapImage(new
Uri("/Imagenes/Relatorio/Errado.png", UriKind.Relative));
}

        if (lblRespostaSelecionada4.Content.ToString() != 
lblRespostaCerta4.Content.ToString())
{
    questao4 = "Errou";
    lblRespostaSelecionada4.Foreground = new
SolidColorBrush(Colors.Red);
    lblRespostaCerta4.Visibility = Visibility.Visible;
    imgRC4.Visibility = Visibility.Visible;
    imgStatusPergunta4.Source = new BitmapImage(new
Uri("/Imagenes/Relatorio/Errado.png", UriKind.Relative));
}

        if (lblRespostaSelecionada5.Content.ToString() != 
lblRespostaCerta5.Content.ToString())
{
    questao5 = "Errou";
    lblRespostaSelecionada5.Foreground = new
SolidColorBrush(Colors.Red);
    lblRespostaCerta5.Visibility = Visibility.Visible;
    imgRC5.Visibility = Visibility.Visible;
    imgStatusPergunta5.Source = new BitmapImage(new
Uri("/Imagenes/Relatorio/Errado.png", UriKind.Relative));
}

public void AtualizarEstatisticaGlobal()
{
    Random random = new Random();

    // Lista de perguntas com acertos e erros
    string[] perguntas = { questao1, questao2, questao3, questao4,
questao5 };
    int[] acertos = { Estatico.QUANTIDADEACERTOSQUEST1,
Estatico.QUANTIDADEACERTOSQUEST2, Estatico.QUANTIDADEACERTOSQUEST3,
Estatico.QUANTIDADEACERTOSQUEST4, Estatico.QUANTIDADEACERTOSQUEST5 };
    int[] erros = { Estatico.QUANTIDADEERROSQUEST1,
Estatico.QUANTIDADEERROSQUEST2, Estatico.QUANTIDADEERROSQUEST3,
Estatico.QUANTIDADEERROSQUEST4, Estatico.QUANTIDADEERROSQUEST5 };
    int[] totais = { Estatico.QUANTIDADEACERTOSQUEST1 +
Estatico.QUANTIDADEERROSQUEST1, Estatico.QUANTIDADEACERTOSQUEST2 +
Estatico.QUANTIDADEERROSQUEST2, Estatico.QUANTIDADEACERTOSQUEST3 +
Estatico.QUANTIDADEERROSQUEST3, Estatico.QUANTIDADEACERTOSQUEST4 +
Estatico.QUANTIDADEERROSQUEST4, Estatico.QUANTIDADEACERTOSQUEST5 +
Estatico.QUANTIDADEERROSQUEST5 };

    for (int i = 0; i < perguntas.Length; i++)
{
    if (perguntas[i] == "Acertou")

```

```

    {
        int fraseIndex = random.Next(1, 4);
        double porcentagemAcerto = Math.Round((double)acertos[i] /
totais[i] * 100, 1);

        switch (fraseIndex)
        {
            case 1:
                string fraseA1 = $"{acertos[i]} Pessoa(s) acertaram
esta questão, e você é uma delas!";
                InserirFrase(fraseA1, i);
                break;
            case 2:
                string fraseA2 = $"Você está entre {acertos[i]} das
pessoas que acertaram esta pergunta!";
                InserirFrase(fraseA2, i);
                break;
            case 3:
                string fraseA3 = $"Você está entre os
{porcentagemAcerto}% que acertaram esta pergunta!";
                InserirFrase(fraseA3, i);
                break;
            default:
                //MessageBox.Show("Sem frase");
                break;
        }
    }
    else if (perguntas[i] == "Errou")
    {
        int fraseIndex = random.Next(1, 4);
        double porcentagemErro = Math.Round((double)erros[i] /
totais[i] * 100, 1);

        switch (fraseIndex)
        {
            case 1:
                string fraseE1 = $"{erros[i]} Pessoa(s) erraram esta
questão!";
                InserirFrase(fraseE1, i);
                break;
            case 2:
                string fraseE2 = $"Não se preocupe, {erros[i]}
pessoa(s) também erraram!";
                InserirFrase(fraseE2, i);
                break;
            case 3:
                string fraseE3 = $"{porcentagemErro}% das pessoas
erraram esta pergunta!";
                InserirFrase(fraseE3, i);
                break;
            default:
                //MessageBox.Show("Sem frase");
                break;
        }
    }
}

private void InserirFrase(string frase, int lblIndex)
{
    switch (lblIndex)
    {
        case 0:

```

```

        lblPessoasTotalQuestao1.Content = frase;
        break;
    case 1:
        lblPessoasTotalQuestao2.Content = frase;
        break;
    case 2:
        lblPessoasTotalQuestao3.Content = frase;
        break;
    case 3:
        lblPessoasTotalQuestao4.Content = frase;
        break;
    case 4:
        lblPessoasTotalQuestao5.Content = frase;
        break;

    default:
        break;
    }
}

public void ProximaPagina(int pagina)
{
    switch (pagina)
    {
        case 1:

            ConfigurarTimer();
            ResetarTimer();
            ChecarRespostas();
            lblSuasRespostas.Visibility = Visibility.Visible;

            lblPerguntaNumeracao1.Visibility = Visibility.Visible;
            lblPerguntaNumeracao2.Visibility = Visibility.Visible;
            lblPerguntaNumeracao3.Visibility = Visibility.Visible;
            lblPerguntaNumeracao4.Visibility = Visibility.Visible;
            lblPerguntaNumeracao5.Visibility = Visibility.Visible;

            // Exibindo as respostas selecionadas
            lblRespostaSelecionada1.Visibility = Visibility.Visible;
            lblRespostaSelecionada2.Visibility = Visibility.Visible;
            lblRespostaSelecionada3.Visibility = Visibility.Visible;
            lblRespostaSelecionada4.Visibility = Visibility.Visible;
            lblRespostaSelecionada5.Visibility = Visibility.Visible;

            lblEstatisticas.Visibility = Visibility.Hidden;

            // Ocultando o total de pessoas por questão
            lblPessoasTotalQuestao1.Visibility = Visibility.Hidden;
            lblPessoasTotalQuestao2.Visibility = Visibility.Hidden;
            lblPessoasTotalQuestao3.Visibility = Visibility.Hidden;
            lblPessoasTotalQuestao4.Visibility = Visibility.Hidden;
            lblPessoasTotalQuestao5.Visibility = Visibility.Hidden;

            imgAgradecimento.Visibility = Visibility.Hidden;

            break;

        case 2:

            AtualizarEstatisticaGlobal();

            lblSuasRespostas.Visibility = Visibility.Hidden;
    }
}

```

```

lblPerguntaNumeracao1.Visibility = Visibility.Visible;
lblPerguntaNumeracao2.Visibility = Visibility.Visible;
lblPerguntaNumeracao3.Visibility = Visibility.Visible;
lblPerguntaNumeracao4.Visibility = Visibility.Visible;
lblPerguntaNumeracao5.Visibility = Visibility.Visible;

// Repositioning the question numbering
lblPerguntaNumeracao1.Margin = new Thickness(12, 84, 0, 0);
lblPerguntaNumeracao2.Margin = new Thickness(12, 156, 0, 0);
lblPerguntaNumeracao3.Margin = new Thickness(12, 229, 0, 0);
lblPerguntaNumeracao4.Margin = new Thickness(12, 301, 0, 0);
lblPerguntaNumeracao5.Margin = new Thickness(12, 377, 0, 0);

// Repositioning the total number of people per question
lblPessoasTotalQuestao1.Margin = new Thickness(42, 84, 0, 0);
lblPessoasTotalQuestao2.Margin = new Thickness(42, 156, 0,
0);
lblPessoasTotalQuestao3.Margin = new Thickness(42, 229, 0,
0);
lblPessoasTotalQuestao4.Margin = new Thickness(42, 301, 0,
0);
lblPessoasTotalQuestao5.Margin = new Thickness(42, 377, 0,
0);

// Hiding selected answers
lblRespostaSelecionada1.Visibility = Visibility.Hidden;
lblRespostaSelecionada2.Visibility = Visibility.Hidden;
lblRespostaSelecionada3.Visibility = Visibility.Hidden;
lblRespostaSelecionada4.Visibility = Visibility.Hidden;
lblRespostaSelecionada5.Visibility = Visibility.Hidden;

// Hiding correct answers
lblRespostaCerta1.Visibility = Visibility.Hidden;
lblRespostaCerta2.Visibility = Visibility.Hidden;
lblRespostaCerta3.Visibility = Visibility.Hidden;
lblRespostaCerta4.Visibility = Visibility.Hidden;
lblRespostaCerta5.Visibility = Visibility.Hidden;

imgRC1.Visibility = Visibility.Hidden;
imgRC2.Visibility = Visibility.Hidden;
imgRC3.Visibility = Visibility.Hidden;
imgRC4.Visibility = Visibility.Hidden;
imgRC5.Visibility = Visibility.Hidden;

imgStatusPergunta1.Visibility = Visibility.Hidden;
imgStatusPergunta2.Visibility = Visibility.Hidden;
imgStatusPergunta3.Visibility = Visibility.Hidden;
imgStatusPergunta4.Visibility = Visibility.Hidden;
imgStatusPergunta5.Visibility = Visibility.Hidden;

lblEstatisticas.Visibility = Visibility.Visible;
lblEstatisticas.Margin = new Thickness(12, 9, 0, 0);

// Displaying the total number of people per question
lblPessoasTotalQuestao1.Visibility = Visibility.Visible;
lblPessoasTotalQuestao2.Visibility = Visibility.Visible;
lblPessoasTotalQuestao3.Visibility = Visibility.Visible;
lblPessoasTotalQuestao4.Visibility = Visibility.Visible;
lblPessoasTotalQuestao5.Visibility = Visibility.Visible;

imgAgradecimento.Visibility = Visibility.Hidden;

```

```
        break;

    case 3:
        imgAgradecimento.Visibility = Visibility.Visible;
        break;

    default:
        timer.Stop();

        // Reiniciar a variável de controle
        Estatico.RELATORIOPAGINA = 1;

        // Acessar a janela principal e limpar o conteúdo do Frame,
removendo a página
        var mainWindow = (MainWindow)Application.Current.MainWindow;
        mainWindow.MainFrame.Content = null;

        // Forçar a coleta de lixo para liberar recursos da página
        GC.Collect();
        GC.WaitForPendingFinalizers();
        break;
    }
}

e)
{
    Estatico.RELATORIOPAGINA++;
    ProximaPagina(Estatico.RELATORIOPAGINA);
}

private void imgAgradecimento_MouseDown(object sender,
MouseEventArgs e)
{
    Estatico.RELATORIOPAGINA++;
    ProximaPagina(Estatico.RELATORIOPAGINA);
}
}
```

APÊNDICE AZ – CODIGO TOTEM – CONTROLE – C#

```

using ExplorandoMarteComTecnologia_WPF.DTO;
using ExplorandoMarteComTecnologia_WPF.Modelos;
using ExplorandoMarteComTecnologia_WPF.Service;
using Newtonsoft.Json;

namespace ExplorandoMarteComTecnologia_WPF.Controllers
{
    internal class Controle
    {
        public int ConverterStringParaInt(string valor)
        {
            Validacao validacao = new Validacao();
            return validacao.ConverterStringParaInt(valor);
        }

        public void ValidarResposta(int idPergunta, string resposta)
        {
            Validacao validacao = new Validacao();
            switch (validacao.ValidarResposta(idPergunta, resposta))
            {
                case 0:
                    Estatico.questionarioRespostas.Add(new QuestionarioRespostasModel { Acertos = 0, Erros = 1 });
                    break;

                case 1:
                    Estatico.questionarioRespostas.Add(new QuestionarioRespostasModel { Acertos = 1, Erros = 0 });
                    break;
            }
        }

        public string MontarJsonQuestionarioAvaliacao(List<QuestionarioRespostasModel> questionarioRespostas, List<AvaliacaoRespostasDTO> avaliacaoRespostas, string sugestao)
        {
            List<QuestionarioRespostasDTO> questionarioDTO = new List<QuestionarioRespostasDTO>();
            for (int i = 0; i < questionarioRespostas.Count; i++)
            {
                if (i >= 5)
                {
                    return null;
                }
                string pergunta = IdentificarPergunta(i);
                questionarioDTO.Add(new QuestionarioRespostasDTO
                {
                    Id = i + 1,
                    Pergunta = pergunta,
                    Acertos = questionarioRespostas[i].Acertos,
                    Erros = questionarioRespostas[i].Erros
                });
            }

            if (string.IsNullOrEmpty(sugestao))
            {
                var requestSemSugestao = new QuestionarioAvaliacaoSemSugestao
                {
                    QuestionarioRespostas = questionarioDTO,

```

```

        AvaliacaoRespostas = avaliacaoRespostas
    };

    return JsonConvert.SerializeObject(requestSemSugestao,
Formatting.Indented);
}

var request = new QuestionarioAvaliacaoRequest
{
    QuestionarioRespostas = questionarioDTO,
    AvaliacaoRespostas = avaliacaoRespostas,
    AvaliacaoSugestao = string.IsNullOrEmpty(sugestao) ? null : new
AvaliacaoSugestao { Sugestao = sugestao }
};

return JsonConvert.SerializeObject(request, Formatting.Indented);
}

private string IdentificarPergunta(int id)
{
    switch (id)
    {
        case 0:
            return "VulcaoSistemaSolar";

        case 1:
            return "FundadorSpaceX";

        case 2:
            return "FenomenoVermelho";

        case 3:
            return "VidaPassada";

        case 4:
            return "MonteOlimpo";

        default:
            return "";
    }
}

public async void AtualizarValoresAcertosErrosPerguntas(string endpoint)
{
    ControllService controllService = new ControllService();
    List<QuestionarioRespostasDTO> questionarioRespostas = await
controllService.GetDataAsync(endpoint);

    if (Estatico.ERROMENSAGEM == "ERRO")
    {
        return;
    }

    Estatico.QUANTIDADEACERTOSQUEST1 = questionarioRespostas[0].Acertos;
    Estatico.QUANTIDADEERROSQUEST1 = questionarioRespostas[0].Erros;
    Estatico.QUANTIDADEACERTOSQUEST2 = questionarioRespostas[1].Acertos;
    Estatico.QUANTIDADEERROSQUEST2 = questionarioRespostas[1].Erros;
    Estatico.QUANTIDADEACERTOSQUEST3 = questionarioRespostas[2].Acertos;
    Estatico.QUANTIDADEERROSQUEST3 = questionarioRespostas[2].Erros;
    Estatico.QUANTIDADEACERTOSQUEST4 = questionarioRespostas[3].Acertos;
    Estatico.QUANTIDADEERROSQUEST4 = questionarioRespostas[3].Erros;
}

```

```

Estatico.QUANTIDADEACERTOSQUEST5 = questionarioRespostas[4].Acertos;
Estatico.QUANTIDADEERROSQUEST5 = questionarioRespostas[4].Erros;

}

public async Task ArmazenarRespostas(string endpoint, string json)
{
    ControllService controllService = new ControllService();
    if (endpoint != null && json != null)
    {
        await controllService.PutDataAsync(endpoint, json);
    }
}

#region Teclado
public string Teclado(int KeyID)
{
    switch (KeyID)
    {
        #region Letras Maiusculas
        case 1: return "A"; break;
        case 2: return "B"; break;
        case 3: return "C"; break;
        case 4: return "D"; break;
        case 5: return "E"; break;
        case 6: return "F"; break;
        case 7: return "G"; break;
        case 8: return "H"; break;
        case 9: return "I"; break;
        case 10: return "J"; break;
        case 11: return "K"; break;
        case 12: return "L"; break;
        case 13: return "M"; break;
        case 14: return "N"; break;
        case 15: return "O"; break;
        case 16: return "P"; break;
        case 17: return "Q"; break;
        case 18: return "R"; break;
        case 19: return "S"; break;
        case 20: return "T"; break;
        case 21: return "U"; break;
        case 22: return "V"; break;
        case 23: return "W"; break;
        case 24: return "X"; break;
        case 25: return "Y"; break;
        case 26: return "Z"; break;
        #endregion

        #region Letras Minusculas
        case 27: return "a"; break;
        case 28: return "b"; break;
        case 29: return "c"; break;
        case 30: return "d"; break;
        case 31: return "e"; break;
        case 32: return "f"; break;
        case 33: return "g"; break;
        case 34: return "h"; break;
        case 35: return "i"; break;
        case 36: return "j"; break;
        case 37: return "k"; break;
        case 38: return "l"; break;
        case 39: return "m"; break;
    }
}

```

```
        case 40: return "n"; break;
        case 41: return "o"; break;
        case 42: return "p"; break;
        case 43: return "q"; break;
        case 44: return "r"; break;
        case 45: return "s"; break;
        case 46: return "t"; break;
        case 47: return "u"; break;
        case 48: return "v"; break;
        case 49: return "w"; break;
        case 50: return "x"; break;
        case 51: return "y"; break;
        case 52: return "z"; break;
    #endregion

    #region Teclas Especiais
    case 53: return ","; break;
    case 54: return "."; break;
    case 55: return " "; break;
    case 56: return "Ç"; break;
    case 57: return "ç"; break;
    #endregion

    #region Numeros
    case 58: return "1"; break;
    case 59: return "2"; break;
    case 60: return "3"; break;
    case 61: return "4"; break;
    case 62: return "5"; break;
    case 63: return "6"; break;
    case 64: return "7"; break;
    case 65: return "8"; break;
    case 66: return "9"; break;
    case 67: return "0"; break;
    #endregion

        default: return " "; break;
    }

    return " ";
}
#endregion
}
```

APÊNDICE BA – CODIGO TOTEM – ESTATICO – C#

```

using ExplorandoMarteComTecnologia_WPF.DTO;
using ExplorandoMarteComTecnologia_WPF.Modelos;

namespace ExplorandoMarteComTecnologia_WPF.Controllers
{
    internal static class Estatico
    {

        public static string LINKAPI = "";
        public static string ERROMENSAGEM = "";

        public static int RELATORIOPAGINA = 1; //Gambiarra para resolver um bug

        public static List<QuestionarioRespostasModel> questionarioRespostas =
new List<QuestionarioRespostasModel>();
        public static List<AvaliacaoRespostasDTO> avaliacaoRespostas = new
List<AvaliacaoRespostasDTO>();
        public static string COMENTARIO = "";

        public static int QUANTIDADEACERTOSQUEST1 = 0;
        public static int QUANTIDADEACERTOSQUEST2 = 0;
        public static int QUANTIDADEACERTOSQUEST3 = 0;
        public static int QUANTIDADEACERTOSQUEST4 = 0;
        public static int QUANTIDADEACERTOSQUEST5 = 0;

        public static int QUANTIDADEERROSQUEST1 = 0;
        public static int QUANTIDADEERROSQUEST2 = 0;
        public static int QUANTIDADEERROSQUEST3 = 0;
        public static int QUANTIDADEERROSQUEST4 = 0;
        public static int QUANTIDADEERROSQUEST5 = 0;

        public static string TEMPRESPOSTAQUEST1 = "";
        public static string TEMPRESPOSTAQUEST2 = "";
        public static string TEMPRESPOSTAQUEST3 = "";
        public static string TEMPRESPOSTAQUEST4 = "";
        public static string TEMPRESPOSTAQUEST5 = "";

        public static string MENSAGEMCOMUNICACAO = "";

    }
}

```

APÊNDICE BB – CODIGO TOTEM – VALIDACAO – C#

```

namespace ExplorandoMarteComTecnologia_WPF.Controllers
{
    internal class Validacao
    {

        public int ConverterStringParaInt(string valor)
        {
            try
            {
                return Convert.ToInt32(valor);
            }
            catch (Exception)
            {
                return 0;
            }
        }

        public int ValidarResposta(int idPergunta, string Resposta)
        {

            switch (idPergunta)
            {

                case 1:
                    Estatico.TEMPRESPOSTAQUEST1 = Resposta;
                    if (Resposta.Equals("D) Monte Olimpo, com 27 km de altura"))
                    {
                        return 1;
                    }
                    break;

                case 2:
                    Estatico.TEMPRESPOSTAQUEST2 = Resposta;
                    if (Resposta.Equals("C) Elon Musk"))
                    {
                        return 1;
                    }
                    break;

                case 3:
                    Estatico.TEMPRESPOSTAQUEST3 = Resposta;
                    if (Resposta.Equals("B) A presença de óxido de ferro em sua
superfície"))
                    {
                        return 1;
                    }
                    break;

                case 4:
                    Estatico.TEMPRESPOSTAQUEST4 = Resposta;
                    if (Resposta.Equals("A) Evidências de água líquida em rios e
lagos secos."))
                    {
                        return 1;
                    }
                    break;

                case 5:
                    Estatico.TEMPRESPOSTAQUEST5 = Resposta;

```

```
        if (Resposta.Equals("C) Missões robóticas para mapear a
superfície."))
{
    return 1;
}
break;

default:
    break;
}

return 0;
}
```

APÊNDICE BC – CODIGO TOTEM – CONTROLLSERVICE – C#

```

using ExplorandoMarteComTecnologia_WPF.Controllers;
using ExplorandoMarteComTecnologia_WPF.DTO;
using System.Net.Http;
using System.Net.Http.Json;
using System.Text;
using System.Windows;

namespace ExplorandoMarteComTecnologia_WPF.Service
{
    internal class ControllService
    {

        private readonly HttpClient _httpClient;

        public ControllService()
        {
            _httpClient = new HttpClient();
        }

        public async Task<List<QuestionarioRespostasDTO>>
GetDataAsync(string endpoint)
        {

            if (String.IsNullOrEmpty(Estatico.LINKAPI))
            {
                MessageBox.Show($"API não configurada\nPedir ajuda de um
funcionario!");
                return new List<QuestionarioRespostasDTO>();
            }

            try
            {
                var response = await
.httpClient.GetAsync($"{Estatico.LINKAPI}{endpoint}");
                if (response.IsSuccessStatusCode)
                {
                    // Lê e desserializa o conteúdo JSON para uma lista de
QuestionarioRespostasDTO
                    var content = await
response.Content.ReadFromJsonAsync<List<QuestionarioRespostasDTO>>();

                    // Retorna a lista de QuestionarioRespostasDTO
                    return content ?? new List<QuestionarioRespostasDTO>();
                }
                // Retorna uma lista vazia se o conteúdo for nulo
                else
                {
                    MessageBox.Show($"Erro: {response.StatusCode} -
{response.ReasonPhrase}");
                    return new List<QuestionarioRespostasDTO>(); // Retorna
uma lista vazia em caso de erro
                }
            }
            catch (Exception e)
            {
                MessageBox.Show($"Erro ao obter dados: {e.Message}\nChame
por um funcionario!");
                Estatico.ERROMENSAGEM = "ERRO";
                return new List<QuestionarioRespostasDTO>(); // Retorna uma
lista vazia em caso de exceção
            }
        }
    }
}

```

```
        }

    public async Task<string> PutDataAsync(string endpoint, string
jsonData)
{
    // O jsonData já está em formato JSON, então usamos diretamente
    var content = new StringContent(jsonData, Encoding.UTF8,
"application/json");
    var mensagem = "";

    try
    {
        var response = await
.httpClient.PostAsync($"'{Estatico.LINKAPI}{endpoint}", content);
        if (response.IsSuccessStatusCode)
        {
            mensagem = await response.Content.ReadAsStringAsync();
        }
        else
        {
            mensagem = $"Erro: {response.StatusCode} -
{response.ReasonPhrase}";
        }
    }
    catch (Exception e)
    {
        Estatico.ERROMENSAGEM = "ERRO";
        return $"Erro ao atualizar dados: {e.Message}\nChame por um
funcionario!";
    }

    return mensagem;
}

}
```

APÊNDICE BD – CODIGO TOTEM – QUESTIONARIORESPOSTASMODEL – C#

```
namespace ExplorandoMarteComTecnologia_WPF.Modelos
{
    internal class QuestionarioRespostasModel
    {
        public int Acertos { get; set; }
        public int Erros { get; set; }
    }
}
```

APÊNDICE BE – CODIGO TOTEM – AVALIACAORESPOSTASDTO – C#

```
namespace ExplorandoMarteComTecnologia_WPF.DTO
{
    internal class AvaliacaoRespostasDTO
    {
        public int Id { get; set; }
        public string Pergunta { get; set; }
        public int Excelente { get; set; }
        public int Bom { get; set; }
        public int Regular { get; set; }
        public int Ruim { get; set; }
        public int Pessimo { get; set; }
    }
}
```

APÊNDICE BF – CODIGO TOTEM – AVALIACAO SUGESTAO DTO – C#

```
namespace ExplorandoMarteComTecnologia_WPF.DTO
{
    internal class AvaliacaoSugestao
    {
        public string Sugestao { get; set; }
    }
}
```

APÊNDICE BG – CODIGO TOTEM – QUESTIONARIOAVALIACAOREQUESTDTO – C#

```
namespace ExplorandoMarteComTecnologia_WPF.DTO
{
    internal class QuestionarioAvaliacaoRequest
    {
        public List<QuestionarioRespostasDTO> QuestionarioRespostas { get; set; }
        public List<AvaliacaoRespostasDTO> AvaliacaoRespostas { get; set; }
        public AvaliacaoSugestao? AvaliacaoSugestao { get; set; }
    }
}
```

APÊNDICE BH – CODIGO TOTEM -
QUESTIONARIOAVALIACAOSEMSUGESTAODTO – C#

```
namespace ExplorandoMarteComTecnologia_WPF.DTO
{
    internal class QuestionarioAvaliacaoSemSugestao
    {
        public List<QuestionarioRespostasDTO> QuestionarioRespostas { get; set; }
        public List<AvaliacaoRespostasDTO> AvaliacaoRespostas { get; set; }
    }
}
```

APÊNDICE BI – CODIGO TOTEM – QUESTIONARIORESPOSTASDTO – C#

```
namespace ExplorandoMarteComTecnologia_WPF.DTO
{
    internal class QuestionarioRespostasDTO
    {
        public int Id { get; set; }
        public string Pergunta { get; set; }
        public int Acertos { get; set; }
        public int Erros { get; set; }
    }
}
```

APÊNDICE BJ – CODIGO SITE – APICONFIG – JAVASCRIPT

```
document.addEventListener("DOMContentLoaded", function () {
    localStorage.setItem("apiBase", "https://8543-2804-14d-32d4-8cbd-e912-5ba4-
6276-376e.ngrok-free.app");
});
```

APÊNDICE BK – CODIGO SITE – AUTH – JAVASCRIPT

```

// Inicializa a variável Logged com o valor armazenado no localStorage (ou false
// se não existir)
let Logged = localStorage.getItem("Logged") === "true";

// Adiciona eventos para os links do menu
document.querySelectorAll(".nav-bar a").forEach(link => {
    link.addEventListener("click" || "tap", function (event) {
        const href = link.getAttribute("href");

        // Páginas liberadas para todos os usuários
        const paginasLiberadas = ["login.html", "home.html", "informações.html",
        "compras.html"];

        // Verifica se o usuário está tentando acessar uma página restrita
        if (!Logged && !paginasLiberadas.includes(href)) {
            event.preventDefault(); // Impede o comportamento padrão do link
            alert("Você precisa estar logado para acessar esta página."); //
Mensagem para o usuário
            window.location.href = "login.html"; // Redireciona para a página de
login
        }
    });
});

document.addEventListener("DOMContentLoaded", function () {verificarLogin();});

function verificarLogin() {
    const loginBtn = document.getElementById("loginBtn"); // Botão de login
    const logoutBtn = document.getElementById("logoutBtn"); // Botão de logout

    if (Logged) {
        if (loginBtn) loginBtn.style.display = "none"; // Esconde o botão de
login
        if (logoutBtn) logoutBtn.style.display = "inline"; // Exibe o botão de
logout
    } else {
        if (loginBtn) loginBtn.style.display = "inline"; // Exibe o botão de
login
        if (logoutBtn) logoutBtn.style.display = "none"; // Esconde o botão de
logout
    }
}

function fazerLogout() {
    localStorage.setItem("Logged", "false"); // Define o status de login como
falso
    alert("Você foi deslogado com sucesso!");
    window.location.href = "home.html";
}

```

APÊNDICE BL – CODIGO SITE – HOME – HTML

```

<!DOCTYPE html>
<html lang="pt-br">

<head>
    <!-- Meta tag que define o conjunto de caracteres como UTF-8, garantindo que
caracteres especiais sejam exibidos corretamente -->
    <meta charset="UTF-8">

    <!-- Meta tag para garantir que a página seja renderizada corretamente em
versões antigas do Internet Explorer -->
    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <!-- Meta tag que faz a página ser responsiva em dispositivos móveis,
ajustando o layout conforme a largura da tela -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <!-- Título da página, que aparece na aba do navegador -->
    <title>Museu Planeta Vermelho</title>

    <!-- Link para a biblioteca de ícones Font Awesome (para adicionar ícones de
redes sociais, entre outros) -->
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.4/css/all.min.css">

    <!-- Link para o arquivo CSS que define o estilo da página -->
    <link rel="stylesheet" href="Pim/Home/home.css">
</head>

<body>

    <header>
        <nav class="nav-bar">
            <div class="logo">
                <h1>MECH</h1>
            </div>
            <div class="nav-list">
                <ul>
                    <li class="nav-item"><a href="home.html" class="nav-
link">Página Principal</a></li>
                    <li class="nav-item"><a href="informações.html" class="nav-
link">Informações</a></li>
                    <li class="nav-item"><a href="exposição.html" class="nav-
link"> Exposições</a></li>
                    <li class="nav-item"><a href="compras.html" class="nav-link">
Compras</a></li>
                    <li class="nav-item"><a href="questionario.html" class="nav-
link"> Questionario/Avaliação</a></li>
                

```

```

        </div>
    </nav>
    <div class="mobile-menu">
        <ul>
            <li class="nav-item"><a href="home.html" class="nav-link">Página Principal</a></li>
            <li class="nav-item"><a href="informações.html" class="nav-link">Informações</a></li>
                <li class="nav-item"><a href="exposição.html" class="nav-link">Exposições</a></li>
                <li class="nav-item"><a href="compras.html" class="nav-link">Compras</a></li>
                <li class="nav-item"><a href="questionario.html" class="nav-link"> Questionario/Avaliação</a></li>
        </ul>

        <div class="login-button">
            <button id="loginBtn" class="login"><a href="login.html" id="loginLink">Entrar</a></button>
            <button id="logoutBtn" class="logout" onclick="fazerLogout()">Logout</button>
        </div>
    </header>

    <!-- Seção principal do site, contendo o conteúdo sobre o Museu -->
    <section class="home" id="home">

        <!-- Div que contém o texto principal sobre o Museu -->
        <div class="content">
            <!-- Título principal da seção -->
            <h3>MECH</h3>

            <!-- Parágrafo explicativo sobre o Museu -->
            <p>
                O Museu da Evolução e Conhecimento Humano (MECH) é uma instituição futurista dedicada à história e exploração de Marte. Localizado em uma cúpula transparente com vistas deslumbrantes das paisagens marcianas, o museu oferece exposições interativas sobre as missões espaciais que desbravaram o planeta, artefatos dos primeiros assentamentos humanos, e simulações imersivas da vida em Marte. Combinando ciência, tecnologia e educação, o museu inspira visitantes a imaginar o futuro da colonização espacial e o papel de Marte como um novo lar para a humanidade.
            </p>
        </div>

    </section>
    <script src="auth.js"></script>
    <script src="https://kit.fontawesome.com/16ffb79ef8.js" crossorigin="anonymous"></script>
    <script src="https://hammerjs.github.io/dist/hammer.min.js"></script>
    <script src="Pim/Home/home.js"></script>
</body>

</html>

```

APÊNDICE BM – CODIGO SITE– HOME – CSS

```

* {
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    outline: none;
    border: none;
    text-decoration: none;
    text-transform: capitalize;
    transition: .2s linear;
}

body {
    height: 100vh;
    overflow: hidden;
    background-size: cover;
}

/* Começo-Header */
header {
    background-color: #444;
}

.nav-bar {
    display: flex;
    justify-content: space-between;
    padding: 1.5rem 6rem;
}

.logo {
    display: flex;
    align-items: center;
}

.logo h1 {
    color: #fff;
}

.nav-list {
    display: flex;
    align-items: center;
}

.nav-list ul {
    display: flex;
    justify-content: center;
    list-style: none;
}

.nav-item {
    margin: 0 35px;
}

.nav-link {
    text-decoration: none;
    font-size: 1.15rem;
    color: #fff;
    font-weight: 400;
}

```

```
.login-button button {
    border: none;
    padding: 10px 15px;
    border-radius: 5px;
    background-color: #970000;
}

.login-button button a {
    text-decoration: none;
    color: #fff;
    font-weight: 500;
    font-size: 1.1rem;
}

.login-button button.login {
    background-color: #970000; /* Cor de fundo do botão de login */
    color: #fff;
    border: none;
    padding: 10px 20px;
    cursor: pointer;
}

/* Estilos para o botão de logout */
.login-button button.logout {
    color: #fff;
    border: 1px solid #970000;
    padding: 10px 20px;
    cursor: pointer;
    display: none; /* Esconde o botão de logout por padrão */
}

.login-button button:empty {
    display: none;
}

.mobile-menu-icon {
    display: none;
}

.mobile-menu {
    display: none; /* Inicialmente o menu não está visível */
}

/* Fim-header */

.home {
    padding: 40px;
    background-image: url(src/647675.jpg);
    min-height: 110vh;
    display: flex;
    align-items: center;
    background-size: cover;
    background-position: center;
}

.home .content {
    height: 560px;
    max-width: 820px;
}

.home .content h3 {
    font-size: 50px;
    text-transform: uppercase;
```

```
    color: #fff;
}

.home .content p {
    font-size: 20px;
    font-weight: lighter;
    line-height: 1.8;
    padding: 1rem 0;
    color: #eee;
}

@media screen and (max-width: 440px) {
    body {
        overflow-y: scroll;
    }

    .nav-bar {
        padding: 1.5rem 4rem;
    }

    .nav-item {
        display: none;
    }

    .login-button {
        display: none;
    }

    .mobile-menu-icon {
        display: block;
    }

    .mobile-menu-icon button {
        background-color: transparent;
        border: none;
        cursor: pointer;
    }

    .mobile-menu ul {
        display: flex;
        flex-direction: column;
        text-align: center;
        padding-bottom: 1rem;
    }

/* Ajustes para os itens do menu */
    .mobile-menu .nav-item {
        display: block;
        padding-top: 1.2rem;
    }

    .mobile-menu .login-button {
        display: block;
        padding: 1rem 2rem;
    }

    .mobile-menu .login-button button {
        width: 100%;
    }

/* Quando o menu está aberto, ele deve ser exibido */
    .open {
        display: block;
    }
}
```

```
}

/* Ajuste da transição para animar a altura ao abrir/fechar o menu */
.mobile-menu {
    overflow: hidden; /* Impede o conteúdo de transbordar */
    height: 0; /* Inicialmente o menu tem altura 0 (fechado) */
    transition: height 0.3s ease-in-out; /* Transição suave para abrir e fechar */
}

/* Quando o menu estiver aberto, ele deve ter a altura necessária */
.mobile-menu.open {
    height: auto; /* A altura do menu será automaticamente ajustada */
}

.home {
    scroll-behavior: auto;
    width: 100%;
    height: 750px;
    text-align: center;
    justify-items: center;
}

.home .content h3 {
    font-size: 22px;
}

.home .content p {
    font-size: 18px;
    color: #eee;
}
}

/* Ajustes para telas menores (max-width: 375px) */
@media screen and (max-width: 375px) {
    body {
        overflow-y: scroll;
    }
    .nav-bar {
        padding: 1.5rem 4rem;
    }
    .nav-item {
        display: none;
    }
    .login-button {
        display: none;
    }
    .mobile-menu-icon {
        display: block;
    }
    .mobile-menu-icon button {
        background-color: transparent;
        border: none;
        cursor: pointer;
    }
    .mobile-menu ul {
        display: flex;
        flex-direction: column;
        text-align: center;
        padding-bottom: 1rem;
    }
    .mobile-menu .nav-item {
        display: block;
    }
}
```

```
        padding-top: 1.2rem;
    }
.mobile-menu .login-button {
    display: block;
    padding: 1rem 2rem;
}
.mobile-menu .login-button button {
    width: 100%;
}
.open {
    display: block;
}
.home {
    scroll-behavior: auto;
    width: 100%;
    height: 850px;
    text-align: center;
    justify-items: center;
}
.home .content h3 {
    font-size: 22px;
}
.home .content p {
    font-size: 18px;
    color: #eee;
}
}
/* Ajustes para telas ainda menores (max-width: 360px) */
@media screen and (max-width: 360px) {
    body {
        overflow-y: scroll;
    }
    .nav-bar {
        padding: 1.5rem 4rem;
    }
    .nav-item {
        display: none;
    }
    .login-button {
        display: none;
    }
    .mobile-menu-icon {
        display: block;
    }
    .mobile-menu-icon button {
        background-color: transparent;
        border: none;
        cursor: pointer;
    }
    .mobile-menu ul {
        display: flex;
        flex-direction: column;
        text-align: center;
        padding-bottom: 1rem;
    }
    .mobile-menu .nav-item {
        display: block;
        padding-top: 1.2rem;
    }
    .mobile-menu .login-button {
        display: block;
        padding: 1rem 2rem;
    }
}
```

```
}

.mobile-menu .login-button button {
    width: 100%;
}
.open {
    display: block;
}
.home {
    scroll-behavior: auto;
    width: 100%;
    height: 950px;
    text-align: center;
    justify-items: center;
}
.home .content h3 {
    font-size: 22px;
}
.home .content p {
    font-size: 18px;
    color: #eee;
}
}
```

APÊNDICE BN – CODIGO SITE – HOME – JAVASCRIPT

```
// Função para mostrar/ocultar o menu
function menuShow() {
    let menuMobile = document.querySelector('.mobile-menu');
    let menuIcon = document.querySelector('.icon');

    // Verifica se o menu está aberto ou fechado e alterna o estado
    if (menuMobile.classList.contains('open')) {
        menuMobile.classList.remove('open'); // Fecha o menu
        menuIcon.src = "assets/img/menu_white_36dp.svg"; // Troca o ícone para
        "abrir"

        // Resetando a altura para 0 após o fechamento
        setTimeout(() => {
            menuMobile.style.height = '0'; // A altura do menu é definida como 0
            // Atualiza a página após o fechamento do menu
            window.location.reload(); // Recarrega a página
        }, 300); // Aguardar a transição de fechamento antes de resetar a altura
    } else {
        menuMobile.classList.add('open'); // Abre o menu
        menuIcon.src = "assets/img/close_white_36dp.svg"; // Troca o ícone para
        "fechar"

        // Ajusta a altura para auto para que o menu ocupe o espaço necessário
        menuMobile.style.height = 'auto';
    }
}

// Função para fechar o menu ao clicar fora (e resetar)
window.addEventListener('click', function (event) {
    let menuMobile = document.querySelector('.mobile-menu');
    let menuIconButton = document.querySelector('.mobile-menu-icon button');

    // Verifica se o clique foi fora do menu e do ícone do menu
    if (!menuMobile.contains(event.target) &&
        !menuIconButton.contains(event.target)) {
        if (menuMobile.classList.contains('open')) {
            menuMobile.classList.remove('open'); // Fecha o menu
            document.querySelector('.icon').src =
            "assets/img/menu_white_36dp.svg"; // Troca o ícone para "abrir"

            // Resetando a altura para 0 após o fechamento
            setTimeout(() => {
                menuMobile.style.height = '0'; // A altura do menu é definida
                como 0
                // Atualiza a página após o fechamento do menu
                window.location.reload(); // Recarrega a página
            }, 300); // Aguardar a transição de fechamento antes de resetar a
            altura
        }
    }
});

// Impede que o clique no ícone do menu feche o menu ao clicar nele
document.querySelector('.mobile-menu-icon button').addEventListener('click',
function(event) {
    event.stopPropagation(); // Impede que o clique no ícone seja tratado como
    um clique fora do menu
});
```

APÊNDICE BO – CODIGO SITE – INFORMACOES – HTML

```

<!DOCTYPE html>
<html lang="pt-br">

<head>
    <!-- Declaração do tipo de documento HTML5 -->
    <meta charset="UTF-8">
    <!-- Define o conjunto de caracteres como UTF-8 para garantir que o texto
        seja exibido corretamente -->
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <!-- Define o modo de renderização da página para navegadores antigos -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- Configura a página para ser responsiva em dispositivos móveis -->
    <title>Informações</title>
    <!-- Título da página -->

    <!-- Link para a biblioteca de ícones Font Awesome -->
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
    awesome/5.15.4/css/all.min.css">

    <!-- Link para o arquivo CSS que estiliza a página de informações -->
    <link rel="stylesheet" href="Pim/Home/Informações/informações.css">
</head>

<body>

    <!-- Cabeçalho da página -->
    <header>
        <nav class="nav-bar">
            <div class="logo">
                <h1>MECH</h1>
            </div>
            <div class="nav-list">
                <ul>
                    <li class="nav-item"><a href="home.html" class="nav-
                    link">Página Principal</a></li>
                    <li class="nav-item"><a href="informações.html" class="nav-
                    link">Informações</a></li>
                    <li class="nav-item"><a href="exposição.html" class="nav-
                    link"> Exposições</a></li>
                    <li class="nav-item"><a href="compras.html" class="nav-link">
                    Compras</a></li>
                    <li class="nav-item"><a href="questionario.html" class="nav-
                    link"> Questionario/Avaliação</a></li>
                </ul>
            </div>
            <div class="login-button">
                <button id="loginBtn" class="login"><a href="login.html"
                id="loginLink">Entrar</a></button>
                <button id="logoutBtn" class="logout"
                onclick="fazerLogout()">Logout</button>
            </div>
        </nav>
        <div class="mobile-menu">
            <ul>

```

```

        <li class="nav-item"><a href="home.html" class="nav-link">Página
Principal</a></li>
            <li class="nav-item"><a href="informações.html" class="nav-
link">Informações</a></li>
                <li class="nav-item"><a href="exposição.html" class="nav-link">
Exposições</a></li>
                <li class="nav-item"><a href="compras.html" class="nav-link">
Compras</a></li>
                <li class="nav-item"><a href="questionario.html" class="nav-
link"> Questionario/Avaliação</a></li>
            </ul>

        <div class="login-button">
            <button id="loginBtn" class="login"><a href="login.html"
id="loginLink">Entrar</a></button>
            <button id="logoutBtn" class="logout"
onclick="fazerLogout()">Logout</button>
        </div>
    </div>
</header>


<section class="sobre">
    <div class="interface">
        <!-- Flexbox para dispor as imagens e texto --&gt;
        &lt;div class="flex"&gt;

            <!-- Imagem do local com um mapa incorporado via iframe --&gt;
            &lt;div class="img-sobre"&gt;
                &lt;iframe
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3656.8761781208523!2d-
46.67863462466945!3d-
23.572890178791923!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x94ce5764ba
70b965%3A0x94b834e23de5d4c1!2sMuseu%20da%20Imagem%20e%20do%20Som%20(MIS)!5e0!3m2!
1spt-BR!2sbr!4v1728433400114!5m2!1spt-BR!2sbr"
width="400" height="250" style="border:0;" allowfullscreen="" loading="lazy" referrerPolicy="no-referrer-when-
downgrade"&gt;&lt;/iframe&gt;
            &lt;/div&gt;

            <!-- Texto explicativo sobre o museu --&gt;
            &lt;div class="txt-sobre"&gt;
                &lt;h2&gt;&lt;span&gt;Descubra os mistérios de Marte,&lt;/span&gt; no coração
de São Paulo!&lt;/h2&gt;
                &lt;p&gt;O Museu Planeta Vermelho te convida a uma jornada
inesquecível pelo universo!
                    &lt;br&gt;Localizado no charmoso bairro do Jardim Europa, em
pleno centro da cidade, nosso museu oferece
                    &lt;br&gt; fácil acesso para visitantes de todas as partes.
                    Mergulhe em uma experiência única e aprenda &lt;br&gt; tudo sobre o Planeta Vermelho,
desde sua formação até as mais recentes descobertas científicas.&lt;/p&gt;
                &lt;p&gt;Av. Europa, 158 – Jardim Europa, São Paulo – SP,
01449-000 &lt;br&gt; Telefone: 3532-8596 &lt;br&gt; &lt;/p&gt;
                &lt;!-- Botões de redes sociais --&gt;
                &lt;div class="btn-social"&gt;
                    &lt;a href="#"&gt;&lt;button class="icons"&gt; &lt;i class="fa-brands
fa-instagram"&gt;&lt;/i&gt;&lt;/button&gt;&lt;/a&gt;
                    &lt;!-- Ícone do Instagram --&gt;
                    &lt;a href="#"&gt;&lt;button&gt;&lt;i class="fa-brands fa-
facebook"&gt;&lt;/i&gt;&lt;/button&gt;&lt;/a&gt;
                    &lt;!-- Ícone do YouTube --&gt;
                    &lt;a href="#"&gt;&lt;button&gt;&lt;i class="fa-brands fa-
linkedin"&gt;&lt;/i&gt;&lt;/button&gt;&lt;/a&gt;
                &lt;/div&gt;
            &lt;/div&gt;
        &lt;/div&gt;
    &lt;/div&gt;
&lt;/section&gt;
</pre>

```

```
        <!-- Ícone do LinkedIn -->
        </div>
    </div>
</section>
<script src="auth.js"></script>
<script src="Pim/Home/Informações/informações.js"></script>
<script src="https://kit.fontawesome.com/16ffb79ef8.js"
crossorigin="anonymous"></script>
</body>

</html>
```

APÊNDICE BP – CODIGO SITE – INFORMACOES – CSS

```
body {
    height: 100vh;
    overflow: hidden;
    background-size: cover;
    background-image: url(src/193.jpg);
}
/* Começo-Header */

* {
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    outline: none;
    border: none;
    text-decoration: none;
    text-transform: capitalize;
    transition: .2s linear;
}

/* Começo-Header */
header {
    background-color: #444;
}

.nav-bar {
    display: flex;
    justify-content: space-between;
    padding: 1.5rem 6rem;
}

.logo {
    display: flex;
    align-items: center;
}

.logo h1 {
    color: #fff;
}

.nav-list {
    display: flex;
    align-items: center;
}

.nav-list ul {
    display: flex;
    justify-content: center;
    list-style: none;
}

.nav-item {
    margin: 0 35px;
}
```

```
.nav-link {
    text-decoration: none;
    font-size: 1.15rem;
    color: #fff;
    font-weight: 400;
}

.login-button button {
    border: none;
    padding: 10px 15px;
    border-radius: 5px;
    background-color: #970000;
}

.login-button button a {
    text-decoration: none;
    color: #fff;
    font-weight: 500;
    font-size: 1.1rem;
}

.login-button button.login {
    background-color: #970000; /* Cor de fundo do botão de
login */
    color: #fff;
    border: none;
    padding: 10px 20px;
    cursor: pointer;
}

/* Estilos para o botão de logout */
.login-button button.logout {
    color: #fff;
    border: 1px solid #970000;
    padding: 10px 20px;
    cursor: pointer;
    display: none; /* Esconde o botão de logout por padrão */
}

.login-button button:empty {
    display: none;
}

.mobile-menu-icon {
    display: none;
}

.mobile-menu {
    display: none;
}

.interface {
    align-items: center;
    max-width: 1300px;
    padding: 150px;
    margin: 0 auto;
}

.mapa {
    position: relative;
    top: 100px;
}
```

```
.flex {
    display: flex;
}

section.sobre {
    padding: 85px 4%;
}

section.sobre .flex {
    align-items: center;
    gap: 55px;
}

.sobre .txt-sobre {
    color: #fff;
}

.sobre .txt-sobre h2 {
    font-size: 40px;
    line-height: 40px;
    margin-bottom: 30px;
}

.sobre .txt-sobre h2 span {
    font-size: 40px;
    color: #ffffff;
    display: block;
}

.sobre .txt-sobre p {
    font-size: 20px;
    margin: 25px 0;
    text-align: justify;
}

.btn-social button {
    width: 60px;
    height: 60px;
    border-radius: 50%;
    border: none;
    background-color: #ffffff;
    font-size: 22px;
    cursor: pointer;
    margin: 0 5px;
}

@media screen and (max-width: 440px) {
    body {
        overflow-y: scroll;
    }
    .nav-bar {
        padding: 1.5rem 4rem;
    }
    .nav-item {
        display: none;
    }
    .login-button {
        display: none;
    }
    .mobile-menu-icon {
        display: block;
    }
}
```

```
.mobile-menu-icon button {
    background-color: transparent;
    border: none;
    cursor: pointer;
}
.mobile-menu ul {
    display: flex;
    flex-direction: column;
    text-align: center;
    padding-bottom: 1rem;
}
.mobile-menu .nav-item {
    display: block;
    padding-top: 1.2rem;
}
.mobile-menu .login-button {
    display: block;
    padding: 1rem 2rem;
}
.mobile-menu .login-button button {
    width: 100%;
}
.open {
    display: block;
}
.home {
    scroll-behavior: auto;
    width: 100%;
    height: 750px;
    text-align: center;
    justify-items: center;
}
.home .content h3 {
    font-size: 22px;
}
.home .content p {
    font-size: 18px;
    color: #eee;
}
.flex {
    scroll-behavior: auto;
    flex-direction: column-reverse;
    width: 100%;
    height: 730px;
    text-align: center;
}
.sobre .txt-sobre h2 {
    font-size: 25px;
    line-height: 40px;
    margin-bottom: 30px;
    width: 350px;
}
.sobre .txt-sobre h2 span {
    font-size: 25px;
    color: #ffffff;
    display: block;
}
.sobre .txt-sobre h2 {
    font-size: 30px;
    line-height: 40px;
    margin-bottom: 30px;
}
.sobre .txt-sobre h2 span {
```

```
        font-size: 30px;
    }
.sobre .txt-sobre p {
    font-size: 18px;
}
}
```

APÊNDICE BQ – CODIGO SITE – INFORMACOES – JAVASCRIPT

```

// Função para mostrar/ocultar o menu
function menuShow() {
    let menuMobile = document.querySelector('.mobile-menu');
    let menuIcon = document.querySelector('.icon');

    // Verifica se o menu está aberto ou fechado e alterna o estado
    if (menuMobile.classList.contains('open')) {
        menuMobile.classList.remove('open'); // Fecha o menu
        menuIcon.src = "assets/img/menu_white_36dp.svg"; // Troca o ícone para
        "abrir"

        // Resetando a altura para 0 após o fechamento
        setTimeout(() => {
            menuMobile.style.height = '0'; // A altura do menu é definida como 0
            // Atualiza a página após o fechamento do menu
            window.location.reload(); // Recarrega a página
        }, 300); // Aguardar a transição de fechamento antes de resetar a altura
    } else {
        menuMobile.classList.add('open'); // Abre o menu
        menuIcon.src = "assets/img/close_white_36dp.svg"; // Troca o ícone para
        "fechar"

        // Ajusta a altura para auto para que o menu ocupe o espaço necessário
        menuMobile.style.height = 'auto';
    }
}

// Função para fechar o menu ao clicar fora (e resetar)
window.addEventListener('click', function (event) {
    let menuMobile = document.querySelector('.mobile-menu');
    let menuIconButton = document.querySelector('.mobile-menu-icon button');

    // Verifica se o clique foi fora do menu e do ícone do menu
    if (!menuMobile.contains(event.target) &&
        !menuIconButton.contains(event.target)) {
        if (menuMobile.classList.contains('open')) {
            menuMobile.classList.remove('open'); // Fecha o menu
            document.querySelector('.icon').src =
            "assets/img/menu_white_36dp.svg"; // Troca o ícone para "abrir"

            // Resetando a altura para 0 após o fechamento
            setTimeout(() => {
                menuMobile.style.height = '0'; // A altura do menu é definida
                como 0
                // Atualiza a página após o fechamento do menu
                window.location.reload(); // Recarrega a página
            }, 300); // Aguardar a transição de fechamento antes de resetar a
            altura
        }
    }
});

// Impede que o clique no ícone do menu feche o menu ao clicar nele
document.querySelector('.mobile-menu-icon button').addEventListener('click',
function(event) {
    event.stopPropagation(); // Impede que o clique no ícone seja tratado como
    um clique fora do menu
});

```

APÊNDICE BR – CODIGO SITE – COMPRAS – HTML

```

<!DOCTYPE html>
<html lang="pt-br">

<head>
    <!-- Seção de metadados e links para recursos externos. -->
    <meta charset="UTF-8">
    <!-- Define o conjunto de caracteres como UTF-8 para suportar a maioria dos
        caracteres de línguas escritas. -->

    <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <!-- Define a escala da página para telas de dispositivos móveis, ajustando o
            layout para caber corretamente. -->

    <title>Compras</title>
        <!-- Título da página que aparecerá na aba do navegador. -->
    <script src="apiConfig.js"></script>

    <link rel="stylesheet" href="Pim/Home/Compras/compras.css">
        <!-- Link para um arquivo CSS local que contém as regras de estilo para a
            página. -->

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
        awesome/5.15.4/css/all.min.css">
        <!-- Link para uma folha de estilos externa que contém ícones da biblioteca
            Font Awesome, utilizada para exibir ícones, como carrinho de compras. -->
</head>

<body>
    <!-- Início do corpo da página, onde o conteúdo visível será renderizado. -->

    <header>
        <nav class="nav-bar">
            <div class="logo">
                <h1>MECH</h1>
            </div>
            <div class="nav-list">
                <ul>
                    <li class="nav-item"><a href="home.html" class="nav-
                        link">Página Principal</a></li>
                    <li class="nav-item"><a href="informações.html" class="nav-
                        link">Informações</a></li>
                    <li class="nav-item"><a href="exposição.html" class="nav-
                        link"> Exposições</a></li>
                    <li class="nav-item"><a href="compras.html" class="nav-link">
                        Compras</a></li>
                    <li class="nav-item"><a href="questionario.html" class="nav-
                        link"> Questionario/Avaliação</a></li>
                </ul>
            </div>
            <div class="login-button">
                <button id="loginBtn" class="login"><a href="login.html"
                    id="loginLink">Entrar</a></button>
                <button id="logoutBtn" class="logout"
                    onclick="fazerLogout()">Logout</button>
            </div>

            <div class="mobile-menu-icon">
                <button onclick="menuShow()"></button>
            </div>
        </nav>
    </header>

```

```

        </div>
    </nav>
    <div class="mobile-menu">
        <ul>
            <li class="nav-item"><a href="home.html" class="nav-link">Página Principal</a></li>
            <li class="nav-item"><a href="informações.html" class="nav-link">Informações</a></li>
            <li class="nav-item"><a href="exposição.html" class="nav-link">Exposições</a></li>
            <li class="nav-item"><a href="compras.html" class="nav-link">Compras</a></li>
            <li class="nav-item"><a href="questionario.html" class="nav-link">Questionário/Avaliação</a></li>
        </ul>

        <div class="login-button">
            <button id="loginBtn" class="login"><a href="login.html" id="loginLink">Entrar</a></button>
            <button id="logoutBtn" class="logout" onclick="fazerLogout()">Logout</button>
        </div>
    </header>
    <main class="container">
        <!-- Container principal da página, que engloba o conteúdo central. -->

        <section class="ui">
            <!-- Seção de conteúdo com interface de usuário (UI) relacionada às compras. -->

            <div class="container-left">
                <!-- Div para agrupar o conteúdo à esquerda da página. -->
                <h1>Nova Compra</h1>
                <!-- Título principal da seção de compras. -->

                <form id="apiForm">
                    <!-- Início do formulário para coleta de informações do usuário. -->

                    <div class="number-container">
                        <!-- Div que envolve o campo de e-mail. -->
                        <input type="email" name="email" id="email" placeholder="Email" required>
                        <!-- Campo de entrada para o usuário inserir seu e-mail, obrigatório para preenchimento. -->
                    </div>

                    <div class="number-container">
                        <!-- Div que envolve o campo de número do cartão. -->
                        <input type="text" name="card-number" id="card-number" maxlength="19" placeholder="Número do Cartão" required>
                        <!-- Campo para o usuário inserir o número do cartão de crédito, com um limite de 19 caracteres. -->
                    </div>

                    <div class="name-container">
                        <!-- Div que envolve o campo de nome completo. -->
                        <input type="text" name="name-text" id="name-text" maxlength="30" placeholder="Nome Completo" required>
                        <!-- Campo de entrada para o nome completo do usuário, também obrigatório. -->
                    </div>

```

```

        <div class="infos-container">
            <!-- Div que envolve os campos de validade do cartão e
            código de segurança (CVV). -->

            <div class="expiration-container">
                <!-- Div para agrupar o campo de validade do cartão.
-->
                <input type="text" name="valid-thru-text" id="valid-
thru-text" maxlength="5" placeholder="Validade" required>
                    <!-- Campo de entrada para a data de validade do
cartão (formato MM/AA). -->
                </div>

                <div class="cvv-container">
                    <!-- Div para agrupar o campo de CVV. -->
                    <input type="text" name="cvv-text" id="cvv-text"
maxlength="4" placeholder="CVV" required>
                        <!-- Campo de entrada para o código de segurança do
cartão (CVV), com limite de 4 caracteres. -->
                    </div>

                    <label for="metodoPagamento">Método de Pagamento:</label>
                    <!-- Rótulo descritivo para o campo de seleção do método
de pagamento. -->

                    <select id="metodoPagamento" name="metodoPagamento"
required>
                        <!-- Campo de seleção para o usuário escolher entre
diferentes métodos de pagamento. -->
                        <option value="" disabled selected>Escolha uma
opção</option>
                        <option value="Debito">Débito</option>
                        <option value="Credito">Crédito</option>
                        <option value="Pix">Pix</option>
                    </select>
                </div>

                <button type="button" id="incrementar">+</button>
                <!-- Botão para aumentar a quantidade de itens (como
ingressos). -->

                <button type="button" id="decrementar">-</button>
                <!-- Botão para diminuir a quantidade de itens. -->

                <label for="quantidade">Quantidade:</label>
                <!-- Rótulo descritivo para o campo de quantidade. -->

                <div class="quantidade-container">
                    <input type="number" id="quantidade" name="quantidade"
min="1" value="1" readonly>
                        <!-- Campo numérico para mostrar a quantidade de itens,
com valor mínimo de 1 e readonly (somente leitura). -->
                </div>

                <button type="submit" class="enviar">Comprar</button>
                <!-- Botão para enviar o formulário com as informações
preenchidas. -->

            </form>

```

```
<!-- Div que será usada para exibir a resposta da API, por  
exemplo, uma confirmação de pagamento. -->  
  
<div id="ingressosContainer">  
    <!-- Div para adicionar dinamicamente os ingressos gerados. -->  
->  
    </div>  
  
    <div id="response"></div>  
    </div>  
    </section>  
</main>  
  
<!-- Link para o arquivo JavaScript que conterá a lógica do formulário,  
manipulação da API e outras interações. -->  
  
</body>  
<script src="Pim/Home/Compras/compras.js"></script>  
<script src="auth.js"></script>  
<script src="https://kit.fontawesome.com/16ffb79ef8.js"  
crossorigin="anonymous"></script>  
</body>  
  
</html>
```

APÊNDICE BS – CODIGO SITE – COMPRAS – CSS

```
* {
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    outline: none;
    border: none;
    text-decoration: none;
    text-transform: capitalize;
    transition: .2s linear;
}

body {
    height: 100vh;
    background-color: #010103;
    color: #fff;
}

/* Começo-Header */
header {
    background-color: #444;
}

.nav-bar {
    display: flex;
    justify-content: space-between;
    padding: 1.5rem 6rem;
}

.logo {
    display: flex;
    align-items: center;
}

.logo h1 {
    color: #fff;
}

.nav-list {
    display: flex;
    align-items: center;
}

.nav-list ul {
    display: flex;
    justify-content: center;
    list-style: none;
}

.nav-item {
    margin: 0 35px;
}

.nav-link {
    text-decoration: none;
    font-size: 1.15rem;
    color: #fff;
    font-weight: 400;
}
```

```
.login-button button {
    border: none;
    padding: 10px 15px;
    border-radius: 5px;
    background-color: #970000;
}

.login-button button a {
    text-decoration: none;
    color: #fff;
    font-weight: 500;
    font-size: 1.1rem;
}

.login-button button.login {
    background-color: #970000; /* Cor de fundo do botão de login */
    color: #fff;
    border: none;
    padding: 10px 20px;
    cursor: pointer;
}

/* Estilos para o botão de logout */
.login-button button.logout {
    color: #fff;
    border: 1px solid #970000;
    padding: 10px 20px;
    cursor: pointer;
    display: none; /* Esconde o botão de logout por padrão */
}

.login-button button:empty {
    display: none;
}

.mobile-menu-icon {
    display: none;
}

.mobile-menu {
    display: none;
}

/* Estilo do container principal */
.container {
    width: 100%;
    height: 100%;
    margin: 7% auto;
}

.ui {
    backdrop-filter: blur(50px);
    border: 0;
    box-shadow: 0 0 10px 0 rgba(149, 184, 209, .5);
    border-radius: 10px;
    width: 50%;
    margin: 0 auto;
    padding: 5%;
    display: flex;
    gap: 20px;
    background-color: rgb(0, 0, 0);
    border: 3px solid #fffff4d;
```

```
}

/* Estilo dos botões */
button {
    background-color: #cc5252;
    color: white;
    border: none;
    padding: 10px 15px;
    margin: 10px;
    border-radius: 5px;
    cursor: pointer;
}

button:hover {
    background-color: #b92323;
}

input[type="text"],
input[type="email"],
select {
    width: calc(100% - 22px);
    padding: 10px;
    margin: 10px 0;
    border: 1px solid #fffffff4d;
    border-radius: 4px;
}

select {
    color: #ffffff;
    background-color: #333333;
}

#credit-card {
    display: flex;
    flex-direction: column;
    gap: 15px;
    font-size: 1.1rem;
    color: white;
    text-transform: uppercase;
}

.number-container,
.name-container {
    width: 50%;
    display: flex;
    flex-direction: column;
}

.expiration-container,
.cvv-container {
    width: 50%;
}

input {
    border: 0;
    border-radius: 5px;
    padding: 10px;
    background-color: #333333;
    color: white;
    font-weight: 600;
    font-size: 0.8rem;
    outline: 0;
    width: 90%;
```

```
}

input[type="text"]:focus {
    border: 1px solid #cc5252;
}

input#valid-thru-text,
input#cvv-text {
    width: 80%;
}

input[type="submit"] {
    width: 95%;
    background-color: #cc5252;
    cursor: pointer;
}

.infos-container {
    display: flex;
}

/* Estilo do container de ingressos */
.ingresso {
    border: 1px solid #fffff4d;
    padding: 10px;
    margin: 10px 0;
    border-radius: 4px;
    background-color: #fffff4d;
}

.resposta-ingresso {
    margin-top: 20px;
    margin-bottom: 10px;
    padding: 10px;
    border: 1px solid #cc5252;
    border-radius: 5px;
    background-color: #fffff4d;
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
}

/* Esconde o container de ingressos se não houver ingressos */
#ingressosContainer {
    display: none;
}

/* Exibe o container de ingressos quando houver ingressos */
#ingressosContainer:has(.ingresso) {
    display: block;
}
#response {
    margin-top: 20px;
    padding: 15px;
    border: 1px solid #cc5252;
    border-radius: 5px;
    background-color: #fffff4d;
    color: #cc5252;
    text-align: left;
}

#ingressosContainer {
    margin-top: 20px;
}
```

```
@media screen and (max-width: 440px) {
  body {
    overflow-y: scroll;
  }
  .nav-bar {
    padding: 1.5rem 4rem;
  }
  .nav-item {
    display: none;
  }
  .login-button {
    display: none;
  }
  .mobile-menu-icon {
    display: block;
  }
  .mobile-menu-icon button {
    background-color: transparent;
    border: none;
    cursor: pointer;
  }
  .mobile-menu ul {
    display: flex;
    flex-direction: column;
    text-align: center;
    padding-bottom: 1rem;
  }
  .mobile-menu .nav-item {
    display: block;
    padding-top: 1.2rem;
  }
  .mobile-menu .login-button {
    display: block;
    padding: 1rem 2rem;
  }
  .mobile-menu .login-button button {
    width: 100%;
  }
  .open {
    display: block;
  }
}

.home {
  scroll-behavior: auto;
  width: 100%;
  height: 750px;
  text-align: center;
  justify-items: center;
}

.home .content h3 {
  font-size: 22px;
}

.home .content p {
  font-size: 18px;
  color: #eee;
}

.ui {
```

```
    margin: 12% auto;
    width: 100%;
}

select {
    font-size: 10px;
}

.infos-container label {
    font-size: 14px;
}
}
```

APÊNDICE BT – CODIGO SITE – COMPRAS – JAVASCRIPT

```

// Variável para controlar a quantidade de ingressos
let quantidadeIngressos = 0;
console.log(localStorage.getItem("apiBase"));
const apiBase = localStorage.getItem("apiBase") || "https://default-api-url.com";

// Função para mostrar/ocultar o menu
function menuShow() {
    let menuMobile = document.querySelector('.mobile-menu');
    let menuIcon = document.querySelector('.icon');

    // Verifica se o menu está aberto ou fechado e alterna o estado
    if (menuMobile.classList.contains('open')) {
        menuMobile.classList.remove('open'); // Fecha o menu
        menuIcon.src = "assets/img/menu_white_36dp.svg"; // Troca o ícone para
        "abrir"

        // Resetando a altura para 0 após o fechamento
        setTimeout(() => {
            menuMobile.style.height = '0'; // A altura do menu é definida como 0
            // Atualiza a página após o fechamento do menu
            window.location.reload(); // Recarrega a página
        }, 300); // Aguardar a transição de fechamento antes de resetar a altura
    } else {
        menuMobile.classList.add('open'); // Abre o menu
        menuIcon.src = "assets/img/close_white_36dp.svg"; // Troca o ícone para
        "fechar"

        // Ajusta a altura para auto para que o menu ocupe o espaço necessário
        menuMobile.style.height = 'auto';
    }
}

// Função para fechar o menu ao clicar fora (e resetar)
window.addEventListener('click', function (event) {
    let menuMobile = document.querySelector('.mobile-menu');
    let menuIconButton = document.querySelector('.mobile-menu-icon button');

    // Verifica se o clique foi fora do menu e do ícone do menu
    if (!menuMobile.contains(event.target) &&
        !menuIconButton.contains(event.target)) {
        if (menuMobile.classList.contains('open')) {
            menuMobile.classList.remove('open'); // Fecha o menu
            document.querySelector('.icon').src =
            "assets/img/menu_white_36dp.svg"; // Troca o ícone para "abrir"

            // Resetando a altura para 0 após o fechamento
            setTimeout(() => {
                menuMobile.style.height = '0'; // A altura do menu é definida
                como 0
                // Atualiza a página após o fechamento do menu
                window.location.reload(); // Recarrega a página
            }, 300); // Aguardar a transição de fechamento antes de resetar a
            altura
        }
    }
});

// Impede que o clique no ícone do menu feche o menu ao clicar nele
document.querySelector('.mobile-menu-icon button').addEventListener('click',
function(event) {

```

```

    event.stopPropagation(); // Impede que o clique no ícone seja tratado como
    um clique fora do menu
});

// Inicializa o valor do campo de quantidade
document.getElementById('quantidade').value = quantidadeIngressos;

// Evento para incrementar a quantidade de ingressos
document.getElementById('incrementar').addEventListener('click', function() {
    quantidadeIngressos++;
    document.getElementById('quantidade').value = quantidadeIngressos; //
Atualiza o valor exibido
    addIngresso(); // Adiciona um novo ingresso
});

// Evento para decrementar a quantidade de ingressos
document.getElementById('decrementar').addEventListener('click', function() {
    if (quantidadeIngressos > 0) {
        quantidadeIngressos--;
        document.getElementById('quantidade').value = quantidadeIngressos; //
Atualiza o valor exibido
        removeIngresso(); // Remove um ingresso
    }
});

// Função para adicionar um novo ingresso ao formulário
function addIngresso() {
    const ingressosContainer = document.getElementById('ingressosContainer'); //
Seleciona o container de ingressos
    const ingressos = ingressosContainer.getElementsByClassName('ingresso'); //
Obtém todos os ingressos existentes

    // Adiciona o novo ingresso se o número de ingressos é menor que a quantidade
    const newIngresso = document.createElement('div'); // Cria um novo elemento
div
    newIngresso.classList.add('ingresso'); // Adiciona a classe 'ingresso'
    newIngresso.innerHTML =
        '<label for="nome">Nome do Ingresso:</label>
        <input type="text" class="nome" required> <!-- Campo para o nome do
ingresso -->

        <label for="tipo">Tipo de Ingresso:</label>
        <select class="tipo" required> <!-- Seleção do tipo de ingresso -->
            <option value="Inteiro">Inteiro</option>
            <option value="Meia">Meia</option>
            <option value="Isento">Isento</option>
        </select>
';
    ingressosContainer.appendChild(newIngresso); // Adiciona o novo ingresso ao
container
    ingressosContainer.style.display = 'block'; // Garante que o container de
ingressos esteja visível
}

// Função para remover o último ingresso do formulário
function removeIngresso() {
    const ingressosContainer = document.getElementById('ingressosContainer'); //
Seleciona o container de ingressos
    const ingressos = ingressosContainer.getElementsByClassName('ingresso'); //
Obtém todos os ingressos existentes

    // Verifica se existem ingressos para remover
}

```

```

if (ingressos.length > 0) {
    ingressosContainer.removeChild(ingressos[ingressos.length - 1]); // Remove o último ingresso
}

// Esconde o container se não houver ingressos
if (ingressosContainer.getElementsByClassName('ingresso').length === 0) {
    ingressosContainer.style.display = 'none';
}
}

// Evento para enviar os dados para a API ao submeter o formulário
document.getElementById('apiForm').addEventListener('submit', function(event) {
    event.preventDefault(); // Impede o envio padrão do formulário

    // Verifica se a quantidade de ingressos é maior que zero
    if (quantidadeIngressos === 0) {
        alert("A quantidade deve ser maior que 0."); // Alerta ao usuário
        return;
    }

    const ingressos = []; // Array para armazenar os ingressos
    const ingressoElements = document.querySelectorAll('.ingresso'); // Obtém todos os elementos de ingresso

    // Itera sobre cada ingresso e armazena os dados em um array
    ingressoElements.forEach(ingresso => {
        const nome = ingresso.querySelector('.nome').value; // Nome do ingresso
        const tipo = ingresso.querySelector('.tipo').value; // Tipo do ingresso
        ingressos.push({ nome, tipo }); // Adiciona ao array de ingressos
    });

    // Captura os dados do formulário
    const email = document.getElementById('email').value;
    const metodoPagamento = document.getElementById('metodoPagamento').value;

    // Cria o objeto que será enviado para a API
    const novaVendaDTO = {
        ingressoDTO: ingressos,
        venda: {
            email: email,
            quantidade: quantidadeIngressos,
            metodoPagamento: metodoPagamento
        }
    };

    const jsonString = JSON.stringify(novaVendaDTO); // Converte o objeto em string JSON

    console.log('Dados a serem enviados:', jsonString); // Log para depuração

    // Envia os dados para a API
    fetch(apiBase + '/api/Master/novavenda', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json',
            "ngrok-skip-browser-warning": "true" // Define o tipo de conteúdo
        },
        body: jsonString // Dados a serem enviados
    })
    .then(response => {
        console.log('Status da resposta:', response.status); // Log do status da resposta
    })
}
}

```

```
        return response.text(); // Lê a resposta como texto
    })
    .then(text => {
        console.log('Resposta da API:', text); // Log da resposta da API
        if (text) {
            return JSON.parse(text); // Converte o texto em JSON
        } else {
            throw new Error('Resposta vazia da API'); // Lança erro se a
resposta estiver vazia
        }
    })
    .then(data => {
        // Formata a resposta da API
        const formattedResponse = data.map(item => `
            <div class="resposta-ingresso">
                <strong>Nome:</strong> ${item.nome} <br>
                <strong>Chave:</strong> ${item.key}
            </div>
        `).join('');

        // Exibe a resposta formatada na div de resposta
        document.getElementById('response').innerHTML = formattedResponse;
    })
    .catch(error => {
        console.error('Erro:', error); // Log de erros
        document.getElementById('response').textContent = 'Erro ao enviar os
dados para a API: ' + error.message; // Exibe erro na tela
    });
});
```

APÊNDICE BU – CODIGO SITE – LOGIN – HTML

```

<!DOCTYPE html>
<html lang="pt-br" dir="ltr">

<head>
    <!-- Declaração do tipo de documento HTML5 -->
    <meta charset="utf-8">
    <!-- Define o conjunto de caracteres para UTF-8 -->
    <title>Pagina de Login</title>
    <!-- Título da página -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <script src="apiconfig.js"></script>
    <!-- Link para o arquivo de estilo CSS externo -->
    <link rel="stylesheet" href=".//Pim/Home/login/login.css">

    <!-- Link para a biblioteca de ícones Font Awesome -->
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css">
</head>

<body>

    <!-- Cabeçalho da página -->
    <header>
        <nav class="nav-bar">
            <div class="logo">
                <h1>MECH</h1>
            </div>
            <div class="nav-list">
                <ul>
                    <li class="nav-item"><a href="home.html" class="nav-link">Página Principal</a></li>
                    <li class="nav-item"><a href="informações.html" class="nav-link">Informações</a></li>
                    <li class="nav-item"><a href="exposição.html" class="nav-link"> Exposições</a></li>
                    <li class="nav-item"><a href="compras.html" class="nav-link"> Compras</a></li>
                    <li class="nav-item"><a href="questionario.html" class="nav-link"> Questionario/Avaliação</a></li>
                </ul>
            </div>
            <div class="login-button">
                <button id="loginBtn" class="login"><a href="login.html" id="loginLink">Entrar</a></button>
                <button id="logoutBtn" class="logout" onclick="fazerLogout()">Logout</button>
            </div>
            <div class="mobile-menu-icon">
                <button onclick="menuShow()"></button>
            </div>
        </nav>
        <div class="mobile-menu">
            <ul>
                <li class="nav-item"><a href="home.html" class="nav-link">Página Principal</a></li>
                <li class="nav-item"><a href="informações.html" class="nav-link">Informações</a></li>
            </ul>
        </div>
    </header>

```

```

        <li class="nav-item"><a href="exposicao.html" class="nav-link">
Exposições</a></li>
        <li class="nav-item"><a href="compras.html" class="nav-link">
Compras</a></li>
        <li class="nav-item"><a href="questionario.html" class="nav-
link"> Questionario/Avaliação</a></li>
    </ul>

    <div class="login-button">
        <button id="loginBtn" class="login"><a href="login.html"
id="loginLink">Entrar</a></button>
        <button id="logoutBtn" class="logout"
onclick="fazerLogout()">Logout</button>
    </div>
</div>
</header>

<!-- Conteúdo centralizado da página, onde está o formulário de login --&gt;
&lt;div class="center"&gt;
    &lt;h1&gt;Login&lt;/h1&gt;
    <!-- Formulário de login, enviado via método POST --&gt;
    &lt;form method="post" class="formulario" id="login-form" onsubmit="return
fazerLogin(event)"&gt;

        <!-- Campo para o email do usuário --&gt;
        &lt;div class="txt_field"&gt;
            &lt;span&gt;
                &lt;input type="text" id="email" required&gt; &lt;!-- Campo de entrada
para o email --&gt;
                &lt;label&gt;Email &lt;i class="bx bx-user icon"&gt;&lt;/i&gt; &lt;/label&gt; &lt;!--
Rótulo e ícone de usuário --&gt;
            &lt;/span&gt;
        &lt;/div&gt;

        <!-- Campo para a senha do usuário --&gt;
        &lt;div class="txt_field"&gt;
            &lt;span&gt;
                &lt;input type="password" id="password" required&gt; &lt;!-- Campo de
entrada para a senha --&gt;
                &lt;label&gt;Key &lt;i class="bx bx-lock - alt icon"&gt;&lt;/i&gt; &lt;/label&gt; &lt;!--
- Rótulo e ícone de cadeado --&gt;
            &lt;/span&gt;
        &lt;/div&gt;

        <!-- Botão para enviar o formulário --&gt;
        &lt;input type="submit" value="Login"&gt;

        <!-- Link para realizar compra se o usuário não tiver cadastro --&gt;
        &lt;div class="signup_link"&gt;
            Sem cadastro ? &lt;a href="compras.html"&gt;Realizar Compra&lt;/a&gt;
        &lt;/div&gt;
    &lt;/form&gt;
&lt;/div&gt;

<!-- Link para o arquivo JavaScript que vai adicionar interatividade ao login
--&gt;
&lt;script src="auth.js"&gt;&lt;/script&gt;
&lt;script src="Pim/Home/login/login.js"&gt;&lt;/script&gt;
&lt;script src="https://kit.fontawesome.com/16ffb79ef8.js"
crossorigin="anonymous"&gt;&lt;/script&gt;
&lt;/body&gt;
</pre>

```

</html>

APÊNDICE BV – CODIGO SITE – LOGIN – CSS

```

/* Reset básico */
* {
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    outline: none;
    border: none;
    text-decoration: none;
    text-transform: capitalize;
    transition: .2s linear;
}

/* Estilo para o corpo da página */
body {
    height: 100vh;
    overflow: hidden;
    background-size: cover;
    background-image: url(src/Imagen\ do\ WhatsApp\ de\ 2024-09-09\ à\s\)\20.05.33_e4f746e4.jpg);
}

/* Estilo para o Header */
header {
    background-color: #444;
    position: fixed;
    width: 100%;
    top: 0;
    z-index: 1000;
}

.nav-bar {
    display: flex;
    justify-content: space-between;
    padding: 1.5rem 6rem;
}

.logo {
    display: flex;
    align-items: center;
}

.logo h1 {
    color: #fff;
}

.nav-list {
    display: flex;
    align-items: center;
}

.nav-list ul {
    display: flex;
    justify-content: center;
    list-style: none;
}

.nav-item {
    margin: 0 35px;
}

```

```
.nav-link {  
    text-decoration: none;  
    font-size: 1.15rem;  
    color: #fff;  
    font-weight: 400;  
}  
  
.login-button button {  
    border: none;  
    padding: 10px 15px;  
    border-radius: 5px;  
    background-color: #970000;  
}  
  
.login-button button a {  
    text-decoration: none;  
    color: #fff;  
    font-weight: 500;  
    font-size: 1.1rem;  
}  
  
.login-button button.login {  
    background-color: #970000; /* Cor de fundo do botão de login */  
    color: #fff;  
    border: none;  
    padding: 10px 20px;  
    cursor: pointer;  
}  
  
/* Estilos para o botão de logout */  
.login-button button.logout {  
    color: #fff;  
    border: 1px solid #970000;  
    padding: 10px 20px;  
    cursor: pointer;  
    display: none; /* Esconde o botão de logout por padrão */  
}  
  
.login-button button:empty {  
    display: none;  
}  
  
.mobile-menu-icon {  
    display: none;  
}  
  
.mobile-menu {  
    display: none;  
}  
  
/* Estilo do Centro */  
.center {  
    position: absolute;  
    top: 50%;  
    left: 50%;  
    transform: translate(-50%, -50%);  
    width: 400px;  
    backdrop-filter: blur(35px);  
    border-radius: 10px;  
    box-shadow: 10px 10px 15px rgba(0, 0, 0, 0.05);  
}  
  
.center h1 {
```

```
text-align: center;
padding: 20px 0;
color: white;
border-bottom: 1px solid silver;
}

.center form {
    padding: 0 40px;
    box-sizing: border-box;
}

form .txt_field {
    position: relative;
    border-bottom: 2px solid #adadad;
    margin: 30px 0;
}

.txt_field input {
    width: 100%;
    padding: 0 5px;
    height: 40px;
    font-size: 16px;
    border: none;
    background: none;
    outline: none;
    color: white; /* Cor do texto nos campos de entrada */
}

.txt_field label {
    position: absolute;
    top: 50%;
    left: 5px;
    color: #adadad;
    transform: translateY(-50%);
    font-size: 16px;
    pointer-events: none;
    transition: .5s;
}

.txt_field span::before {
    position: absolute;
    top: 40px;
    left: 0;
    width: 0%;
    height: 2px;
    background: #2691d9;
    transition: .5s;
}

.txt_field input:focus~label,
.txt_field input:valid~label {
    top: -5px;
    color: #2691d9;
}

.txt_field input:focus~span::before,
.txt_field input:valid~span::before {
    width: 100%;
}

input[type="submit"] {
    width: 100%;
    height: 50px;
```

```
border: 1px solid;
background: #000000;
border-radius: 25px;
font-size: 18px;
color: #e9f4fb;
font-weight: 700;
cursor: pointer;
outline: none;
}

input[type="submit"]:hover {
    border-color: #000000;
    transition: .5s;
}

.signup_link {
    margin: 30px 0;
    text-align: center;
    font-size: 16px;
    color: #ffffff;
}

.signup_link a {
    color: black;
    text-decoration: none;
}

.signup_link a:hover {
    text-decoration: underline;
}

/* Media Query para mobile */
@media screen and (max-width: 440px) {
    body {
        background-size: auto;
        height: 100%;
        overflow-y: scroll;
    }
    .nav-bar {
        padding: 1.5rem 4rem;
    }
    .nav-item {
        display: none;
    }
    .login-button {
        display: none;
    }
    .mobile-menu-icon {
        display: block;
    }
    .mobile-menu-icon button {
        background-color: transparent;
        border: none;
        cursor: pointer;
    }
    .mobile-menu ul {
        display: flex;
        flex-direction: column;
        text-align: center;
        padding-bottom: 1rem;
    }
    .mobile-menu .nav-item {
        display: block;
    }
}
```

```
    padding-top: 1.2rem;
}
.mobile-menu .login-button {
    display: block;
    padding: 1rem 2rem;
}
.mobile-menu .login-button button {
    width: 100%;
}
.open {
    display: block;
}

.home {
    scroll-behavior: auto;
    width: 100%;
    height: 750px;
    text-align: center;
    justify-items: center;
}
.home .content h3 {
    font-size: 22px;
}
.home .content p {
    font-size: 18px;
    color: #eee;
}
}

.center .formualrio {
    width: 100%;
}
}
```

APÊNDICE BW – CODIGO SITE – LOGIN – JAVASCRIPT

```

console.log(localStorage.getItem("apiBase"));
const apiBase = localStorage.getItem("apiBase") || "https://default-api-url.com";

if (!localStorage.getItem("Logged")) {
    localStorage.setItem("Logged", "false");
}

// Função para mostrar/ocultar o menu
function menuShow() {
    let menuMobile = document.querySelector('.mobile-menu');
    let menuIcon = document.querySelector('.icon');

    // Verifica se o menu está aberto ou fechado e alterna o estado
    if (menuMobile.classList.contains('open')) {
        menuMobile.classList.remove('open'); // Fecha o menu
        menuIcon.src = "assets/img/menu_white_36dp.svg"; // Troca o ícone para
        "abrir"

        // Resetando a altura para 0 após o fechamento
        setTimeout(() => {
            menuMobile.style.height = '0'; // A altura do menu é definida como 0
            // Atualiza a página após o fechamento do menu
            window.location.reload(); // Recarrega a página
        }, 300); // Aguardar a transição de fechamento antes de resetar a altura
    } else {
        menuMobile.classList.add('open'); // Abre o menu
        menuIcon.src = "assets/img/close_white_36dp.svg"; // Troca o ícone para
        "fechar"

        // Ajusta a altura para auto para que o menu ocupe o espaço necessário
        menuMobile.style.height = 'auto';
    }
}

// Função para fechar o menu ao clicar fora (e resetar)
window.addEventListener('click', function (event) {
    let menuMobile = document.querySelector('.mobile-menu');
    let menuIconButton = document.querySelector('.mobile-menu-icon button');

    // Verifica se o clique foi fora do menu e do ícone do menu
    if (!menuMobile.contains(event.target) &&
        !menuIconButton.contains(event.target)) {
        if (menuMobile.classList.contains('open')) {
            menuMobile.classList.remove('open'); // Fecha o menu
            document.querySelector('.icon').src =
            "assets/img/menu_white_36dp.svg"; // Troca o ícone para "abrir"

            // Resetando a altura para 0 após o fechamento
            setTimeout(() => {
                menuMobile.style.height = '0'; // A altura do menu é definida
                como 0
                // Atualiza a página após o fechamento do menu
                window.location.reload(); // Recarrega a página
            }, 300); // Aguardar a transição de fechamento antes de resetar a
            altura
        }
    }
});

// Impede que o clique no ícone do menu feche o menu ao clicar nele

```

```

document.querySelector('.mobile-menu-icon button').addEventListener('click',
function(event) {
    event.stopPropagation(); // Impede que o clique no ícone seja tratado como
    // um clique fora do menu
});

// Função de validação de email usando regex
function validarEmail(email) {
    // Regex para validar o formato do email
    const regex = /^[^s@]+@[^\s@]+\.[^\s@]+$/;
    return regex.test(email); // Retorna true se o email for válido, false caso
    contrário
}

// Função para validar a Key (verifica se tem exatamente 9 caracteres)
function validarKey(key) {
    return key.length === 9; // Retorna true se a key tiver exatamente 9
    caracteres
}

// Função de login que faz a validação e chama a API
function fazerLogin(event) {
    event.preventDefault();

    if (validateLogin(event)) {
        return;
    }

    // Obtém os valores dos campos de entrada
    const email = document.getElementById("email").value.trim().toLowerCase();
    const key = document.getElementById("password").value.trim().toUpperCase();
    const mensagem = document.getElementById("mensagem");

    // Reseta as mensagens
    if (mensagem) mensagem.textContent = "";

    // Validação de email e Key
    if (!validarEmail(email)) {
        mensagem.textContent = "Formato de email inválido!";
        return; // Sai da função se o email for inválido
    }
    if (!validarKey(key)) {
        mensagem.textContent = "A Key deve ter exatamente 9 caracteres!";
        return; // Sai da função se a key for inválida
    }

    // Formatação do parâmetro [emailKey]
    const emailKey = email + key;

    // URL da API com o emailKey
    const apiUrl = `${apiBase}/api/Master/login/${emailKey}`;

    // Faz a requisição GET para a API
    fetch(apiUrl, {
        method: "GET",
        headers: {
            "Accept": "application/json",
            "ngrok-skip-browser-warning": "true" // Adiciona o cabeçalho para
            pular a página de aviso
        },
    })
    .then(response => {
        if (!response.ok) {

```

```

        return response.text().then(text => {
            throw new Error(`Erro ${response.status}: ${text}`);
        });
    }

    const contentType = response.headers.get("Content-Type");
    if (contentType && contentType.includes("application/json")) {
        return response.json(); // Apenas tenta parsear JSON se o
Content-Type for JSON
    } else {
        throw new Error("Resposta inesperada (não JSON). Conteúdo
retornado não esperado.");
    }
}
.then(data => {
    if (data.success) {
        localStorage.setItem("Logged", "true");
        alert("Login bem-sucedido!");
        window.location.href = "home.html";
    } else {
        mensagem.textContent = data.errorMessage || "Erro desconhecido!";
    }
})
.catch(error => {
    mensagem.textContent = "Erro de Login ou Conexão: " + error.message;
});
}

function validateLogin(event) {
    event.preventDefault(); // Impede o envio do formulário padrão

    var email = document.getElementById('email').value;
    var password = document.getElementById('password').value;

    // Lógica para verificar se o login é de admin
    if (email === "admin@marte.com" && password === "admin123") {
        // Redireciona para a página de administração
        window.location.href = "admin_dashboard.html"; // Página exclusiva para
o painel do administrador
    } else {
        return false;
    }

    return false; // Impede o envio do formulário padrão
}

```

APÊNDICE BX – CODIGO SITE – ADMIN_DASHBOARD – HTML

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Painel do Administrador</title>
    <link rel="stylesheet" href="Pim/Home/Admin/admin.css"> <!-- Inclua seu CSS
aqui -->
</head>
<body>
    <header>
        <nav class="nav-bar">
            <div class="logo">
                <h1>MECH – Admin</h1>
            </div>
            <div class="nav-list">
                <ul>
                    <li class="nav-item"><a href="home.html" class="nav-link">Sair</a></li>
                </ul>
            </div>
        </nav>
    </header>

    <main>
        <h2>Bem-vindo ao Painel de Administração</h2>
        <p>Aqui você pode gerenciar exposições, informações, e muito mais.</p>
        <!-- Exemplo de opções de administração -->
        <ul>
            <li><a href="Pim/Home/Admin/RelatorioQuestionarioAvaliacao/index.html">Relatório dos Questionários e Avaliações</a></li>
            <li><a href="Pim/Home/Admin/Relatorios/index.html">Relatórios e Listas</a></li>
        </ul>
    </main>

</body>
</html>

```

APÊNDICE BY – CODIGO SITE – ADMIN – CSS

```

/* Reset básico */
* {
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    outline: none;
    border: none;
    text-decoration: none;
    text-transform: capitalize;
    transition: .2s linear;
}

/* Estilo básico do corpo */
body {
    background-color: #000000; /* Fundo escuro */
    color: #fff; /* Texto branco */
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    height: 100vh;
    text-align: center;
}

/* Estilo do header */
header {
    width: 100%;
    background-color: #444; /* Cor do fundo do header */
    padding: 15px 0;
    position: fixed;
    top: 0;
    z-index: 1000;
}

.nav-bar {
    display: flex;
    justify-content: space-between;
    align-items: center;
    width: 80%;
    margin: 0 auto;
}

.logo h1 {
    font-size: 1.8rem;
    color: #fff;
}

.nav-list ul {
    list-style: none;
    display: flex;
    justify-content: flex-end;
}

.nav-item {
    margin-left: 25px;
}

.nav-link {
    color: #fff;
}

```

```
    font-size: 1rem;
    font-weight: 500;
    transition: color 0.3s ease;
}

.nav-link:hover {
    color: #970000; /* Cor de destaque */
}

/* Estilo do main */
main {
    margin-top: 100px; /* Espaço para o header fixo */
    width: 80%;
    max-width: 800px;
    text-align: center;
}

h2 {
    font-size: 2rem;
    margin-bottom: 20px;
}

p {
    font-size: 1.2rem;
    color: #bbb;
    margin-bottom: 30px;
}

/* Estilo dos links no main */
main ul {
    list-style: none;
    padding: 0;
}

main ul li {
    margin: 20px 0;
}

main ul li a {
    color: #f0f0f0;
    font-size: 1.2rem;
    font-weight: 600;
    padding: 10px 20px;
    display: inline-block;
    border-radius: 5px;
    background-color: #555; /* Cor de fundo dos links */
    transition: background-color 0.3s ease, color 0.3s ease;
}

main ul li a:hover {
    background-color: #970000; /* Cor de fundo ao passar o mouse */
    color: #fff; /* Cor do texto no hover */
}

/* Foco nos dois links principais (relatórios) */
main ul li:nth-child(1) a,
main ul li:nth-child(2) a {
    background-color: #777; /* Cor de fundo inicial */
    font-size: 1.3rem;
    padding: 15px 25px;
}

main ul li:nth-child(1) a:hover,
```

```
main ul li:nth-child(2) a:hover {  
    background-color: #970000; /* Cor de destaque mais intensa */  
    color: #fff;  
}  
  
/* Media Query para dispositivos móveis */  
@media screen and (max-width: 768px) {  
    body {  
        padding: 10px;  
    }  
  
    .nav-bar {  
        width: 100%;  
    }  
  
    .logo h1 {  
        font-size: 1.5rem;  
    }  
  
    main {  
        width: 90%;  
    }  
  
    h2 {  
        font-size: 1.5rem;  
    }  
  
    main ul li a {  
        font-size: 1rem;  
        padding: 8px 16px;  
    }  
}
```

APÊNDICE BZ – CODIGO SITE – RELATORIOQUESTIONARIOAVALIAÇÃO – HTML

```

<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Relatório dos Questionários e Avaliações</title>
    <link rel="stylesheet" href="styles.css"> <!-- Arquivo de estilo externo -->
    <script src="../../apiConfig.js"></script>
    <script src="script.js" defer></script>
</head>
<body>

    <div class="container">
        <!-- Botão Voltar -->
        <button onclick="window.history.back()">Voltar</button>

    <div class="container">
        <h1>Relatório dos Questionários e Avaliações</h1>

        <div class="section" id="questionario">
            <h2>Questionário</h2>
            <table id="questionario-content">
                <thead>
                    <tr>
                        <th>Número</th>
                        <th>Pergunta</th>
                        <th>Acertos</th>
                        <th>Erros</th>
                        <th>Relatório</th>
                    </tr>
                </thead>
                <tbody>
                    <!-- Linhas da tabela inseridas pelo JavaScript -->
                </tbody>
            </table>
        </div>

        <div class="section" id="avaliacao">
            <h2>Avaliação</h2>
            <table id="avaliacao-content">
                <thead>
                    <tr>
                        <th>Número</th>
                        <th>Pergunta</th>
                        <th>Excelente</th>
                        <th>Bom</th>
                        <th>Regular</th>
                        <th>Ruim</th>
                        <th>Péssimo</th>
                        <th>Relatório</th>
                    </tr>
                </thead>
                <tbody>
                    <!-- Linhas da tabela inseridas pelo JavaScript -->
                </tbody>
            </table>
        </div>
    </div>

```

```
<div class="section" id="sugestoes">
  <h2>Sugestões dos Visitantes</h2>
  <div id="sugestoes-container">
    <!-- Sugestões serão inseridas pelo JavaScript -->
    <br> <br>
  </div>
</div>
</body>
</html>
```

APÊNDICE CA – CODIGO SITE – RELATORIOQUESTIONARIOAVALIACAO – CSS

```

/* Estilos gerais */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    font-family: Arial, sans-serif;
    background-color: #000000;
}

.container {
    width: 90%;
    max-width: 1200px;
    margin: 50px auto;
    padding: 20px;
    background-color: #000000;
    border-radius: 8px;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
}

h1 {
    color: #ffffff;
    text-align: center;
    margin-bottom: 20px;
}

.section {
    margin-bottom: 30px;
}

h2 {
    color: #c74444;
    margin-bottom: 15px;
    text-align: center;
}

/* Estilos das tabelas */
table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 10px;
    background-color: #f9f9f9;
}

th, td {
    padding: 10px;
    border: 1px solid #ddd;
    text-align: center;
}

th {
    background-color: #c74444;
    color: white;
}

td {

```

```
        color: #333;
    }

/* Estilos das sugestões */
#sugestoes-container {
    margin-top: 30px;
    padding: 10px;
    background-color: #f9f9f9;
    border-radius: 8px;
}

#sugestoes-container h2 {
    color: #c74444;
    text-align: center;
    margin-bottom: 15px;
}

/* Estilo de cada sugestão como um cartão */
#sugestoes-container p {
    background-color: #ffffff;
    border: 1px solid #ddd;
    border-radius: 8px;
    padding: 15px;
    margin: 15px 20px; /* Margem adicional ao redor dos cartões */
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
    color: #333;
    font-size: 16px;
    line-height: 1.5;
    transition: transform 0.2s;
}

#sugestoes-container p:hover {
    transform: translateY(-3px);
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.15);
}

/* Estilo do botão "Voltar" */
button {
    background-color: #c74444; /* Destaque na cor vermelha */
    color: white;
    border: none;
    border-radius: 5px;
    padding: 10px 20px;
    font-size: 16px;
    cursor: pointer;
    transition: background-color 0.3s ease, transform 0.3s ease;
    margin-bottom: 20px; /* Espaço abaixo do botão */
    display: block;
    margin-left: auto;
    margin-right: auto; /* Centralizar o botão */
}

button:hover {
    background-color: #a33232; /* Efeito ao passar o mouse */
    transform: translateY(-2px); /* Leve animação de hover */
}

button:active {
    transform: translateY(1px); /* Efeito ao clicar */
}
```

APÊNDICE CB – CODIGO SITE – RELATORIOQUESTIONARIOAVALIACAO – JAVASCRIPT

```

console.log(localStorage.getItem("apiBase"));
const apiBase = localStorage.getItem("apiBase") || "https://default-api-url.com";

// Função para obter as perguntas do questionário e preencher a tabela
async function fetchPerguntas() {

    try {
        const response = await
fetch(`${apiBase}/api/Master/QuestionarioAvaliacao/q`, {
            method: "GET",
            headers: {
                "Accept": "application/json",
                "ngrok-skip-browser-warning": "true" // Cabeçalho para evitar
a página de aviso
            }
        });
        // Verifica se a resposta foi bem-sucedida e tem o tipo de conteúdo
esperado
        if (!response.ok) {
            throw new Error(`Erro ${response.status}:
${response.statusText}`);
        }

        const contentType = response.headers.get("content-type");
        if (!contentType || !contentType.includes("application/json")) {
            throw new Error("Resposta não contém JSON válido.");
        }

        const perguntas = await response.json();

        const contentTable = document.querySelector("#questionario-content
tbody");
        contentTable.innerHTML = ""; // Limpa o conteúdo anterior

        // Adiciona cada pergunta como uma linha na tabela
        perguntas.forEach(pergunta => {
            const row = document.createElement("tr");

            row.innerHTML =
                ` ${pergunta.id}</td>                 <td>${pergunta.pergunta}</td>                 <td>${pergunta.acertos}</td>                 <td>${pergunta.erros}</td>                 <td>${calcularRelatorioQuestionario(pergunta.acertos, pergunta .erros)}</td>             `;              contentTable.appendChild(row);         });     } catch (error) {         console.error("Erro ao buscar perguntas do questionário:", error);     } }  // Função para obter as perguntas da avaliação e preencher a tabela async function fetchAvaliacao() { |
```

```

try {
    const response = await
fetch(`${apiBase}/api/Master/QuestionarioAvaliacao/a`, {
    method: "GET",
    headers: {
        "Accept": "application/json",
        "ngrok-skip-browser-warning": "true" // Cabeçalho para evitar a
página de aviso
    }
});

// Verifica se a resposta foi bem-sucedida e tem o tipo de conteúdo
esperado
if (!response.ok) {
    throw new Error(`Erro ${response.status}: ${response.statusText}`);
}

const contentType = response.headers.get("content-type");
if (!contentType || !contentType.includes("application/json")) {
    throw new Error("Resposta não contém JSON válido.");
}

const avaliacoes = await response.json();

const avaliacaoTable = document.querySelector("#avaliacao-content
tbody");
avaliacaoTable.innerHTML = "" // Limpa o conteúdo anterior

// Adiciona cada avaliação como uma linha na tabela
avaliacoes.forEach(avaliacao => {
    const row = document.createElement("tr");

    row.innerHTML =
        ` ${avaliacao.id}</td>         <td>${avaliacao.pergunta}</td>         <td>${avaliacao.excelente}</td>         <td>${avaliacao.bom}</td>         <td>${avaliacao.regular}</td>         <td>${avaliacao.ruim}</td>         <td>${avaliacao.pessimo}</td>         <td>${calcularRelatorioAvaliacao(avaliacao.excelente, avaliacao.bo m, avaliacao.regular, avaliacao.ruim, avaliacao.pessimo)}</td>         `;      avaliacaoTable.appendChild(row); }); } catch (error) {     console.error("Erro ao buscar avaliações:", error); } }  async function fetchSugestoes() { try {     // Faz a requisição para o endpoint de sugestões     const response = await fetch(`${apiBase}/api/Master/QuestionarioAvaliacao/s`, {     method: "GET",     headers: {         "Accept": "application/json",         "ngrok-skip-browser-warning": "true" // Cabeçalho para evitar a página de aviso     } }); } |
```

```

// Verifica se a resposta foi bem-sucedida e tem o tipo de conteúdo
esperado
if (!response.ok) {
    throw new Error(`Erro ${response.status}: ${response.statusText}`);
}

const contentType = response.headers.get("content-type");
if (!contentType || !contentType.includes("application/json")) {
    throw new Error("Resposta não contém JSON válido.");
}

const sugestoes = await response.json();

// Seleciona ou cria o container para exibir as sugestões
let sugestoesContainer = document.querySelector("#sugestoes-container");
if (!sugestoesContainer) {
    sugestoesContainer = document.createElement("div");
    sugestoesContainer.id = "sugestoes-container";
    document.body.appendChild(sugestoesContainer); // Anexa a caixa de
sugestões ao corpo do site
}

// Adiciona cada sugestão como um parágrafo na div
sugestoes.forEach(item => {
    const sugestaoElement = document.createElement("p");
    sugestaoElement.textContent = item.sugestao; // Exibe o texto da
sugestão
    sugestoesContainer.appendChild(sugestaoElement);
});
} catch (error) {
    console.error("Erro ao buscar sugestões:", error);
}
}

function calcularRelatorioQuestionario(acertos, erros){

    if(acertos === 0 && erros === 0){
        return "Nenhuma resposta!"
    }else if(acertos > erros && erros === 0){
        return "Somente respostas certas! Talvez esteja muito fácil!"
    }else if(acertos === 0 && erros > acertos){
        return "Somente respostas erradas! Talvez esteja muito difícil!"
    }else if(acertos == erros){
        return "Respostas Iguais"
    }else if (acertos > erros + 5){
        return "Muitas respostas certas!"
    }else if (erros > acertos + 5){
        return "Muitas respostas erradas!"
    }else if (acertos > erros) {
        return "Algumas respostas certas de diferença!"
    } else if (erros > acertos) {
        return "Algumas respostas erradas de diferença!"
    }else{
        return "Respostas Iguais"
    }
}

function calcularRelatorioAvaliacao(excelente, bom, regular, ruim, pessimo) {
    const totalRespostas = excelente + bom + regular + ruim + pessimo;

    if (totalRespostas === 0) {
        return "Nenhuma resposta!";
    }
}

```

```
}

if (excelente > bom && excelente > regular && excelente > ruim && excelente > pessimo) {
    return "Ótima avaliação! A maioria considera excelente!";
} else if (bom > excelente && bom > regular && bom > ruim && bom > pessimo) {
    return "Boa avaliação! A maioria considera bom!";
} else if (regular > 10) {
    return "Avaliação regular. Pode ser melhorada.";
} else if (ruim > 10 || pessimo > 10) {
    return "Avaliação negativa. Muitos consideram ruim ou péssimo!";
} else {
    return "Avaliação mista. Precisamos de mais feedback.";
}

// Chama as funções para carregar as perguntas e avaliações ao carregar a página
document.addEventListener("DOMContentLoaded", () => {

    fetchPerguntas();
    fetchAvaliacao();
    fetchSugestoes();

});
```

APÊNDICE CC – CODIGO SITE – RELATORIOS – HTML

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Relatórios e Listas</title>
    <link rel="stylesheet" href="style.css">
    <script src="../../apiConfig.js"></script>
    <script src="script.js" defer></script>
</head>
<body>
    <div class="container">
        <!-- Botão Voltar -->
        <button onclick="window.history.back()">Voltar</button>

        <h2 class="container">Relatórios e Listas</h2>

        <div class="container" id="vendas-section">
            <h3>Vendas</h3>
            <div id="pegarVendasBtn" class="tab" onclick="toggleTab('vendas')">
                <span class="tab-title">Pegar Vendas</span>
                <span class="arrow">▼</span>
            </div>
            <div class="tab-content" id="vendas">
                <div id="vendasList"></div>
            </div>

            <div id="pegarVendaBtn" class="tab" onclick="toggleTab('venda')">
                <span class="tab-title">Pegar Venda</span>
                <span class="arrow">▼</span>
            </div>
            <div class="tab-content" id="venda">
                <p>Pegar Venda (ID):<br>
                    <input type="text" id="vendaIdInput" placeholder="Digite o ID" class="input">
                <button id="buscarVendaBtn" class="btn-buscar" onclick="fetchVenda()>Buscar</button>
                </p>
                <div id="vendaResult" class="resultado" style="margin-top: 20px;"><!-- Local para mostrar o resultado -->
            </div>
        </div>
    </div>

    <div class="container" id="ingressos-section">
        <h3>Ingressos</h3>
        <div id="pegarIngressosBtn" class="tab" onclick="toggleTab('ingressos')">
            <span class="tab-title">Pegar Ingressos</span>
            <span class="arrow">▼</span>
        </div>
        <div class="tab-content" id="ingressos">
            <div id="ingressosList"></div>
        </div>

        <div class="tab" onclick="toggleTab('ingresso')">
            <span class="tab-title">Pegar Ingresso</span>
            <span class="arrow">▼</span>
        </div>
    </div>
</body>

```

```

        <div class="tab-content" id="ingresso">
            <p>Pegar Ingresso (KEY):
                <input type="text" id="ingressoKeyInput" placeholder="Digite a
Key de 9 dígitos" class="input" maxlength="9">
                <button id="buscarIngressoBtn" class="btn-buscar"
onclick="fetchIngresso()">Buscar</button>
            </p>
            <div id="ingressoResult" class="resultado" style="margin-top:
20px;"></div>
        </div>

    </div>

    <div class="container" id="relatorios-section">
        <h3>Relatórios</h3>
        <div id="pegarRelatoriosIngressosBtn" class="tab"
onclick="toggleTab('relatoriosIngressos')">
            <span class="tab-title">Pegar Relatórios Ingressos</span>
            <span class="arrow">▼</span>
        </div>
        <div class="tab-content" id="relatoriosIngressos">
            <div id="relatoriosContainer">
                <!-- Os relatórios serão exibidos aqui -->
            </div>
        </div>

        <div class="tab" onclick="toggleTab('relatorioIngressos')">
            <span class="tab-title">Pegar Relatório Ingressos</span>
            <span class="arrow">▼</span>
        </div>
        <div class="tab-content" id="relatorioIngressos">
            <p>Pegar Relatório Ingressos (Data): <input type="date"
id="dataIngresso" class="input" placeholder="Selecione a data"></p>
            <button onclick="fetchRelatorioIngresso()" class="btn-buscar">Enviar
Data</button>
            <div id="relatorioIngressoContainer">
                <!-- O relatório de ingresso será exibido aqui -->
            </div>
        </div>

        <div id="pegarRelatoriosVendasBtn" class="tab"
onclick="toggleTab('relatoriosVendas')">
            <span class="tab-title">Pegar Relatórios Vendas</span>
            <span class="arrow">▼</span>
        </div>
        <div class="tab-content" id="relatoriosVendas">
            <div id="relatoriosVendasContainer">
                <!-- Os relatórios de vendas serão exibidos aqui -->
            </div>
        </div>

        <div class="tab" onclick="toggleTab('relatorioVendas')">
            <span class="tab-title">Pegar Relatório Vendas</span>
            <span class="arrow">▼</span>
        </div>
        <div class="tab-content" id="relatorioVendas">
            <p>Pegar Relatório Vendas (Data): <input type="date" id="dataVenda"
class="input" placeholder="Selecione a data"></p>
            <button onclick="fetchRelatorioVenda()" class="btn-buscar">Enviar
Data</button>
            <div id="relatorioVendaContainer">

```

```
    <!-- O relatório de vendas será exibido aqui -->
</div>
</div>

</div>
</body>
</html>
```

APÊNDICE CD – CODIGO SITE – RELATORIOS – CSS

```

* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    font-family: Arial, sans-serif;
    background-color: #000000;
}

.container {
    width: 90%; /* Aumentando a largura para melhor uso do espaço */
    max-width: 1200px; /* Permitindo mais espaço em telas grandes */
    margin: 50px auto;
    padding: 20px;
    background-color: #000000;
    border-radius: 8px;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
}

.section {
    margin-bottom: 30px; /* Espaço entre as seções */
}

h2 {
    margin-bottom: 20px;
    color: #ffffff;
    text-align: center;
}

h3 {
    margin-bottom: 10px;
    color: #c74444; /* Cor vermelha para títulos das seções */
}

.tabs {
    margin-top: 20px;
}

.tab {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 12px;
    background-color: #c74444;
    color: white;
    cursor: pointer;
    border-radius: 4px;
    margin-bottom: 10px;
    transition: background-color 0.3s, box-shadow 0.3s;
}

.tab:hover {
    background-color: #970000;
    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.2); /* Efeito de sombra ao passar o mouse */
}

.tab-content {

```

```

display: none; /* Esconde as abas inicialmente */
padding: 15px;
background-color: #ffffff;
border: 1px solid #ddd;
border-radius: 4px;
margin-top: 5px;
transition: max-height 0.2s ease-in-out;
max-height: 1200px;
overflow-y: auto;
}

.venda-item {
    margin-bottom: 10px; /* Espaço entre itens de venda */
}

.arrow {
    margin-left: 10px;
    transition: transform 0.3s; /* Efeito de rotação da seta */
}

.tab-content.show {
    display: block;
    max-height: 1000px; /* Altura máxima para a transição */
}

.tab-content.hide {
    max-height: 0; /* Altura mínima para esconder */
}

/* Estilização da textbox */
.input {
    padding: 10px; /* Espaçamento interno */
    border: 2px solid #f03131; /* Borda branca */
    border-radius: 5px; /* Bordas arredondadas */
    font-size: 16px; /* Tamanho da fonte */
    width: 200px; /* Largura fixa */
    transition: border-color 0.3s; /* Efeito de transição na borda */
}

/* Efeito de foco na textbox */
.input:focus {
    border-color: #520a08; /* Cor da borda ao focar */
    outline: none; /* Remove o contorno padrão */
}

/* Estilização do botão de buscar */
.btn-buscar {
    padding: 10px 15px; /* Espaçamento interno */
    background-color: #e63838; /* Cor de fundo */
    color: white; /* Cor do texto */
    border: none; /* Remove borda padrão */
    border-radius: 5px; /* Bordas arredondadas */
    font-size: 16px; /* Tamanho da fonte */
    cursor: pointer; /* Muda o cursor ao passar sobre o botão */
    transition: background-color 0.3s, transform 0.2s; /* Efeito de transição na cor de fundo e transformação */
    margin-left: 10px; /* Margem à esquerda do botão */
}

/* Efeito de hover no botão de buscar */
.btn-buscar:hover {
    background-color: #e92222; /* Cor de fundo ao passar o mouse */
    transform: translateY(-2px); /* Leve movimento para cima */
}

```

```
}

/* Estilização do resultado */
.resultado {
    border: 1px solid #ccc; /* Borda leve */
    border-radius: 5px; /* Bordas arredondadas */
    padding: 10px; /* Espaçamento interno */
    background-color: #f9f9f9; /* Cor de fundo leve */
    margin-top: 10px; /* Margem superior */
}

.relatorio-item {
    border: 1px solid #ccc;
    padding: 10px;
    margin-bottom: 10px;
    border-radius: 5px;
    background-color: #f9f9f9;
}

/* Estilização do botão "Voltar" */
button {
    background-color: #c74444; /* Destaque na cor vermelha */
    color: white;
    border: none;
    border-radius: 5px;
    padding: 10px 20px;
    font-size: 16px;
    cursor: pointer;
    transition: background-color 0.3s ease, transform 0.3s ease;
    margin-bottom: 20px; /* Espaço abaixo do botão */
    display: block;
    margin-left: auto;
    margin-right: auto; /* Centralizar o botão */
}

button:hover {
    background-color: #970000; /* Efeito ao passar o mouse */
    transform: translateY(-2px); /* Leve animação de hover */
}

button:active {
    transform: translateY(1px); /* Efeito ao clicar */
}
```

APÊNDICE CE – CODIGO SITE – RELATORIOS – JAVASCRIPT

```

console.log(localStorage.getItem("apiBase"));
const apiBase = localStorage.getItem("apiBase") || "https://default-api-url.com";

// Função para alternar a exibição das abas
function toggleTab(tabId) {
    const tabContents = document.querySelectorAll('.tab-content');
    const arrow = document.querySelectorAll('.arrow');

    // Fecha todas as abas antes de abrir uma nova
    tabContents.forEach(tab => {
        if (tab.id === tabId) {
            const isActive = tab.style.display === 'block';
            tab.style.display = isActive ? 'none' : 'block'; // Alterna a
            exibição
            tab.classList.toggle('show', !isActive);
            tab.classList.toggle('hide', isActive);

            // Muda a direção da seta
            const index = [...tabContents].indexOf(tab);
            arrow[index].style.transform = isActive ? 'rotate(0deg)' :
            'rotate(180deg)';
        } else {
            tab.style.display = 'none'; // Fecha as outras abas
            tab.classList.remove('show');
            tab.classList.add('hide');

            // Resetar setas das outras abas
            const index = [...tabContents].indexOf(tab);
            arrow[index].style.transform = 'rotate(0deg)';
        }
    });
}

// Função para buscar vendas da API
async function fetchVendas() {
    const apiVenda = apiBase + '/api/Master/vendas'; // URL para buscar as vendas
    try {
        const response = await fetch(apiVenda, {
            method: "GET",
            headers: {
                "Accept": "application/json",
                "ngrok-skip-browser-warning": "true" // Cabeçalho para ignorar o
                aviso do ngrok
            }
        });
        if (!response.ok) {
            throw new Error('Erro ao buscar vendas: ' + response.statusText);
        }
        const contentType = response.headers.get("content-type");
        if (!contentType || !contentType.includes("application/json")) {
            throw new Error("Resposta não contém JSON válido.");
        }
        const vendas = await response.json();
        displayVendas(vendas); // Chama a função para exibir as vendas
    } catch (error) {
        console.error(error);
        document.getElementById('vendasList').innerText = 'Erro ao carregar
        vendas';
    }
}

```

```

// Função para exibir vendas na página
function displayVendas(vendas) {
    const vendasList = document.getElementById('vendasList');
    vendasList.innerHTML = ''; // Limpa a lista antes de adicionar novos itens

    vendas.forEach(venda => {
        const vendaItem = document.createElement('div');
        vendaItem.classList.add('venda-item');
        vendaItem.innerHTML =
            `ID: ${venda.id} <br>
            Email: ${venda.email} <br>
            Quantidade: ${venda.quantidade} <br>
            Método de Pagamento: ${venda.metodoPagamento} <br>
            Preço: R$ ${venda.preco.toFixed(2)} <br>
            <hr>
        `;
        vendasList.appendChild(vendaItem); // Adiciona o item à lista
    });
}

// Adiciona evento ao botão para buscar vendas
document.getElementById('pegarVendasBtn').addEventListener('click', fetchVendas);

async function fetchIngressos() {
    const apiIngressos = apiBase +'/api/Master/ingressos'; // URL para buscar os ingressos
    console.log('Buscando ingressos na URL:', apiIngressos); // Log da URL

    try {
        const response = await fetch(apiIngressos, {
            method: "GET",
            headers: {
                "Accept": "application/json",
                "ngrok-skip-browser-warning": "true" // Cabeçalho para ignorar o aviso do ngrok
            }
        });

        console.log('Status da resposta:', response.status); // Log do status da resposta
        if (!response.ok) {
            throw new Error('Erro ao buscar ingressos: ' + response.statusText);
        }
        const contentType = response.headers.get("content-type");
        if (!contentType || !contentType.includes("application/json")) {
            throw new Error("Resposta não contém JSON válido.");
        }
        const ingressos = await response.json();
        console.log('Ingressos recebidos:', ingressos); // Log dos ingressos recebidos
        displayIngressos(ingressos); // Chama a função para exibir os ingressos
    } catch (error) {
        console.error('Erro:', error);
        document.getElementById('ingressos').innerText = 'Erro ao carregar ingressos'; // Corrigi aqui para usar 'ingressos' e não 'ingressosList'
    }
}

// Função para exibir ingressos na página
function displayIngressos(ingressos) {
    const tabContent = document.getElementById('ingressos');
    tabContent.innerHTML = ''; // Limpa o conteúdo anterior
}

```

```

// Cria uma lista para exibir os ingressos
ingressos.forEach(ingresso => {
    const ingressoElement = document.createElement('div');
    ingressoElement.className = 'ingresso-item'; // Classe para estilização
    ingressoElement.innerHTML =
        `ID: ${ingresso.id}</p>
        Key: ${ingresso.key}</p>
        Nome: ${ingresso.nome}</p>
        Tipo: ${ingresso.tipo}</p>
        <hr>
    `;
    tabContent.appendChild(ingressoElement); // Adiciona o item ao conteúdo
da aba
});
}

// Adiciona evento ao botão para buscar ingressos
document.getElementById('pegarIngressosBtn').addEventListener('click',
fetchIngressos);

async function fetchVenda() {
    const vendaId = document.getElementById('vendaIdInput').value; // Obtém o
valor do input
    const apiVenda = `${apiBase}/api/Master/vendas/${vendaId}`; // URL para
buscar a venda
    console.log('Buscando venda na URL:', apiVenda); // Log da URL

    try {
        const response = await fetch(apiVenda, {
            method: "GET",
            headers: {
                "Accept": "application/json",
                "ngrok-skip-browser-warning": "true" // Cabeçalho para ignorar o
aviso do ngrok
            }
        });

        console.log('Status da resposta:', response.status); // Log do status da
resposta
        if (!response.ok) {
            throw new Error('Erro ao buscar venda: ' + response.statusText);
        }
        const contentType = response.headers.get("content-type");
        if (!contentType || !contentType.includes("application/json")) {
            throw new Error("Resposta não contém JSON válido.");
        }
        const venda = await response.json();
        console.log('Venda recebida:', venda); // Log da venda recebida
        displayVenda(venda); // Chama a função para exibir a venda
    } catch (error) {
        console.error('Erro:', error);
        document.getElementById('vendaResult').innerText = 'Venda não existe!';
    }
}

function displayVenda(venda) {
    const vendaResult = document.getElementById('vendaResult');
    vendaResult.innerHTML = ''; // Limpa o conteúdo anterior

    if (!venda.id) {

```

```

        vendaResult.innerText = 'Venda não encontrada.'; // Mensagem para venda
não encontrada
        return;
    }

// Formata a venda
vendaResult.innerHTML =
    <div style="border: 1px solid #ccc; padding: 10px; border-radius: 5px;
background-color: #f9f9f9;">
        <p><strong>ID:</strong> ${venda.id}</p>
        <p><strong>Email:</strong> ${venda.email}</p>
        <p><strong>Quantidade:</strong> ${venda.quantidade}</p>
        <p><strong>Método de Pagamento:</strong> ${venda.metodoPagamento}</p>
        <p><strong>Preço:</strong> R$ ${venda.preco.toFixed(2)}</p>
    </div>
';

}

document.getElementById('pegarVendaBtn').addEventListener('click', fetchVenda);

// Função para permitir apenas dígitos
function validarCaracteres(event) {
    const charCode = (event.which) ? event.which : event.keyCode;
    // Verifica se o caractere é especial
    if (charCode < 32 || (charCode > 57 && charCode < 65) || (charCode > 90 &&
charCode < 97) || charCode > 122) {
        return false; // Previne a entrada de caracteres especiais
    }
    return true; // Permite a entrada de caracteres alfanuméricos
}

async function fetchIngresso() {
    const ingressoKey = document.getElementById('ingressoKeyInput').value; // Obtém o valor do input

    // Verifica se a chave tem exatamente 9 caracteres
    if (ingressoKey.length !== 9) {
        alert('A chave deve conter exatamente 9 caracteres!'); // Mensagem de erro
        return; // Sai da função se a condição não for atendida
    }

    const apiIngresso = `${apiBase}/api/Master/ingressos/${ingressoKey}`; // URL para buscar o ingresso
    console.log('Buscando ingresso na URL:', apiIngresso); // Log da URL

    try {
        const response = await fetch(apiIngresso, {
            method: "GET",
            headers: {
                "Accept": "application/json",
                "ngrok-skip-browser-warning": "true" // Cabeçalho para ignorar o aviso do ngrok
            }
        });

        console.log('Status da resposta:', response.status); // Log do status da resposta
        if (!response.ok) {
            throw new Error('Erro ao buscar ingresso: ' + response.statusText);
        }
    } catch (error) {
        console.error('Ocorreu um erro ao buscar o ingresso:', error);
    }
}

```

```

        }
        const contentType = response.headers.get("content-type");
        if (!contentType || !contentType.includes("application/json")) {
            throw new Error("Resposta não contém JSON válido.");
        }
        const ingresso = await response.json();
        console.log('Ingresso recebido:', ingresso); // Log do ingresso recebido
        displayIngresso(ingresso); // Chama a função para exibir o ingresso
    } catch (error) {
        console.error('Erro:', error);
        document.getElementById('ingressoResult').innerText = 'Ingresso não encontrado!';
    }
}

function displayIngresso(ingresso) {
    const ingressoResult = document.getElementById('ingressoResult');
    ingressoResult.innerHTML = ''; // Limpa o conteúdo anterior

    if (!ingresso.id) {
        ingressoResult.innerText = 'Ingresso não encontrado.'; // Mensagem para
        ingresso não encontrado
        return;
    }

    // Formata o ingresso
    ingressoResult.innerHTML =
        `<div style="border: 1px solid #ccc; padding: 10px; border-radius: 5px;
background-color: #f9f9f9;">
            <p><strong>ID:</strong> ${ingresso.id}</p>
            <p><strong>Key:</strong> ${ingresso.key}</p>
            <p><strong>Nome:</strong> ${ingresso.nome}</p>
            <p><strong>Tipo:</strong> ${ingresso.tipo}</p>
        </div>
`;
}
}

// Adiciona o evento de validação para permitir caracteres não especiais
document.getElementById('ingressoKeyInput').addEventListener('keypress',
    validarCaracteres);
document.getElementById('buscarIngressoBtn').addEventListener('click',
    fetchIngresso);

async function fetchRelatoriosIngressos() {
    const url = apiBase + '/api/Master/relatorioIngressos'; // Substitua pelo seu
    endpoint real

    fetch(url, {
        method: "GET",
        headers: {
            "Accept": "application/json",
            "ngrok-skip-browser-warning": "true" // Cabeçalho para ignorar o
aviso do ngrok
        }
    })
        .then(response => {
            if (!response.ok) {
                throw new Error('Erro ao buscar relatórios');
            }

            const contentType = response.headers.get("content-type");
            if (!contentType || !contentType.includes("application/json")) {
                throw new Error("Resposta não contém JSON válido.");
            }
        })
}

```

```

        }

        return response.json();
    })
    .then(data => {
        // Limpa o container antes de adicionar os novos relatórios
        const relatoriosContainer =
document.getElementById('relatoriosContainer');
        relatoriosContainer.innerHTML = '';

        // Verifica se há relatórios
        if (data.length === 0) {
            relatoriosContainer.innerHTML = '<p>Nenhum relatório
encontrado.</p>';
            return;
        }

        // Cria uma estrutura para exibir cada relatório
        data.forEach(relatorio => {
            const relatorioDiv = document.createElement('div');
            relatorioDiv.classList.add('relatorio-item');
            relatorioDiv.innerHTML =
                `<p><strong>ID:</strong> ${relatorio.id}</p>
                <p><strong>Data do Relatório:</strong>
${relatorio.relatorioData}</p>
                <p><strong>Total de Ingressos Vendidos:</strong>
${relatorio.totalIngressosVendidos}</p>
                <p><strong>Total de Ingressos Inteiros:</strong>
${relatorio.totalIngressosInteiro}</p>
                <p><strong>Total de Ingressos Meia:</strong>
${relatorio.totalIngressosMeia}</p>
                <p><strong>Total de Ingressos Isentos:</strong>
${relatorio.totalIngressosIsentos}</p>
                `;
            relatoriosContainer.appendChild(relatorioDiv);
        });
    })
    .catch(error => {
        console.error('Erro:', error);
        const relatoriosContainer =
document.getElementById('relatoriosContainer');
        relatoriosContainer.innerHTML = '<p>Erro ao buscar relatórios. Tente
novamente mais tarde.</p>';
    });
}

document.getElementById('pegarRelatoriosIngressosBtn').addEventListener('click',
fetchRelatoriosIngressos);

async function fetchRelatoriosVendas() {
    const url = `${apiBase}/api/Master/relatorioVendas`;

    fetch(url, {
        method: "GET",
        headers: {
            "Accept": "application/json",
            "ngrok-skip-browser-warning": "true" // Cabeçalho para ignorar o
aviso do ngrok
        }
    })
    .then(response => {
        if (!response.ok) {
            throw new Error('Erro ao buscar relatórios de vendas');
        }
    })
}

```

```

        }

        const contentType = response.headers.get("content-type");
        if (!contentType || !contentType.includes("application/json")) {
            throw new Error("Resposta não contém JSON válido.");
        }

        return response.json();
    })
    .then(data => {
        // Limpa o container antes de adicionar os novos relatórios
        const relatoriosVendasContainer =
document.getElementById('relatoriosVendasContainer');
        relatoriosVendasContainer.innerHTML = '';

        // Verifica se há relatórios
        if (data.length === 0) {
            relatoriosVendasContainer.innerHTML = '<p>Nenhum relatório de vendas
encontrado.</p>';
            return;
        }

        // Exibe cada relatório de vendas
        data.forEach(relatorio => {
            const relatorioDiv = document.createElement('div');
            relatorioDiv.classList.add('relatorio-item');
            relatorioDiv.innerHTML =
                `<p><strong>ID:</strong> ${relatorio.id}</p>
                <p><strong>Data do Relatório:</strong>
${relatorio.relatorioData}</p>
                <p><strong>Dinheiro Total:</strong> R$ ${relatorio.dinheiroTotal}</p>
                <p><strong>Pagamentos no Crédito:</strong>
${relatorio.quantidadePagamentoCredito}</p>
                <p><strong>Pagamentos no Débito:</strong>
${relatorio.quantidadePagamentoDebito}</p>
                <p><strong>Pagamentos no Pix:</strong>
${relatorio.quantidadePagamentoPix}</p>
                `;
            relatoriosVendasContainer.appendChild(relatorioDiv);
        });
    })
    .catch(error => {
        console.error('Erro:', error);
        const relatoriosVendasContainer =
document.getElementById('relatoriosVendasContainer');
        relatoriosVendasContainer.innerHTML = '<p>Erro ao buscar relatórios de
vendas. Tente novamente mais tarde.</p>';
    });
}

document.getElementById('pegarRelatoriosVendasBtn').addEventListener('click',
fetchRelatoriosVendas);

function fetchRelatorioIngresso() {
    const dataInput = document.getElementById('dataIngresso').value;

    if (!dataInput) {
        alert('Por favor, selecione uma data.');
        return;
    }
}

```

```

// Extraiendo ano, mês e dia e formatando como anoMesDia
const [ano, mes, dia] = dataInput.split('-');
const anoMesDia = `${ano}${mes}${dia}`;

const url = `${apiBase}/api/Master/relatorioIngressos/${anoMesDia}`;

fetch(url, {
    method: 'GET',
    headers: {
        'Accept': 'application/json',
        'ngrok-skip-browser-warning': 'true' // Cabeçalho para integração com
ngrok
    }
})
.then(response => {
    if (!response.ok) {
        throw new Error('Erro ao buscar relatório de ingresso');
    }

    const contentType = response.headers.get('content-type');
    if (!contentType || !contentType.includes('application/json')) {
        throw new Error('A resposta não contém JSON válido.');
    }

    return response.json();
})
.then(data => {
    const relatorioIngressoContainer =
document.getElementById('relatorioIngressoContainer');
    relatorioIngressoContainer.innerHTML = '';

    // Exibindo dados do relatório
if (data) {
        relatorioIngressoContainer.innerHTML = `
            <div class="relatorio-item">
                <p><strong>ID:</strong> ${data.id}</p>
                <p><strong>Data do Relatório:</strong>
${data.relatorioData}</p>
                <p><strong>Total de Ingressos Vendidos:</strong>
${data.totalIngressosVendidos}</p>
                <p><strong>Ingressos Inteiros:</strong>
${data.totalIngressosInteiro}</p>
                <p><strong>Ingressos Meia:</strong>
${data.totalIngressosMeia}</p>
                <p><strong>Ingressos Isentos:</strong>
${data.totalIngressosIsentos}</p>
            </div>
        `;
    } else {
        relatorioIngressoContainer.innerHTML = '<p>Relatório de ingresso
não encontrado para a data selecionada.</p>';
    }
})
.catch(error => {
    console.error('Erro:', error);
    const relatorioIngressoContainer =
document.getElementById('relatorioIngressoContainer');
    relatorioIngressoContainer.innerHTML = '<p>Relatório não existe ou
ocorreu um erro!.</p>';
});
}

function fetchRelatorioVenda() {

```

```

const dataInput = document.getElementById('dataVenda').value;

if (!dataInput) {
    alert('Por favor, selecione uma data.');
    return;
}

// Extraiendo ano, mês e dia e formatando como anoMesDia
const [ano, mes, dia] = dataInput.split('-');
const anoMesDia = `${ano}${mes}${dia}`;

const url = `${apiBase}/api/Master/relatorioVendas/${anoMesDia}`;

fetch(url, {
    method: 'GET',
    headers: {
        'Accept': 'application/json',
        'ngrok-skip-browser-warning': 'true' // Cabeçalho para integração com
ngrok
    }
})
.then(response => {
    if (!response.ok) {
        throw new Error('Erro ao buscar relatório de venda');
    }
    const contentType = response.headers.get('content-type');
    if (!contentType || !contentType.includes('application/json')) {
        throw new Error('A resposta não contém JSON válido.');
    }
    return response.json();
})
.then(data => {
    const relatorioVendaContainer =
document.getElementById('relatorioVendaContainer');
    relatorioVendaContainer.innerHTML = '';

    // Exibindo dados do relatório de venda
    if (data) {
        relatorioVendaContainer.innerHTML = `
            <div class="relatorio-item">
                <p><strong>ID:</strong> ${data.id}</p>
                <p><strong>Data do Relatório:</strong>
${data.relatorioData}</p>
                <p><strong>Dinheiro Total:</strong> R$ ${data.dinheiroTotal}</p>
                <p><strong>Pagamentos em Crédito:</strong>
${data.quantidadePagamentoCredito}</p>
                <p><strong>Pagamentos em Débito:</strong>
${data.quantidadePagamentoDebito}</p>
                <p><strong>Pagamentos via Pix:</strong>
${data.quantidadePagamentoPix}</p>
            </div>
        `;
    } else {
        relatorioVendaContainer.innerHTML = '<p>Relatório de venda não
encontrado para a data selecionada.</p>';
    }
})
.catch(error => {
    console.error('Erro:', error);
    const relatorioVendaContainer =
document.getElementById('relatorioVendaContainer');
}
);

```

```
    relatorioVendaContainer.innerHTML = '<p>Erro ao buscar o relatório de  
venda. Tente novamente mais tarde.</p>';  
});  
}
```

APÊNDICE CF – CODIGO SITE – EXPOSICAO – HTML

```

<!DOCTYPE html>
<html lang="pt-br">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Exposição</title>
    <link rel="stylesheet" href="Pim/Home/Exposição_1/exposição.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css">
</head>

<body>

    <header>
        <nav class="nav-bar">
            <div class="logo">
                <h1>MECH</h1>
            </div>
            <div class="nav-list">
                <ul>
                    <li class="nav-item"><a href="home.html" class="nav-link">Página Principal</a></li>
                    <li class="nav-item"><a href="informações.html" class="nav-link">Informações</a></li>
                    <li class="nav-item"><a href="exposição.html" class="nav-link"> Exposições</a></li>
                    <li class="nav-item"><a href="compras.html" class="nav-link"> Compras</a></li>
                    <li class="nav-item"><a href="questionario.html" class="nav-link"> Questionario/Avaliação</a></li>
                </ul>
            </div>
            <div class="login-button">
                <button id="loginBtn" class="login"><a href="login.html" id="loginLink">Entrar</a></button>
                <button id="logoutBtn" class="logout" onclick="fazerLogout()">Logout</button>
            </div>
            <div class="mobile-menu-icon">
                <button onclick="menuShow()"></button>
            </div>
        </nav>
        <div class="mobile-menu">
            <ul>
                <li class="nav-item"><a href="home.html" class="nav-link">Página Principal</a></li>
                <li class="nav-item"><a href="informações.html" class="nav-link">Informações</a></li>
                <li class="nav-item"><a href="exposição.html" class="nav-link"> Exposições</a></li>
                <li class="nav-item"><a href="compras.html" class="nav-link"> Compras</a></li>
                <li class="nav-item"><a href="questionario.html" class="nav-link"> Questionario/Avaliação</a></li>
            </ul>
        </div>
    </header>

```

```

        <div class="login-button">
            <button id="loginBtn" class="login"><a href="login.html"
id="loginLink">Entrar</a></button>
            <button id="logoutBtn" class="logout"
onclick="fazerLogout()">Logout</button>
        </div>
    </div>
</header>
<!-- Fim do cabeçalho --&gt;

&lt;div class="slider"&gt;

    &lt;div class="list"&gt;

        &lt;!-- Primeiro item do slider --&gt;
        &lt;div class="item active"&gt;
            &lt;img src="Pim/Home/Exposição 1/src/Colonizacao de Marte.jpg"&gt;
            &lt;div class="content"&gt;
                &lt;Div class="titulo"&gt;
                    &lt;p&gt;Colonização de Marte&lt;/p&gt;
                &lt;/Div&gt;

                &lt;p&gt;
                    A colonização de Marte é um objetivo ambicioso que tem
capturado a imaginação de científicos, engenheiros e visionários por décadas.
                &lt;/p&gt;
                &lt;button class="btn-inf"&gt; &lt;a href="#Colonização"&gt;Saiba
Mais&lt;/a&gt; &lt;/button&gt;
            &lt;/div&gt;
        &lt;/div&gt;
        &lt;!-- Segundo item do slider --&gt;
        &lt;div class="item"&gt;
            &lt;img src="Pim/Home/Exposição 1/src/agua-em-marte.png"&gt;
            &lt;div class="content"&gt;
                &lt;Div class="titulo"&gt;
                    &lt;p&gt;Água em Marte&lt;/p&gt;
                &lt;/Div&gt;

                &lt;p&gt;
                    A possibilidade de vida em Marte é um tema intrigante e
complexo.
                &lt;/p&gt;
                &lt;button class="btn-inf"&gt;&lt;a href="#Agua"&gt;Mais
informações&lt;/a&gt;&lt;/button&gt;
            &lt;/div&gt;
        &lt;/div&gt;

        &lt;!-- Terceiro item do slider --&gt;
        &lt;div class="item"&gt;
            &lt;img src="Pim/Home/Exposição 1/src/Exploracao e Potencial para
Vida.png"&gt;
            &lt;div class="content"&gt;
                &lt;Div class="titulo"&gt;
                    &lt;p&gt;Exploração e Potencial para Vida&lt;/p&gt;
                &lt;/Div&gt;

                &lt;p&gt;
                    A exploração de Marte é fundamental para desvendar os
segredos do sistema solar.
                &lt;/p&gt;
            &lt;/div&gt;
        &lt;/div&gt;
    &lt;/div&gt;
&lt;/div&gt;
</pre>

```

```

        <button class="btn-inf"><a href="#Exploração">Mais
informações</a></button>
        </div>
    </div>

<!-- Quarto item do slider -->
<div class="item">
    
    <div class="content">
        <Div class="titulo">
            <p>Impacto de Asteroides</p>
        </Div>

        <p>
            Marte, o Planeta Vermelho, é um testemunho vivo da
violência cósmica que marca a história do sistema solar.
        </p>
        <button class="btn-inf"><a href="#asteroides">Mais
informações</a></button>
        </div>
    </div>
<!-- Quinto item do slider -->
<div class="item">
    
    <div class="content">
        <Div class="titulo">
            <p>O monte Olimpo</p>
        </Div>

        <p>
            Atualmente robôs pioneiros como o Curiosity Rover,
Perseverance Rover, InSight Lander e ExoMars Rover, foram enviados em missão para
marte.
        </p>
        <button class="btn-inf"><a href="#olimpico">Mais
informações</a></button>
        </div>
    </div>
<!-- Sexto item do slider -->
<div class="item">
    
    <div class="content">
        <Div class="titulo">
            <p>O Planeta Vermelho</p>
        </Div>

        <p>
            Marte é conhecido por sua característica cor vermelha, um
traço distintivo que fascina cientistas e astrônomos.
        </p>
        <button class="btn-inf"><a href="#planeta">Mais
informações</a></button>
        </div>
    </div>
<!-- Sétimo item do slider -->
<div class="item">
    
    <div class="content">
        <Div class="titulo">
            <p>O Terreno Marciano</p>
        </Div>

        <p>

```

Marte apresenta uma paisagem diversificada, marcada por vales profundos, crateras gigantescas e vulcões imponentes.

```

        </p>
        <button class="btn-inf"><a href="#terreno">Mais
informações</a></button>
        </div>
    </div>

        <!-- Oitavo item do slider -->
    <div class="item">
        
        <div class="content">
            <Div class="titulo">
                <p>Valles Marineris</p>
            </Div>

            <p>
                Valles Marineris, o maior canyon do sistema solar, é um
verdadeiro marco natural em Marte.
            </p>
            <button class="btn-inf"><a href="#valles">Mais
informações</a></button>
            </div>
        </div>

        </div>
        <!-- Fim da lista de itens do slider -->
    </div>
    <!-- Fim do slider -->

```

```

<div class="arrows">
    <!-- Div para botões de navegação (setas) -->
    <button id="prev"><</button>
    <!-- Botão para o item anterior -->
    <button id="next">></button>
    <!-- Botão para o próximo item -->
</div>
<!-- Fim da div para botões de navegação -->

```

```

<div class="thumbnail">
    <!-- Div para as miniaturas de imagens -->

    <div class="item active">
        
    </div>

    <div class="item">
        
    </div>

    <div class="item">

```

```

        
    </div>

    <div class="item">
        
    </div>
</div>
<!-- Fim da div para miniaturas --&gt;

&lt;div class="Obras"&gt;
    <!-- Div principal que contém todas as obras da exposição --&gt;

    &lt;div class="Exposição" id="Colonização"&gt;
        &lt;div class="teste"&gt;
            &lt;img src="Pim/Home/Exposição 1/src/Colonizacao de Marte.jpg"
alt="" class="expo"&gt;
            &lt;div class="texto"&gt;
                &lt;h2&gt;Colonização de Marte&lt;/h2&gt;
                &lt;p&gt;A colonização de Marte é um objetivo ambicioso que tem
capturado a imaginação de científicos, engenheiros e visionários por décadas. A
NASA e outras agências espaciais já têm planos concretos para enviar missões
tripuladas a Marte
                    na década de 2030, com a SpaceX, fundada por Elon Musk,
                    trabalhando arduamente para desenvolver tecnologias necessárias para uma
                    colonização sustentável. Com a tecnologia avançando rapidamente, a possibilidade
                    de estabelecer uma
                    comunidade humana no Planeta Vermelho está se tornando
                    cada vez mais realista. No entanto, a colonização de Marte não será fácil.&lt;/p&gt;
            &lt;/div&gt;
        &lt;/div&gt;
    &lt;/div&gt;

    &lt;div class="Exposição" id="Agua"&gt;
        &lt;div class="teste"&gt;
            &lt;img src="Pim/Home/Exposição 1/src/agua-em-marte.png" alt=""
class="expo"&gt;
</pre>

```

```

<div class="texto">
    <h2>Água em Marte</h2>
    <p>A possibilidade de vida em Marte é um tema intrigante e complexo. A água é essencial para a vida como conhecemos, e sua presença é um fator crítico para o desenvolvimento de organismos vivos. A NASA e outras agências espaciais já encontraram evidências de que Marte teve água líquida no passado, como rios e lagos secos, deltas de rios e minerais hidratados. Essas descobertas sugerem que o planeta pode ter tido condições favoráveis à vida no passado.

        </p>
    </div>
</div>

<div class="Exposição" id="Exploração">
    <div class="teste">
        
        <div class="texto">
            <h2>Exploração e Potencial para Vida</h2>
            <p>A exploração de Marte é fundamental para desvendar os segredos do sistema solar. Este planeta vermelho, com sua superfície árida e paisagens deslumbrantes, é um alvo fascinante para cientistas e engenheiros. Atualmente, robôs pioneiros

                como o Curiosity Rover, Perseverance Rover, InSight Lander e ExoMars Rover estão explorando Marte, coletando dados sobre sua geologia, composição do solo, clima, atmosfera e potencial para água líquida.
            </p>
        </div>
    </div>
</div>

<div class="Exposição" id="asteroides">
    <div class="teste">
        
        <div class="texto">
            <h2>Impacto de Asteroides</h2>
            <p>Marte, o Planeta Vermelho, é um testemunho vivo da violência cósmica que marca a história do sistema solar. Sua superfície é pontuada por inúmeras crateras de impacto, resultado da colisão com asteroides e cometas ao longo de milhões de anos, variando em tamanho desde pequenas depressões até gigantescas cavidades como a Cratera Hellas, com mais de 2.200 km de diâmetro. Esses impactos fornecem informações valiosas sobre a composição e estrutura geológica de Marte, além de sugerir que a água pode ter sido trazida por cometas e asteroides.
            </p>
        </div>
    </div>
</div>

<div class="Exposição" id="olimpo">
    <div class="teste">
        
        <div class="texto">
            <h2>O Monte Olimpo</h2>
            <p>O Monte Olimpo, localizado em Marte, é um verdadeiro gigante, com seus impressionantes 27 km de altura, superando o Monte Everest em
        </div>
    </div>
</div>
```

cerca de três vezes. Explorar esse vulcão seria um desafio emocionante e complexo. Para explorar o

Monte Olimpo, seria necessário desenvolver tecnologias avançadas e estratégias inovadoras. Em primeiro lugar, missões robóticas seriam essenciais para mapear a superfície e coletar dados sobre a geologia e composição do vulcão.

Isso ajudaria a identificar áreas de interesse e riscos potenciais.

```

        </p>
        </div>
    </div>
</div>

<div class="Exposição" id="planeta">
    <div class="teste">
        
            <div class="texto" id="planeta">
                <h2>O Planeta Vermelho</h2>
                <p>Marte é conhecido por sua característica cor vermelha, um traço distintivo que fascina cientistas e astrônomos. Essa tonalidade é resultado da presença de óxido de ferro em sua superfície, um produto da oxidação intensa do ferro presente na terra marciana. Em comparação com a Terra, a oxidação em Marte é muito mais acentuada, conferindo ao planeta sua aparência única e deslumbrante.</p>
            </div>
        </div>
    </div>

    <div class="Exposição" id="terreno">
        <div class="teste">
            
                <div class="texto">
                    <h2>O Terreno Marciano</h2>
                    <p>Marte apresenta uma paisagem diversificada, marcada por vales profundos, crateras gigantescas e vulcões imponentes. O mais notável deles é o Olympus Mons, o maior vulcão do sistema solar, com uma altura impressionante de 27 km. A superfície de Marte é dividida em duas regiões principais: a planície setentrional, caracterizada por suas amplas extensões planas, e a região montanhosa meridional, repleta de montanhas e vales.
                </p>
            </div>
        </div>
    </div>

    <div class="Exposição" id="valles">
        <div class="teste">
            
                <div class="texto">
                    <h2>Valles Marineris</h2>
                    <p>Valles Marineris, o maior canyon do sistema solar, é um verdadeiro marco natural em Marte, estendendo-se por uma impressionante distância de 4.000 km. Para colocar essa magnitude em perspectiva, é quatro vezes mais longo do que o Grand Canyon, um dos mais icônicos monumentos naturais da Terra. Sua profundidade e vastidão são um testemunho da força erosiva do vento e da água que, no passado, fluíram em Marte.
                </p>
            </div>
        </div>
    </div>

```

```
        </div>
    </div>
</div>
</div>
<!-- Fim da div principal "Obras" --&gt;
&lt;script src="auth.js"&gt;&lt;/script&gt;
&lt;script src="Pim/Home/Exposição_1/exposição.js"&gt;&lt;/script&gt;
<!-- Inclusão do script JavaScript para a funcionalidade da página --&gt;
&lt;script src="https://kit.fontawesome.com/16ffb79ef8.js"
crossorigin="anonymous"&gt;&lt;/script&gt;
&lt;/body&gt;

&lt;/html&gt;</pre>
```

APÊNDICE CG – CODIGO SITE – EXPOSICAO – CSS

```
* {
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    outline: none;
    border: none;
    text-decoration: none;
    text-transform: capitalize;
    transition: .2s linear;
}

body {
    height: 100vh;

    background-size: cover;
    background-color: #010101;
    color: #eee;
}

/* Começo-Header */
header {
    background-color: #444;
}

.nav-bar {
    display: flex;
    justify-content: space-between;
    padding: 1.5rem 6rem;
}

.logo {
    display: flex;
    align-items: center;
}

.logo h1 {
    color: #fff;
}

.nav-list {
    display: flex;
    align-items: center;
}

.nav-list ul {
    display: flex;
    justify-content: center;
    list-style: none;
}

.nav-item {
    margin: 0 35px;
```

```
}

.nav-link {
    text-decoration: none;
    font-size: 1.15rem;
    color: #fff;
    font-weight: 400;
}

.login-button button {
    border: none;
    padding: 10px 15px;
    border-radius: 5px;
    background-color: #970000;
}

.login-button button a {
    text-decoration: none;
    color: #fff;
    font-weight: 500;
    font-size: 1.1rem;
}

.login-button button.login {
    background-color: #970000; /* Cor de fundo do botão de login */
    color: #fff;
    border: none;
    padding: 10px 20px;
    cursor: pointer;
}

/* Estilos para o botão de logout */
.login-button button.logout {
    color: #fff;
    border: 1px solid #970000;
    padding: 10px 20px;
    cursor: pointer;
    display: none; /* Esconde o botão de logout por padrão */
}

.login-button button:empty {
    display: none;
}

.mobile-menu-icon {
    display: none;
}

.mobile-menu {
    display: none;
}

/* Fim-header */

/* Começo slider */

.slider {
    height: 100vh;
    position: relative;
}

.slider .list .item {
    position: absolute;
```

```
        inset: 0 0 0 0;
        overflow: hidden;
        opacity: 0;
        transition: .10s;
    }

.slider .list .item img {
    width: 100%;
    height: 100%;
    object-fit: cover;
}

.slider .list .item::after {
    content: '';
    width: 100%;
    height: 100%;
    position: absolute;
    left: 0;
    bottom: 0;
    background-image: linear-gradient( to top, #000 20%, transparent);
}

.slider .list .item .content {
    position: absolute;
    left: 22%;
    top: 15%;
    width: 500px;
    max-width: 80%;
    z-index: 1;
}

.slider .list .item .content p:nth-child(1) {
    text-transform: uppercase;
    letter-spacing: 10px;
}

.slider .list .item.active {
    opacity: 1;
    z-index: 10;
}

.slider .list .item.active p {
    font-size: 20px;
}

.btn-inf {
    margin-top: 30px;
    background-color: #000;
    font-size: 15px;
    border: 3px solid #000000;
    border-radius: 50px;
    margin-right: 30px;
}

.btn-inf a {
    color: #fff;
}

/* inicio-MovimentaçãoThumbnail */

@keyframes showContent {
    to {
        transform: translateY(0);
    }
}
```

```
        filter: blur(0);
        opacity: 1;
    }
}

.slider .list .item.active p:nth-child(1) {
    transform: translateY(30px);
    filter: blur(20px);
    opacity: 0;
    animation: showContent .4s .9s ease-in-out 1 forwards;
}

/* Fim-MovimentaçãoThumbnail */

/* Inicio-Setas */

.arrows {
    top: 60%;
    position: absolute;
    right: 878px;
    z-index: 100;
}

.arrows button {
    border: none;
    width: 45px;
    height: 45px;
    border-radius: 5px;
    font-size: x-large;
    background-color: #ccc;
    color: #000000;
    transition: .5s;
}

.arrows button:hover {
    background-color: #ccc;
    color: black;
}

/* Fim-Setas */

/* Inicio-Thumbnail */

.thumbnail {
    position: absolute;
    bottom: 55px;
    z-index: 15;
    display: flex;
    gap: 15px;
    width: 100%;
    height: 250px;
    padding: 0 50px;
    box-sizing: border-box;
    overflow: auto;
    justify-content: center;
}

.thumbnail::-webkit-scrollbar {
    width: 0;
}

.thumbnail .item {
    width: 160px;
```

```
height: 220px;
filter: brightness(.5);
transition: .2s;
flex-shrink: 0;
}

.thumbnail .item img {
    width: 100%;
    height: 100%;
    object-fit: cover;
    border-radius: 15px;
}

.thumbnail .item.active {
    filter: brightness();
}

.thumbnail .item .content {
    position: absolute;
    inset: auto 10px 10px 10px;
}

/* Fim-Slider */

/* Fim-Thumbnail */

/* Inicio-Exposições */

.Exposição {
    padding: 90px 4%;
    border-bottom: 1px solid rgba(255, 255, 255, .3);
    display: flex;
    justify-items: center;
    align-items: center;
    min-height: 100vh;
    margin: 0;
    padding: 0;
}

.teste {
    margin: 150px auto;
    width: 80%;
}

.teste .expo {
    max-width: 700px;
    float: left;
    border: 3PX solid #ccc;
    border-radius: 50px;
    margin-right: 30px;
}

.Obras .texto h2 {
    font-size: 50px;
    color: rgb(255, 255, 255);
}

.Obras .texto P {
    width: 100%;
    font-size: 25PX;
    color: rgb(255, 255, 255);
}
```

```
/* Fim-Exposições */

@media screen and (max-width: 440px) {
    body {
        overflow-y: scroll;
    }
    .nav-bar {
        padding: 1.5rem 4rem;
    }
    .nav-item {
        display: none;
    }
    .login-button {
        display: none;
    }
    .mobile-menu-icon {
        display: block;
    }
    .mobile-menu-icon button {
        background-color: transparent;
        border: none;
        cursor: pointer;
    }
    .mobile-menu ul {
        display: flex;
        flex-direction: column;
        text-align: center;
        padding-bottom: 1rem;
    }
    .mobile-menu .nav-item {
        display: block;
        padding-top: 1.2rem;
    }
    .mobile-menu .login-button {
        display: block;
        padding: 1rem 2rem;
    }
    .mobile-menu .login-button button {
        width: 100%;
    }
    .open {
        display: block;
    }
    .slider .list .item .content {
        position: absolute;
        left: 15%;
        top: 25%;
        width: 450px;
    }
    .slider .list .item .content .titulo {
        align-items: center;
    }
    .btn-inf {
        width: 100%;
    }
    .slider .list .item.active p {
        font-size: 17px;
    }
    .slider .list .item img {
```

```
width: 100%;  
height: 100%;  
object-fit: cover;  
}  
  
.thumbnail {  
    display: none;  
}  
.teste .expo {  
    max-width: 100%;  
    border-radius: 50px;  
}  
.Obras .texto h2{  
font-size: 30px;  
}  
.Obras .texto P {  
    font-size: 20px;  
}  
  
@media screen and (max-width:360px) {  
    .header .navbar a,  
    .header .icons div {  
        font-size: 1.5rem;  
        margin-left: 1.2rem;  
    }  
  
.Obras .texto P {  
    font-size: 18px;  
}  
}  
}
```

APÊNDICE CH – CODIGO SITE – EXPOSICAO – JAVASCRIPT

```

// Função para mostrar/ocultar o menu
function menuShow() {
    let menuMobile = document.querySelector('.mobile-menu');
    let menuIcon = document.querySelector('.icon');

    // Verifica se o menu está aberto ou fechado e alterna o estado
    if (menuMobile.classList.contains('open')) {
        menuMobile.classList.remove('open'); // Fecha o menu
        menuIcon.src = "assets/img/menu_white_36dp.svg"; // Troca o ícone para
        "abrir"

        // Resetando a altura para 0 após o fechamento
        setTimeout(() => {
            menuMobile.style.height = '0'; // A altura do menu é definida como 0
            // Atualiza a página após o fechamento do menu
            window.location.reload(); // Recarrega a página
        }, 300); // Aguardar a transição de fechamento antes de resetar a altura
    } else {
        menuMobile.classList.add('open'); // Abre o menu
        menuIcon.src = "assets/img/close_white_36dp.svg"; // Troca o ícone para
        "fechar"

        // Ajusta a altura para auto para que o menu ocupe o espaço necessário
        menuMobile.style.height = 'auto';
    }
}

// Função para fechar o menu ao clicar fora (e resetar)
window.addEventListener('click', function (event) {
    let menuMobile = document.querySelector('.mobile-menu');
    let menuIconButton = document.querySelector('.mobile-menu-icon button');

    // Verifica se o clique foi fora do menu e do ícone do menu
    if (!menuMobile.contains(event.target) &&
        !menuIconButton.contains(event.target)) {
        if (menuMobile.classList.contains('open')) {
            menuMobile.classList.remove('open'); // Fecha o menu
            document.querySelector('.icon').src =
            "assets/img/menu_white_36dp.svg"; // Troca o ícone para "abrir"

            // Resetando a altura para 0 após o fechamento
            setTimeout(() => {
                menuMobile.style.height = '0'; // A altura do menu é definida
                como 0
                // Atualiza a página após o fechamento do menu
                window.location.reload(); // Recarrega a página
            }, 300); // Aguardar a transição de fechamento antes de resetar a
            altura
        }
    }
});

// Impede que o clique no ícone do menu feche o menu ao clicar nele
document.querySelector('.mobile-menu-icon button').addEventListener('click',
function(event) {
    event.stopPropagation(); // Impede que o clique no ícone seja tratado como
    um clique fora do menu
});

let items = document.querySelectorAll('.slider .list .item');

```

```

let next = document.getElementById('next');
let prev = document.getElementById('prev');
let thumbnails = document.querySelectorAll('.thumbnail .item');

/* Configuração de parametro */
let countItem = items.length;
let itemActive = 0;
/* Avançar */
next.onclick = function() {
    itemActive = itemActive + 1;
    if (itemActive >= countItem) {
        itemActive = 0;
    }
    showSlider();
}
/* Voltar */
prev.onclick = function() {
    itemActive = itemActive - 1;
    if (itemActive < 0) {
        itemActive = countItem - 1;
    }
    showSlider();
}
/* Passar slide automarico */
let refreshInterval = setInterval(() => {
    next.click();
}, 8000)

function showSlider() {
    /* remover antigo novo item */
    let itemActiveOld = document.querySelector('.slider .list .item.active');
    let thumbnailActiveOld = document.querySelector('.thumbnail .item.active');
    itemActiveOld.classList.remove('active');
    thumbnailActiveOld.classList.remove('active');

    /* Ativar novo item */
    items[itemActive].classList.add('active');
    thumbnails[itemActive].classList.add('active');
    setPositionThumbnail();

    /* limpa o controle deslizante de tempo automático */
    clearInterval(refreshInterval);
    refreshInterval = setInterval(() => {
        next.click();
    }, 5000)
}

function setPositionThumbnail() {
    let thumbnailActive = document.querySelector('.thumbnail .item.active');
    let rect = thumbnailActive.getBoundingClientRect();
    if (rect.left < 0 || rect.right > window.innerWidth) {
        thumbnailActive.scrollIntoView({ behavior: 'smooth', inline: 'nearest' });
    }
}

/* Interação com Thumbamil */
thumbnails.forEach((thumbnail, index) => {
    thumbnail.addEventListener('click', () => {
        itemActive = index;
        showSlider();
    })
})

```


APÊNDICE CI – CODIGO SITE – QUESTIONARIO – HTML

```

<!DOCTYPE html>
<html lang="pt-br">
    <head>
        <meta charset="UTF-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <title>Questionário e Avaliação</title>
        <script src="apiConfig.js"></script>
        <script src="app.js" defer></script>
        <!-- Carrega o script JavaScript após o carregamento da página -->
        <link rel="stylesheet" href="Pim/Home/Questionario/questionario.css" />
        <!-- Link para o arquivo de estilo CSS -->
    </head>

    <body>
        <header>
            <nav class="nav-bar">
                <div class="logo">
                    <h1>MECH</h1>
                </div>
                <div class="nav-list">
                    <ul>
                        <li class="nav-item">
                            <a href="home.html" class="nav-link">Página Principal</a>
                        </li>
                        <li class="nav-item">
                            <a href="informações.html" class="nav-link">Informações</a>
                        </li>
                        <li class="nav-item">
                            <a href="exposição.html" class="nav-link"> Exposições</a>
                        </li>
                        <li class="nav-item">
                            <a href="compras.html" class="nav-link"> Compras</a>
                        </li>
                        <li class="nav-item">
                            <a href="questionario.html" class="nav-link">
                                Questionario/Avaliação</a>
                            <br>
                        </li>
                    </ul>
                </div>
                <div class="login-button">
                    <button id="loginBtn" class="login"><a href="login.html" id="loginLink">Entrar</a></button>
                    <button id="logoutBtn" class="logout" onclick="fazerLogout()">Logout</button>
                </div>
            </nav>
            <div class="mobile-menu-icon">
                <button onclick="menuShow()">
                    
                </button>
            </div>
        </div>
        <div class="mobile-menu">
            <ul>
                <li class="nav-item">
                    <a href="home.html" class="nav-link">Página Principal</a>
                </li>
                <li class="nav-item">
                    <a href="informações.html" class="nav-link">Informações</a>
                </li>
            </ul>
        </div>
    </body>

```

```

</li>
<li class="nav-item">
    <a href="exposicao.html" class="nav-link"> Exposições</a>
</li>
<li class="nav-item">
    <a href="compras.html" class="nav-link"> Compras</a>
</li>
<li class="nav-item">
    <a href="questionario.html" class="nav-link">
        Questionário/Avaliação</a>
    >
</li>
</ul>

<div class="login-button">
    <button id="loginBtn" class="login"><a href="login.html" id="loginLink">Entrar</a></button>
    <button id="logoutBtn" class="logout" onclick="fazerLogout()">Logout</button>
</div>
</div>
</header>

<div id="questionarioAvaliacao" class="container">
    <h1>Questionário e Avaliação</h1>
    <!-- Título principal da página -->

    <!-- Seção do Questionário -->
    <h2 class="section-title">Questionário</h2>
    <div id="questionarioRespostas" class="section">
        <!-- Pergunta 1 -->
        <div class="question">
            <p>Qual é o maior vulcão do sistema solar?</p>
            <div class="button-group">
                <!-- Contador de acertos -->
                <button
                    id = '1-e-1'
                    class="erros"
                    onclick="marcarRespostaQuestionario(1, 'erros')"
                >
                    Monte Kilimanjaro, com 5.895 metros de altura
                </button>
                <span id="resposta-1-erros">0</span>
                <!-- Contador de erros -->
                <button
                    id = '1-e-2'
                    class="erros"
                    onclick="marcarRespostaQuestionario(1, 'erros')"
                >
                    Mauna Kea, com 10.203 metros de altura
                </button>
                <span id="resposta-1-erro">0</span>
                <button
                    id = '1-e-3'
                    class="erros"
                    onclick="marcarRespostaQuestionario(1, 'erros')"
                >
                    Monte Everest, com 8.848 metros de altura
                </button>
                <span id="resposta-1-erro">0</span>
                <button
                    id = '1-a'
                    class="acertos"
                >
                    Aconcágua, com 6.962 metros de altura
                </button>
                <span id="resposta-1-acertos">0</span>
            </div>
        </div>
    </div>
</div>

```

```

        onclick="marcarRespostaQuestionario(1, 'acertos')"
    >
        Monte Olimpo, com 27 km de altura
    </button>
    <span id="resposta-1-acertos">0</span>
</div>
</div>
<!-- Pergunta 2 -->
<div class="question">
    <br />
    <p>
        A NASA tem planos concretos para enviar missões tripuladas para
        Marte na década de 2030 com a SpaceX. Quem fundou a SpaceX?
    </p>
    <div class="button-group">

        <button
            id = '2-e-1'
            class="erros"
            onclick="marcarRespostaQuestionario(2, 'erros')"
        >
            Neil Armstrong
        </button>
        <span id="resposta-2-erros">0</span>
        <button
            id = '2-e-2'
            class="erros"
            onclick="marcarRespostaQuestionario(2, 'erros')"
        >
            Michael “Lorde” Jackson
        </button>
        <span id="resposta-2-erro">0</span>
        <button
            id = '2-a'
            class="acertos"
            onclick="marcarRespostaQuestionario(2, 'acertos')"
        >
            Elon Musk
        </button>
        <span id="resposta-2-acertos">0</span>
        <button
            id = '2-e-3'
            class="erros"
            onclick="marcarRespostaQuestionario(2, 'erros')"
        >
            Sally Ride
        </button>
        <span id="resposta-2-erro">0</span>
    </div>
</div>
<!-- Pergunta 3 -->
<div class="question">
    <br />
    <p>
        Qual é o fenômeno que deixa o planeta Marte com a sua cor vermelha?
    </p>
    <div class="button-group">

        <button
            id = '3-e-1'
            class="erros"
            onclick="marcarRespostaQuestionario(3, 'erros')"
        >

```

```

        O alto teor de carbono na atmosfera
    </button>
    <span id="resposta-3-erros">0</span>
    <button
        id = '3-a'
        class="acertos"
        onclick="marcarRespostaQuestionario(3, 'acertos')"
    >
        A presença de óxido de ferro em sua superfície
    </button>
    <span id="resposta-3-acertos">0</span>
    <button
        id = '3-e-2'
        class="erros"
        onclick="marcarRespostaQuestionario(3, 'erros')"
    >
        A proximidade do Sol, causando oxidação intensa
    </button>
    <span id="resposta-3-erro">0</span>
    <button
        id = '3-e-3'
        class="erros"
        onclick="marcarRespostaQuestionario(3, 'erros')"
    >
        A presença de muito planeta deixou com vergonha
    </button>
    <span id="resposta-3-erro">0</span>
</div>
</div>
<!-- Pergunta 4 --&gt;
&lt;div class="question"&gt;
    &lt;br /&gt;
    &lt;p&gt;
        Qual descoberta sobre Marte pode indicar a possibilidade de vida no
        passado?
    &lt;/p&gt;
    &lt;div class="button-group"&gt;
        &lt;button
            id = '4-a'
            class="acertos"
            onclick="marcarRespostaQuestionario(4, 'acertos')"
        &gt;
            Evidências de água líquida em rios e lagos secos.
        &lt;/button&gt;
        &lt;span id="resposta-4-acertos"&gt;0&lt;/span&gt;
        &lt;button
            id = '4-e-1'
            class="erros"
            onclick="marcarRespostaQuestionario(4, 'erros')"
        &gt;
            Traços de fósseis de pequenos organismos
        &lt;/button&gt;
        &lt;span id="resposta-4-erros"&gt;0&lt;/span&gt;
        &lt;button
            id = '4-e-2'
            class="erros"
            onclick="marcarRespostaQuestionario(4, 'erros')"
        &gt;
            A presença de metano na atmosfera.
        &lt;/button&gt;
        &lt;span id="resposta-4-erro"&gt;0&lt;/span&gt;
        &lt;button
            id = '4-e-3'
</pre>

```

```

        class="erros"
        onclick="marcarRespostaQuestionario(4, 'erros')"
    >
        A descoberta de grandes depósitos de gelo.
    </button>
    <span id="resposta-4-erro">0</span>
</div>
</div>
<!-- Pergunta 5 -->
<div class="question">
    <br />
    <p>
        Qual das seguintes tecnologias será necessária para explorar o Monte
        Olímpo em Marte?
    </p>
    <div class="button-group">

        <button
            id ='5-e-1'
            class="erros"
            onclick="marcarRespostaQuestionario(5, 'erros')"
        >
            Módulos infláveis para habitats humanos.
        </button>
        <span id="resposta-5-erros">0</span>
        <button
            id ='5-e-2'
            class="erros"
            onclick="marcarRespostaQuestionario(5, 'erros')"
        >
            Foguetes de propulsão solar.
        </button>
        <span id="resposta-5-erros">0</span>
        <button
            id ='5-a'
            class="acertos"
            onclick="marcarRespostaQuestionario(5, 'acertos')"
        >
            Missões robóticas para mapear a superfície.
        </button>
        <span id="resposta-5-acertos">0</span>
        <button
            id ='5-e-3'
            class="erros"
            onclick="marcarRespostaQuestionario(5, 'erros')"
        >
            Satélites com câmeras de alta resolução para sobrevoos contínuos.
        </button>
        <span id="resposta-5-erro">0</span>
    </div>
</div>
</div>

<!-- Seção de Avaliação -->
<h2 id="avaliacao" class="section-title">Avaliação</h2>
<div id="avaliacaoRespostas" class="section">
    <!-- Pergunta 1 da Avaliação -->
    <div class="question">
        <p>Qualidade Exposições</p>
        <div class="button-group">
            <button
                class="excelente"
                onclick="marcarRespostaAvaliacao(1, 'excelente')"
            >

```

```

>
    Excelente
</button>
<span id="avaliacao-1-excelente">0</span>
<button class="bom" onclick="marcarRespostaAvaliacao(1, 'bom')">
    Bom
</button>
<span id="avaliacao-1-bom">0</span>
<button
    class="regular"
    onclick="marcarRespostaAvaliacao(1, 'regular')"
>
    Regular
</button>
<span id="avaliacao-1-regular">0</span>
<button class="ruim" onclick="marcarRespostaAvaliacao(1, 'ruim')">
    Ruim
</button>
<span id="avaliacao-1-ruim">0</span>
<button
    class="pessimo"
    onclick="marcarRespostaAvaliacao(1, 'pessimo')"
>
    Péssimo
</button>
<span id="avaliacao-1-pessimo">0</span>
</div>
</div>
<!-- Pergunta 2 da Avaliação --&gt;
&lt;div class="question"&gt;
    &lt;br /&gt;
    &lt;p&gt;Interação Totem Site&lt;/p&gt;
    &lt;div class="button-group"&gt;
        &lt;button
            class="excelente"
            onclick="marcarRespostaAvaliacao(2, 'excelente')"
&gt;
            Excelente
&lt;/button&gt;
&lt;span id="avaliacao-2-excelente"&gt;0&lt;/span&gt;
&lt;button class="bom" onclick="marcarRespostaAvaliacao(2, 'bom')"&gt;
    Bom
&lt;/button&gt;
&lt;span id="avaliacao-2-bom"&gt;0&lt;/span&gt;
&lt;button
    class="regular"
    onclick="marcarRespostaAvaliacao(2, 'regular')"
&gt;
    Regular
&lt;/button&gt;
&lt;span id="avaliacao-2-regular"&gt;0&lt;/span&gt;
&lt;button class="ruim" onclick="marcarRespostaAvaliacao(2, 'ruim')"&gt;
    Ruim
&lt;/button&gt;
&lt;span id="avaliacao-2-ruim"&gt;0&lt;/span&gt;
&lt;button
    class="pessimo"
    onclick="marcarRespostaAvaliacao(2, 'pessimo')"
&gt;
    Péssimo
&lt;/button&gt;
&lt;span id="avaliacao-2-pessimo"&gt;0&lt;/span&gt;
&lt;/div&gt;
</pre>

```

```

</div>
<!-- Pergunta 3 da Avaliação -->
<div class="question">
    <br />
    <p>Informações Claras</p>
    <div class="button-group">
        <button
            class="excelente"
            onclick="marcarRespostaAvaliacao(3, 'excelente')"
        >
            Excelente
        </button>
        <span id="avaliacao-3-excelente">0</span>
        <button class="bom" onclick="marcarRespostaAvaliacao(3, 'bom')">
            Bom
        </button>
        <span id="avaliacao-3-bom">0</span>
        <button
            class="regular"
            onclick="marcarRespostaAvaliacao(3, 'regular')"
        >
            Regular
        </button>
        <span id="avaliacao-3-regular">0</span>
        <button class="ruim" onclick="marcarRespostaAvaliacao(3, 'ruim')">
            Ruim
        </button>
        <span id="avaliacao-3-ruim">0</span>
        <button
            class="pessimo"
            onclick="marcarRespostaAvaliacao(3, 'pessimo')"
        >
            Péssimo
        </button>
        <span id="avaliacao-3-pessimo">0</span>
    </div>
</div>

<div class="section">
    <br />
    <label for="sugestao">Sugestão:</label>
    <textarea
        id="sugestao"
        maxlength="150"
        placeholder="Digite sua sugestão aqui..."></textarea>
</div>
</div>

<!-- Botão para enviar as respostas -->
<button id="concluir" class="enviar" onclick="avancarEstado()">Enviar
Respostas</button>
<div id="response"></div>
<!-- Área para exibir a resposta do envio -->
</div>
<script src="auth.js"></script>
<script src="Pim/Home/Questionario/questionario.js"></script>
</body>
</html>

```

APÊNDICE CJ – CODIGO SITE – QUESTIONARIO – CSS

```

/* Reset básico */
* {
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    outline: none;
    border: none;
    text-decoration: none;
    text-transform: capitalize;
    transition: .2s linear;
}

/* Estilo para o corpo da página */
body {
    background-color: #000000; /* Fundo escuro */
    color: #fff; /* Cor do texto */
    padding-top: 100px; /* Compensa o header fixo */
    margin: 0;
    overflow-x: hidden;
}

/* Header fixo no topo */
header {
    background-color: #444;
    position: fixed;
    width: 100%;
    top: 0;
    z-index: 1000;
}

.spinner {
    border: 2px solid #f3f3f3;
    border-top: 2px solid #3498db;
    border-radius: 50%;
    width: 12px;
    height: 12px;
    animation: spin 1s linear infinite;
    display: inline-block;
    margin-right: 8px;
}

@keyframes spin {
    0% { transform: rotate(0deg); }
    100% { transform: rotate(360deg); }
}

.nav-bar {
    display: flex;
    justify-content: space-between;
    padding: 1.5rem 6rem;
}

.logo {
    display: flex;
    align-items: center;
}

.logo h1 {
    color: #fff;
}

```

```
}

.nav-list {
    display: flex;
    align-items: center;
}

.nav-list ul {
    display: flex;
    justify-content: center;
    list-style: none;
}

.nav-item {
    margin: 0 35px;
}

.nav-link {
    text-decoration: none;
    font-size: 1.15rem;
    color: #fff;
    font-weight: 400;
}

.login-button button {
    border: none;
    padding: 10px 15px;
    border-radius: 5px;
    background-color: #970000; /* Cor vermelha do botão Entrar */
}

.login-button button a {
    text-decoration: none;
    color: #fff;
    font-weight: 500;
    font-size: 1.1rem;
}

.login-button button.login {
    background-color: #970000; /* Cor de fundo do botão de login */
    color: #fff;
    border: none;
    padding: 10px 20px;
    cursor: pointer;
}

/* Estilos para o botão de logout */
.login-button button.logout {
    color: #fff;
    border: 1px solid #970000;
    padding: 10px 20px;
    cursor: pointer;
    display: none; /* Esconde o botão de logout por padrão */
}

.login-button button:empty {
    display: none;
}

.mobile-menu-icon {
    display: none;
}
```

```
.mobile-menu {
    display: none;
}

/* Estilo do formulário centralizado */
.container {
    background-color: #444; /* Fundo do formulário */
    padding: 40px;
    border-radius: 10px;
    box-shadow: 0 4px 15px rgba(0, 0, 0, 0.4);
    max-width: 800px;
    width: 100%;
    max-height: 80vh;
    overflow-y: auto; /* Adiciona scroll se o conteúdo exceder a altura */
    color: #fff;
    margin: 0 auto;
}

/* Estilo para o título */
h1 {
    text-align: center;
    color: #fff;
    margin-bottom: 30px;
    font-size: 2rem;
}

/* Seção do Questionário e Avaliação */
h2.section-title {
    color: #ddd;
    border-bottom: 2px solid #555;
    padding-bottom: 10px;
    margin-top: 30px;
    margin-bottom: 20px;
}

/* Estilo das perguntas */
.question p {
    font-weight: bold;
    color: #ccc;
    margin-bottom: 10px;
}

.botao-correcto {
    background-color: green; /* Cor de fundo verde */
    color: white; /* Cor do texto branca */
}

/* Grupo de botões dentro do #questionarioRespostas */
#questionarioRespostas .button-group {
    display: flex;
    flex-direction: column; /* Alinha os botões em coluna */
    gap: 10px; /* Espaço entre os botões */
    align-items: stretch; /* Faz os botões ocuparem toda a largura */
}

#questionarioRespostas .button-group button {
    padding: 10px 15px;
    border: 1px solid #970000; /* Borda vermelha */
    border-radius: 5px;
    cursor: pointer;
    font-size: 14px;
    color: #970000; /* Texto vermelho */
    background-color: white; /* Fundo branco */
```

```

width: 100%; /* Faz com que cada botão ocupe toda a largura disponível */
transition: background-color 0.2s, color 0.2s; /* Transição suave para
mudança de cor */
}

/* Efeito ao clicar no botão */
#questionarioRespostas .button-group button.clicked {
    background-color: #000; /* Fica preto quando clicado */
    color: white; /* Texto branco quando clicado */
}

/* Botões no #avaliacaoRespostas com estilo original */
#avaliacaoRespostas .button-group {
    display: flex;
    gap: 10px;
    align-items: center;
}

#avaliacaoRespostas .button-group button {
    padding: 10px 15px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    font-size: 14px;
    color: #fff;
}

/* Botões com cores diferentes no #avaliacaoRespostas */
#avaliacaoRespostas .button-group .acertos, #avaliacaoRespostas .button-group
.excelente {
    background-color: #4CAF50; /* Verde */
}

#avaliacaoRespostas .button-group .erros, #avaliacaoRespostas .button-group
.pessimo {
    background-color: #f44336; /* Vermelho */
}

#avaliacaoRespostas .button-group .bom {
    background-color: #8BC34A; /* Verde claro */
}

#avaliacaoRespostas .button-group .regular {
    background-color: #FF9800; /* Laranja */
}

#avaliacaoRespostas .button-group .ruim {
    background-color: #FF5722; /* Vermelho escuro */
}

/* Texto dos contadores para ambos os grupos de botões */
.button-group span {
    font-size: 16px;
    color: #ccc;
    visibility: hidden; /* Torna o <span> invisível */
}

/* Área de texto para sugestão */
textarea {
    width: 100%;
    padding: 15px;
    border: 1px solid #666;
    border-radius: 5px;
}

```

```
resize: none;
background-color: #333;
color: #fff;
margin-top: 10px;
}

/* Botão de envio */
.enviar {
    background-color: #970000;
    color: white;
    padding: 12px 25px;
    border: none;
    border-radius: 5px;
    font-size: 16px;
    cursor: pointer;
    display: block;
    margin: 0 auto;
}

.enviar:hover {
    background-color: #b30000;
}

/* Media query para dispositivos móveis */
@media screen and (max-width: 440px) {
    body {
        overflow-y: scroll;
    }

    .nav-bar {
        padding: 1.5rem 4rem;
    }

    .nav-item {
        display: none;
    }

    .login-button {
        display: none;
    }

    .mobile-menu-icon {
        display: block;
    }

    .mobile-menu-icon button {
        background-color: transparent;
        border: none;
        cursor: pointer;
    }

    .mobile-menu ul {
        display: flex;
        flex-direction: column;
        text-align: center;
        padding-bottom: 1rem;
    }

    /* Ajustes para os itens do menu */
    .mobile-menu .nav-item {
        display: block;
        padding-top: 1.2rem;
    }
}
```

```
}

.mobile-menu .login-button {
    display: block;
    padding: 1rem 2rem;
}

.mobile-menu .login-button button {
    width: 100%;
}

/* Quando o menu está aberto, ele deve ser exibido */
.open {
    display: block;
}

/* Ajuste da transição para animar a altura ao abrir/fechar o menu */
.mobile-menu {
    overflow: hidden; /* Impede o conteúdo de transbordar */
    height: 0; /* Inicialmente o menu tem altura 0 (fechado) */
    transition: height 0.3s ease-in-out; /* Transição suave para abrir e fechar */
}

/* Quando o menu estiver aberto, ele deve ter a altura necessária */
.mobile-menu.open {
    height: auto; /* A altura do menu será automaticamente ajustada */
}

.container {
    padding: 20px;
    width: 100%;
    margin: 10px auto;
    max-height: auto;
    overflow-y: auto;
}

h1 {
    font-size: 1.5rem;
    margin-bottom: 20px;
}

h2.section-title {
    font-size: 1.2rem;
}

.button-group {
    display: flex;
    gap: 10px;
    flex-wrap: wrap;
    justify-content: center;
}

.button-group button {
    font-size: 12px;
    padding: 8px 12px;
    margin: 5px;
}

.button-group span {
    font-size: 14px;
}
```

```
.textarea {  
    width: 100%;  
    padding: 10px;  
}  
  
.enviar {  
    font-size: 14px;  
    padding: 10px 20px;  
}  
  
.question p {  
    font-size: 14px;  
    margin-bottom: 8px;  
}  
  
textarea {  
    width: 100%;  
    padding: 10px;  
    border-radius: 5px;  
    resize: none;  
    background-color: #333;  
    color: #fff;  
    margin-top: 10px;  
}  
  
.enviar {  
    font-size: 14px;  
    padding: 10px 20px;  
    display: block;  
    margin: 20px auto;  
}  
}  
  
/* Estilo dos botões dentro de #questionarioRespostas */
```

APÊNDICE CK – CODIGO SITE – QUESTIONARIO – JAVASCRIPT

```

console.log(localStorage.getItem("apiBase"));
const apiBase = localStorage.getItem("apiBase") || "https://default-api-
url.com/";

// Variável para controlar o estado atual (ID da página/etapa)
let estadoAtual = 1;

// Lista para armazenar respostas erradas
let listaRespostas = [];

// Função para mostrar/ocultar o menu
function menuShow() {
    let menuMobile = document.querySelector('.mobile-menu');
    let menuIcon = document.querySelector('.icon');

    // Verifica se o menu está aberto ou fechado e alterna o estado
    if (menuMobile.classList.contains('open')) {
        menuMobile.classList.remove('open'); // Fecha o menu
        menuIcon.src = "assets/img/menu_white_36dp.svg"; // Troca o ícone para
        "abrir"

        // Resetando a altura para 0 após o fechamento
        setTimeout(() => {
            menuMobile.style.height = '0'; // A altura do menu é definida como 0
            // Atualiza a página após o fechamento do menu
            window.location.reload(); // Recarrega a página
        }, 300); // Aguardar a transição de fechamento antes de resetar a altura
    } else {
        menuMobile.classList.add('open'); // Abre o menu
        menuIcon.src = "assets/img/close_white_36dp.svg"; // Troca o ícone para
        "fechar"

        // Ajusta a altura para auto para que o menu ocupe o espaço necessário
        menuMobile.style.height = 'auto';
    }
}

// Função para fechar o menu ao clicar fora (e resetar)
window.addEventListener('click', function (event) {
    let menuMobile = document.querySelector('.mobile-menu');
    let menuIconButton = document.querySelector('.mobile-menu-icon button');

    // Verifica se o clique foi fora do menu e do ícone do menu
    if (!menuMobile.contains(event.target) &&
    !menuIconButton.contains(event.target)) {
        if (menuMobile.classList.contains('open')) {
            menuMobile.classList.remove('open'); // Fecha o menu
            document.querySelector('.icon').src =
            "assets/img/menu_white_36dp.svg"; // Troca o ícone para "abrir"

            // Resetando a altura para 0 após o fechamento
            setTimeout(() => {
                menuMobile.style.height = '0'; // A altura do menu é definida
                como 0
                // Atualiza a página após o fechamento do menu
                window.location.reload(); // Recarrega a página
            }, 300); // Aguardar a transição de fechamento antes de resetar a
            altura
        }
    }
})

```

```

});

// Impede que o clique no ícone do menu feche o menu ao clicar nele
document.querySelector('.mobile-menu-icon button').addEventListener('click',
function(event) {
    event.stopPropagation(); // Impede que o clique no ícone seja tratado como
    // um clique fora do menu
});

// Inicializando arrays com as perguntas e valores zerados
let questionarioRespostas = [
    { id: 1, pergunta: 'VulcaoSistemaSolar', acertos: 0, erros: 0 },
    { id: 2, pergunta: 'FundadorSpaceX', acertos: 0, erros: 0 },
    { id: 3, pergunta: 'FenomenoVermelho', acertos: 0, erros: 0 },
    { id: 4, pergunta: 'VidaPassada', acertos: 0, erros: 0 },
    { id: 5, pergunta: 'MonteOlimpo', acertos: 0, erros: 0 }
];

let avaliacaoRespostas = [
    { id: 1, pergunta: 'QualidadeExposicoes', excelente: 0, bom: 0, regular: 0,
      ruim: 0, pessimo: 0 },
    { id: 2, pergunta: 'InteracaoTotemSite', excelente: 0, bom: 0, regular: 0,
      ruim: 0, pessimo: 0 },
    { id: 3, pergunta: 'InformacoesClaras', excelente: 0, bom: 0, regular: 0,
      ruim: 0, pessimo: 0 }
];

// Obtendo a sugestão do usuário (caso exista)
let sugestao = document.getElementById('sugestao').value.trim();

// Função para marcar resposta do questionário
function marcarRespostaQuestionario(id, resposta) {
    questionarioRespostas.forEach(pergunta => {
        if (pergunta.id === id) {
            // Reseta todas as respostas
            pergunta.acertos = 0;
            pergunta.erros = 0;

            // Define a resposta escolhida
            pergunta[resposta] += 1; // Agora incrementa a contagem

            console.log('acerto: ' + pergunta.acertos);
            console.log('erro: ' + pergunta.erros);
        }
    });
}

// Função para marcar resposta da avaliação
function marcarRespostaAvaliacao(id, resposta) {
    avaliacaoRespostas.forEach(avaliacao => {
        if (avaliacao.id === id) {
            // Reseta todas as respostas
            avaliacao.excelente = 0;
            avaliacao.bom = 0;
            avaliacao.regular = 0;
            avaliacao.ruim = 0;
            avaliacao.pessimo = 0;
        }
    });
}

```

```

        // Define a resposta escolhida
        avaliacao[resposta] += 1; // Agora incrementa a contagem
    }
});

// Atualiza a interface
atualizarContadoresAvaliacao(id);
}

// Função para atualizar contadores de avaliação na interface
function atualizarContadoresAvaliacao(id) {
    const avaliacao = avaliacaoRespostas.find(a => a.id === id);
    if (avaliacao) {
        document.getElementById(`avaliacao-${id}-excelente`).textContent =
        avaliacao.excelente;
        document.getElementById(`avaliacao-${id}-bom`).textContent =
        avaliacao.bom;
        document.getElementById(`avaliacao-${id}-regular`).textContent =
        avaliacao.regular;
        document.getElementById(`avaliacao-${id}-ruim`).textContent =
        avaliacao.ruim;
        document.getElementById(`avaliacao-${id}-pessimo`).textContent =
        avaliacao.pessimo;
    }
}

// Função para verificar se todas as perguntas foram respondidas
function todasAsPerguntasRespondidas() {
    // Verifica se todas as perguntas do questionário foram respondidas
    const todasRespondidasQuestionario = questionarioRespostas.every(pergunta =>
        pergunta.acertos > 0 || pergunta.erros > 0 // Checa se pelo menos uma
    resposta foi marcada
    );

    // Verifica se todas as perguntas da avaliação foram respondidas
    const todasRespondidasAvaliacao = avaliacaoRespostas.every(avaliacao =>
        avaliacao.excelente > 0 || avaliacao.bom > 0 || avaliacao.regular > 0 ||
        avaliacao.ruim > 0 || avaliacao.pessimo > 0
    );

    // Retorna true se todas as perguntas foram respondidas, caso contrário
    // retorna false
    return todasRespondidasQuestionario && todasRespondidasAvaliacao;
}

async function enviarDados() {
    if (!todasAsPerguntasRespondidas()) {
        alert("Por favor, responda todas as perguntas antes de enviar!");
        return; // Não envia os dados se nem todas as perguntas foram respondidas
    }

    // Junta a sugestão com as respostas do questionário e avaliação
    const sugestao = document.getElementById('sugestao').value.trim(); // Captura
    a sugestão

    const questionarioAvaliacaoRespostasDto = {
        questionarioRespostas: questionarioRespostas,
        avaliacaoRespostas: avaliacaoRespostas,
        ...(sugestao && { avaliacaoSugestao: { id: 0, sugestao: sugestao } })
    } // Adiciona somente se a sugestão não estiver vazia
}

// Log para verificar se a sugestão foi unida corretamente

```

```

console.log("Objeto enviado:", questionarioAvaliacaoRespostasDto);

fetch(apiBase + '/api/Master/QuestionarioAvaliacao', {
    method: 'POST',
    headers: {
        'Content-Type': 'application/json',
        "ngrok-skip-browser-warning": "true"
    },
    body: JSON.stringify(questionarioAvaliacaoRespostasDto)
})
.then(response => {
    if (!response.ok) {
        throw new Error('Erro na resposta da API: ' + response.status);
    }
    return response.json(); // Tenta converter a resposta em JSON
})
.then(data => {
    // Acessa diretamente a propriedade message da resposta
    document.getElementById('response').textContent = data.message; // Exibe
    somente a mensagem
})
.catch(error => {
    document.getElementById('response').textContent = 'Erro ao enviar dados:
' + error;
});
}

// Função para mostrar feedback visual ao clicar em um botão
function mostrarFeedback(botao) {
    botao.classList.add('clicked');

    // Remove a classe após 200ms para criar um efeito de clique
    setTimeout(() => {
        botao.classList.remove('clicked');
    }, 200);
}
// Seleciona todos os grupos de botões
const gruposDeBotoes = document.querySelectorAll('.button-group');

// Para cada grupo de botões
gruposDeBotoes.forEach(grupo => {
    // Seleciona todos os botões dentro do grupo
    const botoes = grupo.querySelectorAll('button');

    // Adiciona um evento de clique para cada botão dentro do grupo
    botoes.forEach(botao => {
        botao.addEventListener('click', function() {
            // Remove a cor de fundo preta de todos os botões do grupo
            botoes.forEach(b => b.style.backgroundColor = '');

            // Altera a cor de fundo para preto do botão clicado
            this.style.backgroundColor = 'black';
        });
    });
});

// Seleciona todos os botões dentro do elemento com id "questionarioRespostas"
const botoes = document.querySelectorAll('#questionarioRespostas button');

// Adiciona um evento de clique para cada botão
botoes.forEach(botao => {
    botao.addEventListener('click', function() {
        // Se o botão já foi clicado, não faz nada
    });
});

```

```

    if (this.disabled) return;

    // Desabilita todos os botões da mesma questão
    const questionId = this.closest('.question').dataset.id; // Assume que a
    questão tem um atributo data-id com o id da pergunta
    const botoesQuestion = document.querySelectorAll(`.question[data-
    id="${questionId}"] button`);
    botoesQuestion.forEach(b => b.disabled = true);

    // Marca a resposta para a questão
    const resposta = this.dataset.resposta; // Assume que cada botão tem um
    atributo data-resposta com o valor da resposta
    marcarRespostaQuestionario(questionId, resposta);

    // Altera a cor de fundo para preto do botão clicado
    this.style.backgroundColor = 'black';

    // Exibe feedback visual
    mostrarFeedback(this);
  });
}

let botaoBloqueado = false;

async function avancarEstado() {

  if (botaoBloqueado) {
    console.log('Botão bloqueado');
    return;
  }

  botaoBloqueado = true;
  estadoAtual++;

  switch (estadoAtual) {
    case 1: // Estado inicial: Questionário
      botaoBloqueado = false;
      break;

    case 2:
      // Apenas avança para o próximo estado se todas as perguntas forem
      respondidas
      if (!todasAsPerguntasRespondidas()) {
        alert("Por favor, responda todas as perguntas antes de enviar!");
        estadoAtual = 1;

        botaoBloqueado = false;
        return; // Não envia os dados se nem todas as perguntas foram
      respondidas
      }

      const container = document.getElementById("questionarioAvaliacao");
      container.scrollTo({ top: 0, behavior: "smooth" });

      const botao = document.getElementById('concluir');
      botao.innerHTML = '<span class="spinner"></span> Enviando...';

      await enviarDados().then(() => {
        botao.innerHTML = 'Enviar Respostas';
        atualizarInterfaceParaFeedback();
      });
  }
}

```

```

        botaoBloqueado = false;
    });

    // Atualiza a interface para exibir o feedback de respostas

    botaoBloqueado = false;
    break;

    case 3: // Estado de Feedback

        botaoBloqueado = false;
        // Atualiza a interface para voltar à Home
        voltarParaHome();
        break;

    default:

        botaoBloqueado = false;
        console.error("Estado inválido!");
    }

}

// Função para atualizar a interface para o feedback de respostas
function atualizarInterfaceParaFeedback() {
    // Torna a avaliação invisível
    document.querySelector('#avaliacaoRespostas').style.display = 'none';
    document.querySelector('#avaliacao').style.display = 'none';

    // Atualiza o botão "Concluir" para "Continuar"
    const botaoConcluir = document.querySelector('#concluir');
    botaoConcluir.textContent = 'Continuar';

    // Verifica as respostas e atualiza os botões
    const botoes = [
        '1-e-1', '1-e-2', '1-e-3', '1-a',
        '2-e-1', '2-e-2', '2-e-3', '2-a',
        '3-e-1', '3-e-2', '3-e-3', '3-a',
        '4-e-1', '4-e-2', '4-e-3', '4-a',
        '5-e-1', '5-e-2', '5-e-3', '5-a'
    ];

    botoes.forEach(id => {
        const botao = document.getElementById(id);
        if (botao) {
            botao.disabled = true; // Desabilita o botão

            // Verifica se o botão é a resposta correta (identificado pelo sufixo
            '-a')
            if (id.endsWith('-a')) {
                botao.style.backgroundColor = '#28a745'; // Verde vibrante
                botao.style.color = 'white'; // Texto branco para destacar
                botao.style.boxShadow = '0 4px 6px rgba(40, 167, 69, 0.3)'; // Sombra suave para dar destaque
                botao.style.border = 'none'; // Remover borda para um estilo
                mais limpo
            }
        }
    });
}

// Função para voltar para a Home
function voltarParaHome() {

```

```
// Redireciona ou atualiza a interface para a página inicial
window.location.href = 'home.html'; // Exemplo: redireciona para a página
Home
}
```

APÊNDICE CL – EXPLICAÇÃO TAGS EM CSS

*****: Seleciona todos os elementos da página. Aplica um estilo geral de reset, removendo margens, bordas, e outros estilos padrão, e define fontes e transições.

body: Aplica estilos ao corpo da página, definindo propriedades como cor de fundo, cor do texto e controle de rolagem.

header: Estiliza o cabeçalho fixo no topo da página, definindo o fundo e a posição fixa.

nav-bar: Define a barra de navegação, organizando seus itens de forma flexível e aplicando padding.

.logo: Estiliza o logo dentro da barra de navegação, centralizando seus itens.

.logo h1: Aplica cor branca ao título dentro da classe logo.

.nav-list: Estiliza a lista de navegação, utilizando flex para alinhar os itens horizontalmente.

.nav-list ul: Aplica a listagem flexível aos itens de navegação, removendo os pontos de lista e centralizando os itens.

.nav-item: Estiliza os itens de navegação, definindo margens entre eles.

.nav-link: Aplica estilos aos links dentro da navegação, removendo o sublinhado e ajustando a cor e o peso da fonte.

.login-button button: Define o estilo do botão de login, incluindo cor de fundo, borda arredondada e espaçamento.

.login-button button a: Estiliza o link dentro do botão de login, alterando a cor e o peso da fonte.

.login-button button.login: Aplica estilos específicos para o botão de login, incluindo cor de fundo vermelha e texto branco.

.login-button button.logout: Estiliza o botão de logout, alterando cor de fundo, borda e visibilidade.

.login-button button:empty: Esconde o botão quando ele não tem conteúdo.

.mobile-menu-icon: Estiliza o ícone do menu móvel, tornando-o invisível por padrão.

.mobile-menu: Define o menu móvel como oculto por padrão, e aplica transições suaves para abrir e fechar o menu.

.home: Aplica o estilo à seção principal da página inicial, com uma imagem de fundo e configurações de altura mínima e alinhamento de conteúdo.

.home .content: Estiliza a área de conteúdo dentro da seção home, controlando tamanho máximo e altura.

.home .content h3: Estiliza o título dentro da seção content da página inicial, ajustando tamanho e cor.

.home .content p: Estiliza o parágrafo dentro da seção content da página inicial, ajustando a cor e o espaçamento.

@media screen and (max-width: 440px): Aplica estilos específicos para dispositivos com largura de tela até 440px, ajustando a navegação e a visibilidade de itens.

@media screen and (max-width: 375px): Similar ao anterior, mas para telas menores (375px de largura).

@media screen and (max-width: 360px): Aplica ajustes para telas ainda menores (360px de largura), otimizando a navegação e a disposição do conteúdo.

.spinner: Estiliza um elemento de carregamento circular, utilizando animação para girar.

@keyframes spin: Define a animação de rotação usada na classe .spinner.

.container: Estiliza um formulário ou conteúdo centralizado, com fundo escuro, bordas arredondadas e sombra.

h1: Estiliza o título principal, centralizando o texto e ajustando o tamanho da fonte.

h2.section-title: Aplica estilo a títulos de seção dentro de um formulário, incluindo cor, borda inferior e espaçamento.

.question p: Estiliza os parágrafos dentro de perguntas, ajustando a cor e o peso da fonte.

.botao-correto: Estiliza um botão com fundo verde e texto branco.

#questionarioRespostas .button-group: Agrupa os botões dentro de um formulário de questionário, organizando-os em coluna e com espaçamento.

#questionarioRespostas .button-group button: Estiliza os botões dentro do questionário, ajustando bordas, cores e transições.

#questionarioRespostas .button-group button.clicked: Modifica o estilo do botão ao ser clicado, alterando as cores de fundo e texto.

#avaliacaoRespostas .button-group: Similar ao #questionarioRespostas .button-group, mas para a seção de avaliação, com uma disposição de botões ajustada.

#avaliacaoRespostas .button-group button: Aplica estilo aos botões na avaliação, ajustando bordas e cores de fundo.

#avaliacaoRespostas .button-group .acertos, #avaliacaoRespostas .button-group .excelente: Aplica cor verde aos botões de acertos e excelente na avaliação.

#avaliacaoRespostas .button-group .erros, #avaliacaoRespostas .button-group .pessimo: Aplica cor vermelha aos botões de erros e péssimo na avaliação.

#avaliacaoRespostas .button-group . bom: Aplica cor verde clara ao botão de bom na avaliação.

#avaliacaoRespostas .button-group .regular: Aplica cor laranja ao botão de regular na avaliação.

#avaliacaoRespostas .button-group .ruim: Aplica cor vermelha escura ao botão de ruim na avaliação.

.button-group span: Estiliza um texto de contagem ou indicador dentro dos botões, tornando-o invisível por padrão.

textarea: Estiliza áreas de texto, ajustando fundo, bordas, cor do texto e o comportamento de redimensionamento.

.enviar: Estiliza o botão de envio, aplicando cor de fundo vermelha, bordas arredondadas e um efeito de hover.

.enviar:hover: Aplica uma cor de fundo mais escura ao botão de envio quando o mouse passa sobre ele.