



UNIVERSIDADE FEDERAL DO CEARÁ
INSTITUTO DE ARQUITETURA E URBANISMO E DESIGN
CURSO DE DESIGN

LUNA GONDIM CEZAR

FACILIART: PLATAFORMA WEB PARA CRIAÇÃO DE ARTE GENERATIVA POR INTERFACE
GRÁFICA

FORTALEZA
2023

LUNA GONDIM CEZAR

FACILIART: PLATAFORMA WEB PARA CRIAÇÃO DE ARTE GENERATIVA POR INTERFACE GRÁFICA

Trabalho de Conclusão de Curso apresentado ao Curso de Design da Universidade Federal do Ceará, como requisito parcial à obtenção do título de Bacharel em Design.

Orientador: Prof. Dr. Roberto Cesar Cavalcante Vieira

FORTALEZA

2023

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas

Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

- C419f Cezar, Luna Gondim.
Faciliart : plataforma web para criação de arte generativa por interface gráfica / Luna Gondim Cezar. –
2023.
115 f. : il. color.
- Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Instituto de Arquitetura e
Urbanismo e Design, Curso de Design, Fortaleza, 2023.
Orientação: Prof. Dr. Roberto Cesar Cavalcante Vieira.
1. Arte Generativa. 2. Experiência do Usuário. 3. Interface do Usuário. 4. Desenvolvimento Web. 5.
Modelo de Negócio. I. Título.

CDD 658.575

LUNA GONDIM CEZAR

FACILIART: PLATAFORMA WEB PARA CRIAÇÃO DE ARTE GENERATIVA POR INTERFACE GRÁFICA

Trabalho de Conclusão de Curso apresentado ao Curso de Design da Universidade Federal do Ceará, como requisito parcial à obtenção do título de Bacharel em Design.

Orientador: Prof. Dr. Roberto Cesar Cavalcante Vieira

Aprovada em ____/____/____

BANCA EXAMINADORA

Prof. Dr. Roberto Cesar Cavalcante Vieira (Orientador)
Universidade Federal do Ceará (UFC)

Prof^ª. Dr.^a. Aura Celeste Santana Cunha
Universidade Federal do Ceará (UFC)

Prof. Dr. Paulo Jorge Alcobia Simões
Universidade Federal do Ceará (UFC)

Esp. Maria de Souza Barros Neta
Especialista em Hematologia e Banco de Sangue (UNICHRISTUS)

AGRADECIMENTOS

Agradeço à minha família e, em especial, à minha mãe, Karla Silvana Teixeira Bastos, e à minha avó, Maria Zélia Teixeira Bastos, por todo apoio que recebi até aqui.

Agradeço ao Prof. Dr. Roberto César Cavalcante Vieira por ter me orientado com bastante cuidado e atenção durante todo o processo de produção deste trabalho, de maneira a auxiliar com que o mesmo pudesse ter o nível técnico desejado.

Agradeço aos membros internos da banca Prof.^a Dr.^a Aura Celeste Santana Cunha e Prof. Dr. Paulo Jorge Alcobia Simões por terem participado da avaliação deste trabalho, fornecendo sugestões muito importantes para a melhoria do mesmo.

Agradeço à Esp. Maria de Souza Barros Neta por sua disponibilidade e dedicação na participação como convidada da banca avaliadora, agregando bastante na discussão de novas ideias para este trabalho.

Agradeço ao corpo docente do Curso de Design por toda qualificação que recebi até aqui.

"The process of preparing programs for a digital computer is especially attractive, not only because it can be economically and scientifically rewarding, but also because it can be an aesthetic experience much like composing poetry or music."

Donald E. Knuth

RESUMO

Arte generativa é uma forma alternativa de criação de obras por meio do uso de diretrizes, as quais podem ser automatizadas ou não. Com a evolução tecnológica dos computadores, essa área artística ganhou notoriedade com a difusão de várias ferramentas, possuindo interface gráfica ou não, que permitiam que um usuário pudesse gerar padrões e criar, com isso, uma obra de arte. No entanto, os *softwares* que foram planejados para gerar padrões por interface gráfica possuem limitações, restringindo as possibilidades do que pode ser feito, de forma com que neste projeto seja proposta a implementação de uma plataforma web que permita ao usuário gerar vários tipos de padrões geométricos sem necessitar que o utilizador escreva algoritmos, incluindo a venda do resultado final e, também, a validação da interface de usuário do projeto por meio da aplicação de uma versão simplificada da UEQ (*User Experience Questionnaire*).

Palavras-chave: Arte Generativa; Experiência do Usuário; Interface do Usuário; Desenvolvimento Web; Modelo de Negócio

ABSTRACT

Generative art is an alternative method of creation of artworks by the use of guidelines, that can be automated or not. With the technological evolution of computers, this artistic area gained notoriety through the diffusion of various tools, which may or may not have a graphic interface, that allowed that an user could generate patterns and create, with that, an artwork. However, the softwares that were planned to generate patterns through graphic interface have limitations, restricting the possibilities of what can be made, so it's proposed for this project a web platform implementation that allows the user to generate many types of geometrical patterns without the need of writing algorithms, in addition to the sale of the final result and, also, the validation of the user interface through the use of a simplified version of the UEQ (User Experience Questionnaire).

Keywords: Generative Art; User Experience; User Interface; Web Development; Business Model

LISTA DE FIGURAS

1	Máquina de Loom	5
2	Cartões perfurados compatíveis com a máquina de Loom	6
3	ENIAC em funcionamento	7
4	UNIVAC em funcionamento	7
5	Sketchpad demonstrado por Ivan Sutherland	8
6	HP-85, computador lançado em 1980	9
7	Lingo em Adobe Director	9
8	Plasma desktop, uma interface de janelas e diálogos	11
9	Microsoft Excel, uma interface tabular	12
10	Jupyter Notebook, uma interface de bloco de notas	12
11	Site da Universidade Federal do Ceará, uma interface web	13
12	Xerox Alto executando interface gráfica	14
13	Interface do Smalltalk para o Xerox Alto	14
14	Bravo, um processador de texto WYSIWYG para o Xerox Alto	15
15	Apple Lisa 1	15
16	Interface do Lisa 2	16
17	Apple Macintosh	16
18	Interface do Mac OS 1.0	17
19	Interface do ambiente GEM 1.1	17
20	Interface do ambiente Microsoft Windows 1.01	18
21	Visão geral do Business Model Canvas	22
22	Rails — Listagem de usuários	24
23	Rails — Criação de usuário	24
24	Rails — Edição de usuário	25
25	Rails — Visualização de usuário	25
26	Tipos de diagramas UML	27
27	Diagrama de casos de uso para um sistema de turmas	29
28	Diagrama de atividades com particionamento e paralelismo	30
29	Diagrama de entidade e relacionamento para uma turma	31
30	Diagrama da metodologia de Design de Gui Bonsiepe	34
31	Diagrama da metodologia de desenvolvimento em Cascata	35
32	Looper — Criação da figura	36
33	Looper — Seleção de parâmetros	37
34	Looper — Resultado final	37
35	Everypixel Patterns — Customização do padrão selecionado	38

36	Plain Pattern — Criação de padrão	39
37	Korpus — Padrão aleatório	40
38	Korpus — Geração de autômato celular com a palavra “Design”	40
39	Silk — Imagem realizada por usuário	41
40	GPU Fluid — Imagem realizada por usuário	42
41	Linify — Geração de padrão utilizando logotipo da Universidade Federal do Ceará	42
42	Processing — Interface do programa e renderização de elipse	43
43	Design By Numbers — Exemplo de código e sua renderização	44
44	Chromata — Logotipo da Universidade Federal do Ceará com duas variações de geração	45
45	Página da Etsy	46
46	Página da Printify	47
47	Página da Redbubble	48
48	Página da Society6	49
49	Página da Artsy	50
50	Geniverse — Logotipo	52
51	Faciliart — Logotipo	52
52	Faciliart — Tipografia do projeto	53
53	Faciliart — Paleta de Cores	54
54	Página de edição de arte — Mockup	55
55	Página de listagem de artes — Mockup	56
56	Página de visualização de arte — Mockup	56
57	Página de edição de arte — Primeira implementação	57
58	Página de edição de arte — Segunda implementação	58
59	Página inicial — Primeira implementação	58
60	Layout de registro — Primeira implementação	59
61	Página de visualização de arte — Primeira implementação	59
62	Página de perfil do usuário — Primeira implementação	60
63	Página inicial — Implementação final	66
64	Página de edição de arte — Implementação final	67
65	Página de visualização de arte — Implementação final	67
66	Páginas de registro e login — Implementação final	68
67	Página de perfil de usuário — Implementação final	68
68	Versão <i>mobile</i> da plataforma	69
69	Alterações finais na tela de edição	71
70	Mockup de compra de blusa na plataforma Faciliart	72
71	Estrutura do projeto Faciliart	73

72	Modelo de negócio da plataforma Faciliart	76
73	Matriz SWOT da Faciliart	77
74	Business Model Canvas da Faciliart	78
75	Questionário UEQ em português	80
76	Questionário UEQ-S em português	81
77	Páginas analisadas na pesquisa	82
78	Diagrama de casos de uso em UML	86
79	Diagrama de atividade de registro de usuário em UML	87
80	Diagrama de atividade de login de usuário em UML	88
81	Diagrama de atividade de criação de arte em UML	89
82	Diagrama de atividade de edição de arte em UML	90
83	Diagrama de atividade de exclusão de arte em UML	91
84	Diagrama de atividade de visualização de perfil em UML	92
85	Diagrama de atividade de compra de arte em UML	93
86	Diagrama de atividade de venda de arte em UML	94
87	Diagrama de Entidade e Relacionamento da Faciliart	95

LISTA DE TABELAS

1	Diagramas UML estruturais	28
2	Diagramas UML comportamentais	28
3	Cardinalidade no diagrama de entidade e relacionamento	31
4	Cardinalidade para a representação simplificada de um sistema de turma	32
5	Procedimentos	33
6	Fatores avaliados no UEQ	79
7	Questionário de usabilidade elaborado com base nos fatores do UEQ	81
8	Criação de arte — Resultados da pesquisa de usabilidade	83
9	Visualização de arte — Resultados da pesquisa de usabilidade	83
10	Perfil de usuário — Resultados da pesquisa de usabilidade	84
11	Criação de arte — Resultados da pesquisa de usabilidade — Pós-ajuste	85
12	Visualização de arte — Resultados da pesquisa de usabilidade — Pós-ajuste	85
13	Perfil de usuário — Resultados da pesquisa de usabilidade — Pós-ajuste	85
14	Entidade User	95
15	Entidade Art	96
16	Entidade Comment	96
17	Entidade Item	96
18	Entidade ItemSize	97
19	Entidade Order	97
20	Entidade Setting	97

SUMÁRIO

1	INTRODUÇÃO	1
1.1	Contextualização	1
1.2	Problema	1
1.3	Pergunta	1
1.4	Objetivos gerais	1
1.5	Objetivos específicos	2
1.6	Justificativa	2
1.7	Delimitação	3
1.8	Estrutura de pesquisa	3
2	REFERENCIAL TEÓRICO	4
2.1	Arte generativa	4
<i>2.1.1</i>	<i>Conceito</i>	<i>4</i>
<i>2.1.2</i>	<i>História</i>	<i>5</i>
<i>2.1.3</i>	<i>Questões relevantes</i>	<i>10</i>
2.2	Interface Gráfica do Utilizador	11
<i>2.2.1</i>	<i>Conceito</i>	<i>11</i>
<i>2.2.2</i>	<i>História</i>	<i>13</i>
<i>2.2.3</i>	<i>Métodos de prototipagem</i>	<i>18</i>
<i>2.2.3.1</i>	<i>Protótipo de baixa fidelidade</i>	<i>18</i>
<i>2.2.3.2</i>	<i>Protótipo de alta fidelidade</i>	<i>18</i>
<i>2.2.4</i>	<i>Metodologias de análise de usabilidade</i>	<i>19</i>
<i>2.2.4.1</i>	<i>Avaliação heurística de Nielsen</i>	<i>19</i>
<i>2.2.4.2</i>	<i>Questionário</i>	<i>21</i>
2.3	Ferramentas de modelo de negócio	22
<i>2.3.1</i>	<i>Matriz SWOT</i>	<i>22</i>
<i>2.3.2</i>	<i>Business Model Canvas</i>	<i>22</i>
2.4	Desenvolvimento Web	23
<i>2.4.1</i>	<i>Front-end</i>	<i>23</i>
<i>2.4.2</i>	<i>Back-end</i>	<i>23</i>
2.5	Unified Modeling Language (UML)	26
<i>2.5.1</i>	<i>Conceito</i>	<i>26</i>
<i>2.5.2</i>	<i>Diagramas</i>	<i>26</i>
<i>2.5.3</i>	<i>Diagramas utilizados</i>	<i>29</i>
<i>2.5.3.1</i>	<i>Diagrama de casos de uso</i>	<i>29</i>
<i>2.5.3.2</i>	<i>Diagrama de atividades</i>	<i>30</i>

2.6	Diagrama de Entidade e Relacionamento	30
3	METODOLOGIA	32
3.1	Pesquisa	32
3.1.1	<i>Propósito</i>	32
3.1.2	<i>Abordagem</i>	32
3.1.3	<i>Procedimentos</i>	33
3.2	Projeto	34
3.2.1	<i>Design</i>	34
3.2.2	<i>Computação</i>	35
4	ANÁLISE DE SIMILARES	36
4.1	Geradores	36
4.1.1	<i>Por interface de usuário</i>	36
4.1.1.1	<i>Looper</i>	36
4.1.1.2	<i>Everypixel Patterns</i>	38
4.1.1.3	<i>Plain Pattern</i>	39
4.1.1.4	<i>Korpus</i>	39
4.1.1.5	<i>Silk</i>	41
4.1.1.6	<i>GPU Fluid</i>	41
4.1.1.7	<i>Linify</i>	42
4.1.2	<i>Por algoritmo</i>	43
4.1.2.1	<i>Processing</i>	43
4.1.2.2	<i>Pts.js</i>	44
4.1.2.3	<i>Chromata</i>	45
4.1.2.4	<i>Nannou</i>	45
4.1.2.5	<i>OPENRNDR</i>	45
4.1.2.6	<i>C4iOS</i>	46
4.1.2.7	<i>Canvas Sketch</i>	46
4.2	Modelos de negócio	46
4.2.1	<i>Serviços</i>	46
4.2.1.1	<i>Etsy</i>	46
4.2.1.2	<i>Printify</i>	47
4.2.1.3	<i>Redbubble</i>	48
4.2.1.4	<i>Society6</i>	49
4.2.1.5	<i>Artsy</i>	50
5	PROJETO	51
5.1	Links úteis	51

5.2	Escopo do projeto	51
5.2.1	<i>Requisitos</i>	51
5.2.1.1	<i>Funcionais</i>	51
5.2.1.2	<i>Não funcionais</i>	51
5.3	Design do projeto	52
5.3.1	<i>Identidade visual</i>	52
5.3.2	<i>Interface de usuário</i>	54
5.4	Implementação do projeto	72
5.4.1	<i>Tecnologias utilizadas</i>	72
5.4.2	<i>Processo de desenvolvimento</i>	73
5.5	Modelo de negócio	76
5.5.1	<i>SWOT</i>	77
5.5.2	<i>Canvas</i>	78
5.6	Pesquisa de usabilidade	79
5.6.1	<i>Metodologia</i>	79
5.6.2	<i>Primeira pesquisa</i>	83
5.6.3	<i>Segunda pesquisa — Pós-ajustes</i>	84
5.7	Diagramas	86
5.7.1	<i>UML — Casos de uso</i>	86
5.7.2	<i>UML — Atividades</i>	87
5.7.2.1	<i>Registrar</i>	87
5.7.2.2	<i>Login</i>	88
5.7.2.3	<i>Criar arte</i>	89
5.7.2.4	<i>Editar arte</i>	90
5.7.2.5	<i>Deletar arte</i>	91
5.7.2.6	<i>Ver perfil</i>	92
5.7.2.7	<i>Comprar arte</i>	93
5.7.2.8	<i>Vender arte</i>	94
5.7.3	<i>Diagrama de Entidade e Relacionamento</i>	95
6	CONCLUSÃO	98
	REFERÊNCIAS	99

1 INTRODUÇÃO

1.1 Contextualização

As origens da arte generativa provêm dos anos 1950 com a aplicação de algoritmos computacionais nas diferentes formas de arte, como pintura e música e, durante os anos, evoluiu de acordo com a evolução tecnológica (BODEN; EDMONDS, 2009). No entanto, a principal forma de interação do usuário para a criação deste tipo de arte continua sendo por meio de código (ver seção 4.1.2), sendo, embora hajam outras alternativas de ferramentas via algoritmo, o Processing¹ uma das principais opções (FISHWICK, 2006, p. 9) para este caso de uso, por conta do mesmo método oferecer um maior controle sob as variações de padrões para a concepção da peça de arte. Mesmo que hajam algumas tentativas de implementação por interface gráfica (ver seção 4.1.1), as mesmas possuem limitações, como a necessidade de criação ou *upload* de gravuras pelo usuário para haver a aplicação de repetição em padrões assim como, quando há a possibilidade de aleatoriedade, apenas existir a geração de um tipo de padrão que, aliadas à ausência ou a baixa disponibilidade de parâmetros para a customização do estilo da arte, resultam na carência de *features* para tais ferramentas, sendo necessário que o usuário utilize vários softwares dependendo do tipo de arte a ser gerada caso não opte por escrever um algoritmo. Logo, desta forma, pode-se afastar parte do público-alvo interessado na criação desta nova forma de arte.

1.2 Problema

A limitação das ferramentas atuais de arte generativa por meio de interface gráfica, tanto por necessidade de *input* manual de figuras quanto por geração de um único tipo de estilo de arte, podem desincentivar pessoas interessadas a criar peças neste ramo. Como cada utilitário de arte generativa por interface, na maioria dos casos, tem apenas uma modalidade de geração de imagem, o usuário necessita utilizar várias alternativas a depender do propósito.

1.3 Pergunta

De acordo com a problemática exposta, é possível realizar a implementação de uma ferramenta de arte generativa que dispense o uso de programação por parte do usuário e que forneça muitas possibilidades de criação utilizando apenas interface gráfica?

1.4 Objetivos gerais

O referido projeto, com o desenvolvimento do sistema proposto, busca criar um conceito alternativo de usabilidade referente à criação de artes generativas, de forma a apresentar, por meio de interface gráfica, várias alternativas de formas e parâmetros a serem escolhidos pelo usuário, sendo possível que a geração da arte possibilite que a mesma seja postada na plataforma e, então, permitindo que esta possa se tornar um quadro ou uma estampa de blusa, sendo também exequível que criadores de tais artes possam receber uma porcentagem do valor pago na efetivação do pedido de cada estampa de blusa.

¹<https://processing.org/>

1.5 Objetivos específicos

- Criar uma interface de usuário que possibilite, sem edição de código, a geração de artes por meio da disponibilização de vários padrões predefinidos, permitindo que o usuário consiga apenas selecionar os padrões desejados, possibilitando a posterior edição dos parâmetros, como coordenadas de início, distância entre pontos, entre outros, por meio de componentes como botões e sliders.
- Desenvolver estudos de viabilidade de negócio e de experiência do usuário em relação à obtenção de quadros ou de blusas com estampas provenientes da galeria de artes geradas por diversos usuários, na qual, com a efetivação do pedido, parte do valor volte ao criador da arte gerada.
- Implementar um *front-end* que respeite os *inputs* definidos pelo usuário por meio da interface e passe tais dados para um *canvas*, o qual serão exibidas todas as camadas geradas pelo utilizador.
- Implementar a estrutura de dados e a lógica do *back-end* do sistema de artes generativas, permitindo que informações oriundas da geração de cada peça; quantidade, assim como o tipo, de pedidos por peça; a contabilização de saldo e variáveis importantes para a administração do sistema sejam armazenadas, formando assim uma galeria desta forma de arte, na qual cada artista possa, também, verificar o seu saldo geral e por peça de acordo com o número dos pedidos.
- Validar o comportamento do usuário diante do conceito da interface e das funcionalidades oferecidas pelo sistema, de forma a analisar a eficiência de navegação do usuário para os casos de uso no qual o projeto se compromete a cumprir, descobrindo se o utilizador está tendo facilidade em manusear os recursos ofertados pelo *software*.

1.6 Justificativa

Embora possa haver resistência à mudança de paradigma de interface de usuário para algum tipo de atividade feita no computador (FISHWICK, 2006, p. 10), a proposta de criação de uma plataforma web que busque resolver este tipo de caso de uso é bastante pertinente para a discussão de novas alternativas de implementação para este tipo de arte, de forma a trazer novos conceitos para a experiência do usuário na área de estudo deste trabalho.

Dessa forma, tal conceito permite que o usuário que ainda não possua fluência na construção de algoritmos computacionais tenha interesse em realizar arte generativa por meio de uma interface de usuário que disponibilize vários tipos de figuras ou padrões que possam ser gerados aleatoriamente e controlados por meio de parâmetros, o que permite uma customização adequada para o processamento do resultado final.

Logo, em meio a este contexto, é proposta a criação de uma plataforma *web* que disponibilize uma grande quantidade de padrões e formas geométricas, permitindo a seleção das mesmas e a configuração de seus parâmetros por meio de uma interface de usuário, que possam ser geradas de forma aleatória, sendo então possível a postagem da obra finalizada na mesma plataforma.

Além disso, com a popularização de serviços nos quais artistas podem disponibilizar suas peças para impressão sob demanda (ver seção 4.2), há a motivação de que tal funcionalidade, com o estudo dos modelos de negócio dos *marketplaces* citados, também seja implementada na plataforma web no presente trabalho, para que todo

o processo de geração e comercialização de arte em estampas de blusa ou impressões de quadros sob demanda possa ser feito da maneira mais uniforme e integrada possível, de forma a aprimorar a experiência do usuário neste âmbito, visto que o mesmo não necessitará realizar vários passos, como exportar a peça e realizar *upload* em vários portais, para realizar tal atividade.

1.7 Delimitação

Em relação ao assunto, este trabalho busca contemplar três áreas: Design, Computação e Marketing. A área de Design será a principal deste trabalho e limita-se à subárea de UI/UX Design, com a aplicação de pesquisas de usabilidade, sendo possível estudar a eficiência da navegação por parte do usuário, além da apresentação de diagramas para cada caso de uso, assim como a exposição das especificações utilizadas pela interface da plataforma, explicando as decisões acerca dos protótipos realizados. Já a área de Computação será auxiliar na realização deste trabalho, com a apresentação do conceito de Arte Generativa, de especificações técnicas acerca das tecnologias utilizadas por este projeto, assim como a estrutura de dados aplicada à plataforma, sendo possível visualizar como todas as entidades implementadas se relacionam. A área de Marketing, também auxiliar a este projeto, busca ser limitada à análise de negócio, com a concepção de uma matriz SWOT e de um modelo de negócio em *Business Model Canvas*.

Em relação à pesquisa, referente à amostra, serão realizadas avaliações de usabilidade acerca da plataforma por meio de dois questionários abertos ao público, mas priorizando alunos do Curso de Design, os quais contemplam um dos públicos-alvo do projeto. Sendo o primeiro questionário realizado após a finalização do MVP (Mínimo Produto Viável), e o segundo realizado após ajustes realizados com base nas sugestões enviadas pelos participantes.

Após compilar todos os resultados e referências sobre o assunto, o ponto de vista da análise geral acerca da plataforma e dos problemas que a mesma busca solucionar contemplará artistas e profissionais da área de Design.

1.8 Estrutura de pesquisa

O presente trabalho está dividido nestas seções: Introdução, Referencial Teórico, Metodologia, Etapas, Análise de Similares e Projeto.

A introdução visa orientar ao leitor sobre a contextualização do trabalho e os problemas que o referido projeto busca resolver, além de definir, de maneira mais detalhada, quais são os objetivos e as restrições do referido projeto.

A seção de referencial teórico visa apresentar, de maneira minuciosa, o assunto de arte generativa, embasando-se em livros, artigos e sites especializados com o objetivo de servir como guia em relação ao ramo, de forma a explanar a história do mesmo, o funcionamento da referida tecnologia e como artistas da atualidade estão usando-a, além de questões éticas sobre o assunto. Também, a mesma seção trata dos assuntos de interface gráfica do utilizador, assim como uma introdução acerca do desenvolvimento web, sendo dividido em *front-end*, *back-end*, *Unified Modeling Language* (UML) e Diagrama de Entidade e Relacionamento.

A metodologia busca especificar quais serão os métodos aplicados, de maneira a discorrer sobre como será feita a produção, análise e validação das características oriundas do projeto.

A análise de similares tem o propósito de explicar, de maneira detalhada, acerca das alternativas já existentes na comunidade, sendo dividida em duas subseções: geradores e modelo de negócio. Esta divisão ocorre por conta do projeto possuir a proposta de aplicar uma estrutura de negócio rentável à área de arte generativa.

A seção de projeto tem o desígnio de listar escopo, regras de negócio, especificações técnicas da interface (*sketches* e *mockups*) e da plataforma (tecnologias utilizadas e estrutura de dados), processo de produção de todas as características do projeto, além da pesquisa sobre a usabilidade da plataforma, sendo possível descobrir se os usuários estão satisfeitos com os recursos que a plataforma disponibiliza, assim como quais pontos podem ser melhorados. Além disso, nesta seção se detalham fatores oriundos da gestão do negócio, com o uso de uma matriz SWOT e de um *Business Model Canvas*.

A última seção, de conclusão, possui o intento de realizar considerações finais acerca de como o projeto foi recebido pelos usuários com base nas pesquisas realizadas, além de sintetizar quais contribuições o referido método gerou para a área em estudo.

2 REFERENCIAL TEÓRICO

2.1 Arte generativa

2.1.1 Conceito

A arte generativa é realizada quando o artista decide aplicar algum tipo de sistema que automatiza o processo de criação para o mesmo, ou seja, o autor terceiriza as suas decisões para um algoritmo, não precisando obrigatoriamente ser computacional (BODEN; EDMONDS, 2009, p. 3), que concretizará tal arte por meio de um conjunto de instruções executado de maneira iterativa (GALANTER, 2016, p. 154). Para a geração de tal arte, a mesma precisa ter uma proceduralidade, devendo seguir uma classificação de ordenamento, podendo ser considerada ordenada (GALANTER, 2016, p. 158–159), a qual pode ser baseada, por exemplo, em fórmulas matemáticas, assim como desordenada (GALANTER, 2016, p. 160–161), a qual se baseia puramente na aleatoriedade, não necessitando obrigatoriamente que a fonte de aleatoriedade venha do computador, mas o lançamento de dados ou a associação a algum fenômeno natural, por exemplo. Além disso, é possível combinar, em diversos graus, tanto a ordem quanto a desordem em uma peça de arte generativa, sendo possível criar, com a combinação de equações com aleatoriedade, assim como o uso, por exemplo, de referências provenientes da biologia, uma obra com relevante complexidade (GALANTER, 2016, p. 161–166).

2.1.2 História

A arte generativa, embora esteja passando uma maior difusão atualmente por conta da evolução da Computação e do aumento de disponibilidade de ferramentas focadas para este propósito, é um conceito que começou a ser aplicado muito antes do progresso tecnológico presenciado, pois existem relatos da aplicação desta forma de arte por volta de 77.000 anos atrás por meio da execução manual de um algoritmo que gerava um padrão geométrico (GALANTER, 2016, p. 158–159).

Por meio do conhecimento da formação de elementos visuais padronizados ao longo da história da humanidade, tal técnica foi aplicada na produção de tecidos, sendo primeiramente exercida com o auxílio de teares manuais. No entanto, no início do século XIX, em meio à revolução industrial, foi possível realizar a automatização da formação de tais padrões com a invenção (figura 1) de Jacquard Loom, que propôs uma nova forma para gerar os elementos visuais mediante uma interface que permitia que o usuário pudesse inserir cartões perfurados (figura 2) que ditavam os padrões visuais desejados, os quais seriam aplicados no tecido a ser produzido. Além disso, vale ressaltar que a inovação dos cartões perfurados como um formato de entrada de dados foi futuramente utilizada na Computação. (GALANTER, 2016, p. 151).

Figura 1: Máquina de Loom



Fonte: Imagem retirada de <https://www.nms.ac.uk/explore-our-collections/stories/science-and-technology/jacquard-loom/>. Acesso em 21 de Junho de 2023

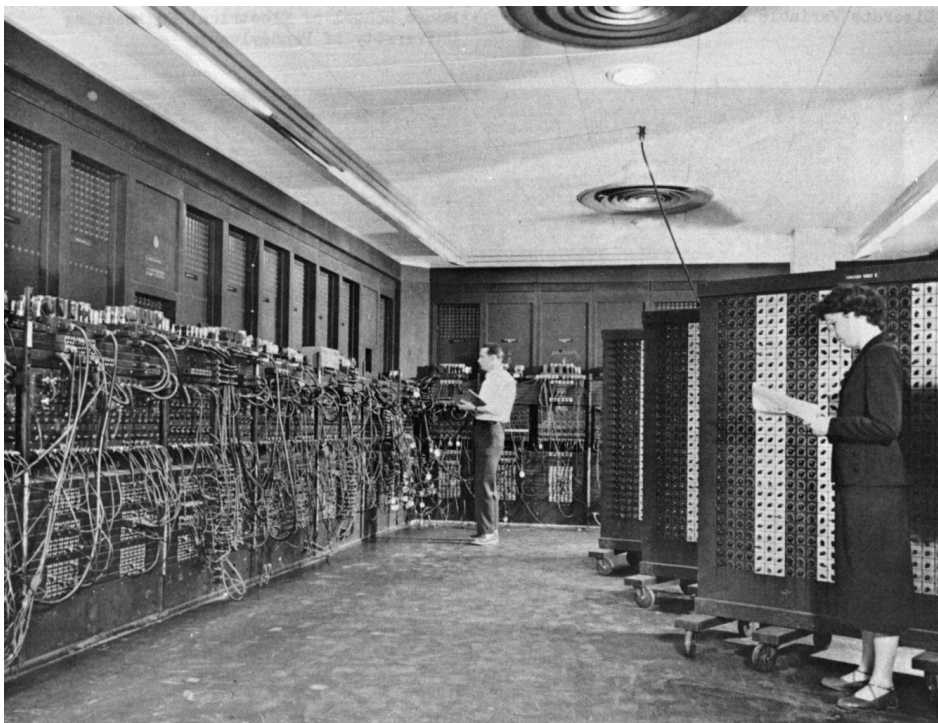
Figura 2: Cartões perfurados compatíveis com a máquina de Loom



Fonte: Imagem retirada de <https://www.nms.ac.uk/explore-our-collections/stories/science-and-technology/jacquard-loom/>. Acesso em 21 de Junho de 2023

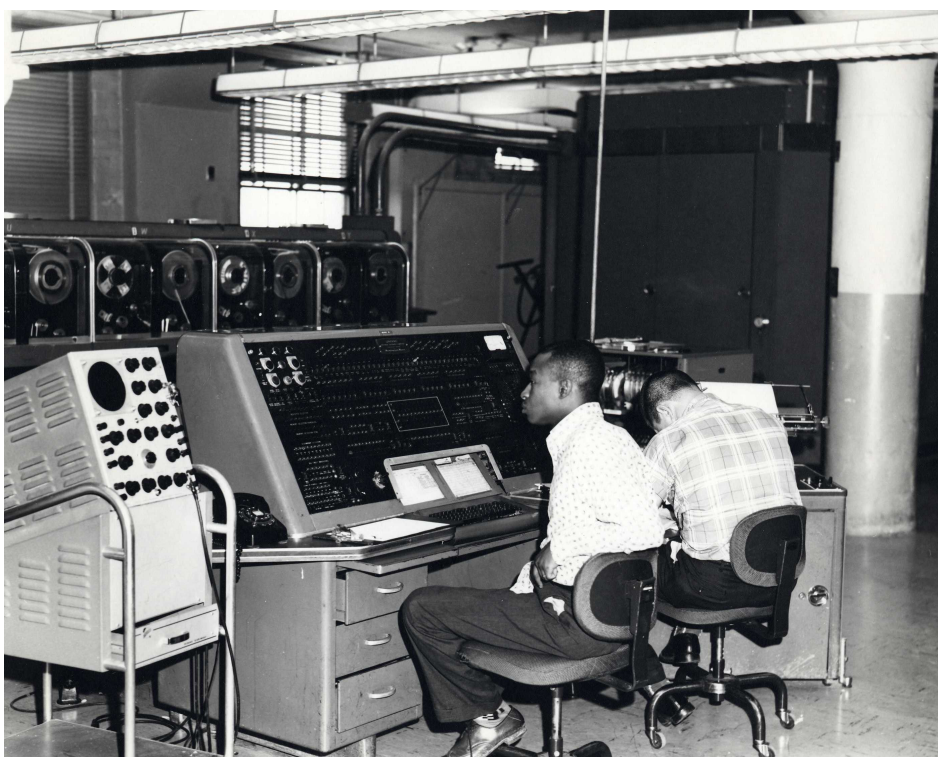
Com a criação do ENIAC (figura 3), o primeiro computador digital, em 1946 e, então, do UNIVAC (figura 4), o primeiro dispositivo computacional disponibilizado comercialmente, em 1951 (HOPE; RYAN, 2014, p. 50–51), a área da Computação, atualmente indispensável para a produção de arte generativa e a qual tinha como conceito, até então, a máquina analítica de Charles Babbage conceituada em 1834, mas que a sua produção foi inviabilizada por conta de ser uma máquina totalmente mecânica (TANEMBAUM; WOODHULL, 2006, p. 6), observou grandes progressos ao decorrer das décadas. Antes do surgimento dos microcomputadores nos anos 1980, o acesso à tal tecnologia era restrito por conta dos computadores serem, até então, uma ferramenta muito onerosa nas questões financeiras e espaciais, além de exigirem conhecimento técnico significativo para a sua operação, fazendo com que os pioneiros da arte generativa eletrônica fossem, em sua maioria, engenheiros associados a grandes laboratórios, como a Bell Labs (HOPE; RYAN, 2014, p. 51). Mesmo com estas restrições, algumas ferramentas voltadas para a arte foram criadas, como o BEFLIX, *software* de criação de animações (REAS; MCWILLIAMS; LUST, 2010, p. 22–23), e o *Sketchpad* (figura 5), dispositivo criado em 1963 por Ivan Sutherland que permitia o desenho em tela com o uso de uma caneta específica (REAS; MCWILLIAMS; LUST, 2010, p. 29).

Figura 3: ENIAC em funcionamento



Fonte: U.S. Army, c. 1949-1955

Figura 4: UNIVAC em funcionamento



Fonte: United States Census Bureau, c. 1960

Figura 5: Sketchpad demonstrado por Ivan Sutherland



Fonte: Autoria desconhecida. c. 1963. Imagem retirada de <https://www.computerhistory.org/collections/catalog/102652182>. Acesso em 21 de Julho de 2023

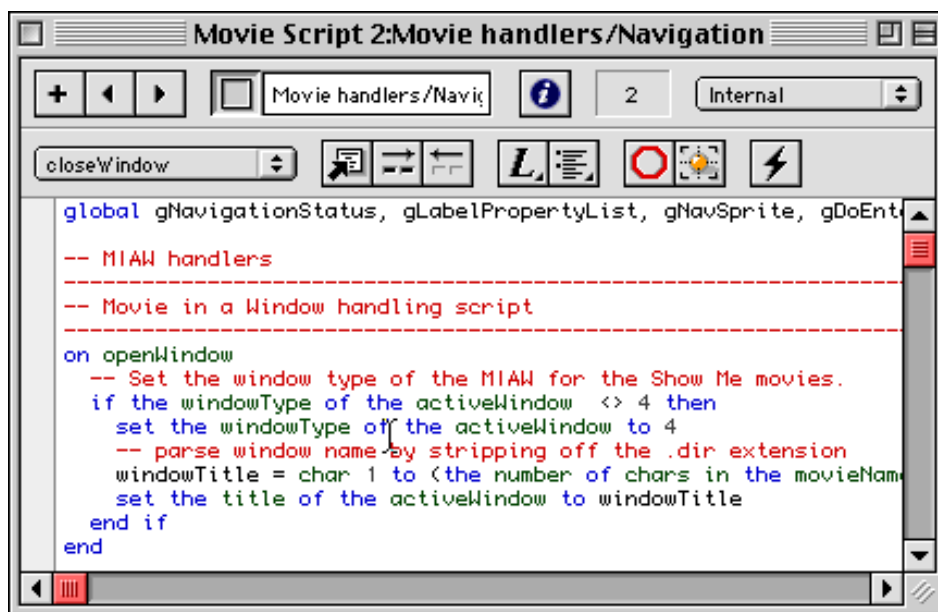
Por meio da concepção do microprocessador, o qual comportava milhares de transistores em um espaço muito pequeno (TANEMBAUM; WOODHULL, 2006, p. 14), houve a criação do computador pessoal, sendo os primeiros dispositivos deste tipo comercializados entre os anos 1980 e 1981 (figura 6) (HOPE; RYAN, 2014, p. 51), sendo uma tecnologia que tornou a Computação acessível ao público em geral, incluindo artistas, por possuir um preço bem inferior em relação às máquinas anteriores, que eram direcionadas às universidades e empresas (TANEMBAUM; WOODHULL, 2006, p. 14). Desta forma, com a maior difusão desta tecnologia, também houve o início do desenvolvimento de ferramentas destinadas a facilitar o ofício dos artistas por meio da automatização com algoritmos, como a criação da linguagem de programação Lingo para o *software* Adobe Director (figura 7) em 1988 (REAS; MCWILLIAMS; LUST, 2010, p. 23).

Figura 6: HP-85, computador lançado em 1980



Fonte: Imagem retirada de <http://computermuseum.informatik.uni-stuttgart.de/dev/hp85/>. Acesso em 16 de Junho de 2023

Figura 7: Lingo em Adobe Director



Fonte: Imagem retirada de <https://users.cs.cf.ac.uk/dave/Multimedia/node63.html>. Acesso em 16 de Junho de 2023

Com a notável evolução dos computadores ocorrida nos anos 1990 até o presente, em meio à popularização desta tecnologia, surgiu uma ampla criação de ferramentas de criação de arte generativa, predominantemente de código aberto, que conseguiu difundir aos artistas conceitos de automatização e geração de peças de arte, possibilitando a criação de uma grande comunidade interessada nesta vertente e que está constantemente implementando novos conceitos (GALANTER, 2016, p. 163). Para mais informações sobre as ferramentas de arte generativa existentes, ver seção 4.1.

2.1.3 Questões relevantes

Embora a arte generativa seja um método que possibilite a produção de resultados inéditos em relação à abordagem tradicional, pois, como citado na seção 2.1.1, o artista possui a opção de automatizar todas as suas decisões acerca da criação da peça, ela possui questões que discutem a legitimidade desta técnica, de forma a dificultar a definição se o uso da mesma pode ser considerado arte. Galanter (2016) cita vários fatores que contribuem para este questionamento: autoria, intenção, unicidade, autenticidade, dinâmica, pós-modernidade, código, criatividade e significado.

A questão da autoria é referente à indefinição do criador da arte generativa, visto que tal técnica necessita de um algoritmo para a sua concepção, de forma que existe a dúvida se o autor é o desenvolvedor do algoritmo, sendo computacional ou não, ou o artista que definiu os parâmetros de tal algoritmo, para que então houvesse a geração de uma composição com os elementos visuais definidos pelo mesmo (GALANTER, 2016, p. 166–167).

A intenção busca definir a motivação do artista a tomar certas decisões em uma obra de arte, algo frequente em uma produção artística realizada de maneira manual. Porém, ao delegar a criação da peça a um algoritmo, há a dúvida se existe algum processo de formação de decisão por parte do artista, de forma a dificultar a definição do escopo da obra de arte, se é apenas considerado o produto do algoritmo, o algoritmo em si, ou até mesmo a máquina que realizou o processo de formação desta obra (GALANTER, 2016, p. 167–168).

O fator da unicidade se associa à impossibilidade de replicação no meio artístico. Segundo Galanter (2016), pinturas, por exemplo, produzidas de maneira totalmente manual possuem a agregação de item raro, visto que é praticamente impossível de reproduzir, de maneira exata, todas as pinceladas postas na obra original. No entanto, com a arte generativa, um artista pode gerar quaisquer tipos de formas aleatórias quando desejar, as quais, também, podem ser de difícil replicação. Logo, é criado o conceito da quantificação da produção de obras completamente singulares, algo inédito para o meio artístico (GALANTER, 2016, p. 168–169).

A autenticidade está relacionada à expressividade do artista, na qual, utilizando métodos manuais, enfatiza as suas emoções ao construir uma obra de arte. No entanto, uma das críticas da utilização da arte generativa é que o sistema provedor da automatização não é capaz de expressar emoções ao criar uma obra, visto que há apenas a execução de instruções que, adicionalmente, possuem restrições, muito provavelmente em forma de números ou lógica booleana, impostas pelo artista em forma de parâmetros (GALANTER, 2016, p. 169).

A dinâmica é uma questão relacionada à validação se algo é definido como arte generativa ou não dependendo de sua apresentação. Neste caso, há divergências se uma peça só pode ser considerada generativa caso seu estado esteja em constante transformação ou não (GALANTER, 2016, p. 169–170).

A questão da pós-modernidade é inerente às questões ideológicas acerca da criação de peças artísticas na contemporaneidade. Logo, é ressaltado que a arte generativa possibilita a isenção de vieses pelo fato da maior parte da criação da arte ser feita por uma máquina, tal desprovida de qualquer contextualização externa e a qual apenas estará executando um algoritmo preestabelecido pelo autor. Além disso, é citado que este modelo de arte propõe novos conceitos de autoria, se encaixando no contexto do movimento pós-modernista (GALANTER, 2016, p. 170–171).

O questionamento da arte generativa em relação ao código é sobre a definição específica de onde estaria o trabalho artístico diante do aspecto colaborativo encontrado em comunidades desta área, pois, com a difusão de ferramentas como o Processing, houve o aumento significativo do contingente dos usuários que utilizam tecnologia generativa para criar obras de arte, também aumentando a prática colaborativa entre tais por meio do código aberto, fomentando a discussão sobre, diante deste processo, qual seria o produto final que significaria uma arte (GALANTER, 2016, p. 171)?

Em relação à criatividade, tal aspecto é discutido visto que algoritmos executados em computadores, na forma

como são apresentados hoje, não possuem nenhum tipo de discernimento acerca de referências para se criar uma boa peça artística que, embora haja a possibilidade de bons resultados com tal técnica, o sistema não terá a criatividade necessária por conta de sua inconsciência (GALANTER, 2016, p. 172).

Acerca do quesito de significado, tal fator é questionado no tipo de arte em estudo devido ao mesmo ter a possibilidade de divergência entre as técnicas adotadas no algoritmo e a interpretação percebida pelo público, pois pode-se optar por uma implementação do algoritmo de geração que pode não estar associada à temática que o artista busca levar (GALANTER, 2016, p. 173).

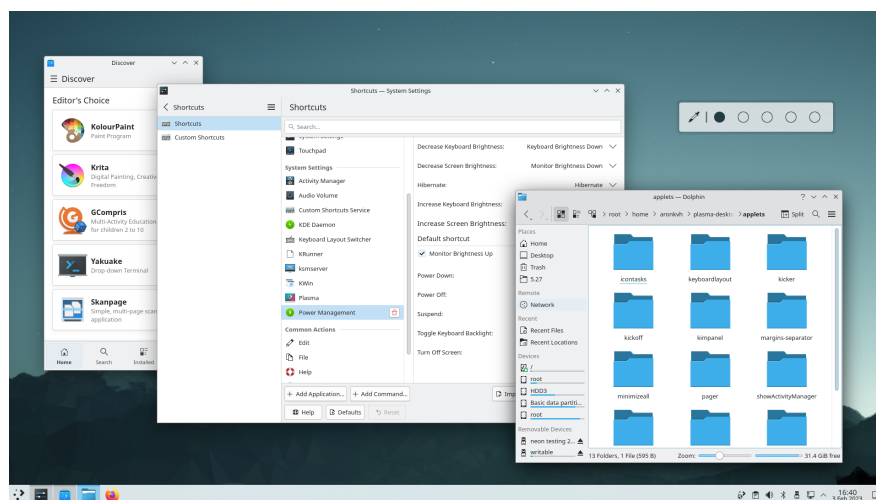
2.2 Interface Gráfica do Utilizador

2.2.1 Conceito

A Interface Gráfica do Utilizador é um recurso utilizado em *softwares* que busca representar graficamente componentes interativos, como janelas, menus e botões, que estão associados a eventos, de maneira que promova a alteração do estado da interface da aplicação por conta de alguma ação realizada pelo usuário (MARTINEZ, 2011, p. 119). Este conceito pertence à grande área chamada de Interface Humano-Computador (IHC), na qual define que, para haver troca de informações entre o utilizador e a máquina, devem haver dispositivos de entrada (*input*), que buscam fornecer os dados provenientes do usuário e levam em consideração fatores de ergonomia, como mouse e teclado; e de saída (*output*), responsáveis pela exibição dos dados provenientes da máquina ao usuário, como monitores (TAN, 2022).

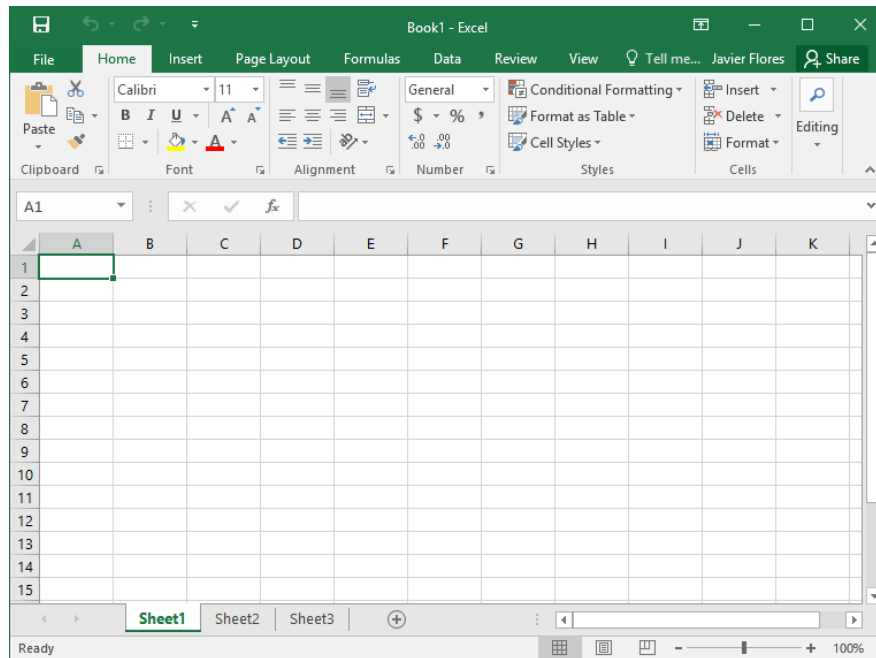
Segundo Martinez (2011), interfaces gráficas podem ser divididas em quatro tipos: baseada em menus e diálogos, existentes em ambientes de desktop, sendo o Plasma Desktop (figura 8) um exemplo; tabulares, encontrados em *softwares* como o Microsoft Excel (figura 9) com o uso de células que permitem o *input* de dados para posterior processamento; bloco de notas, que combina as capacidades de edição de texto com a possibilidade de execução de código e visualização do *output* do algoritmo na própria interface, sendo possível encontrar tal funcionalidade em aplicações como o Jupyter Notebook (figura 10); e interfaces web (figura 11), que são disponibilizadas na internet e são produzidas utilizando técnicas de front-end e web design (ver seção 2.4.1).

Figura 8: Plasma desktop, uma interface de janelas e diálogos



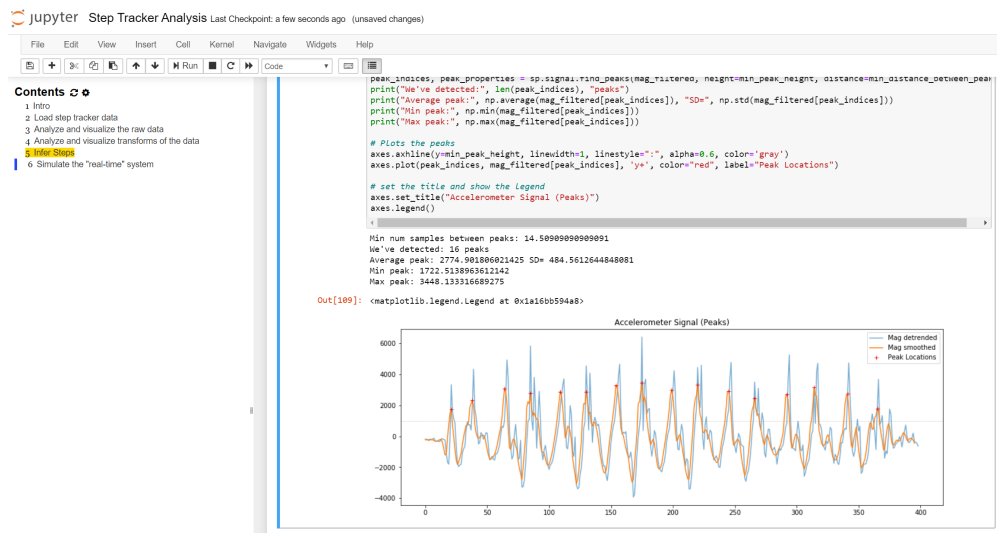
Fonte: Imagem retirada de <https://kde.org/>. Acesso em 22 de Junho de 2023

Figura 9: Microsoft Excel, uma interface tabular



Fonte: Imagem retirada de <https://edu.gcfglobal.org/en/excel/getting-started-with-excel/1/>. Acesso em 22 de Junho de 2023

Figura 10: Jupyter Notebook, uma interface de bloco de notas



Fonte: Imagem retirada de <https://makeabilitylab.github.io/physcomp/signals/jupyter-notebook.html>. Acesso em 22 de Junho de 2023

Figura 11: Site da Universidade Federal do Ceará, uma interface web



Fonte: Captura de tela de <https://www.ufc.br/>. Acesso em 22 de Junho de 2023

2.2.2 História

O início da história desta área de estudo é remetido ao artigo publicado no *The Atlantic* em 1945 por Vannevar Bush (MARTINEZ, 2011, p. 119–120), engenheiro e administrador do órgão do governo americano de pesquisa científica e desenvolvimento (BUSH, 1945), com a concepção de uma ferramenta chamada de Memex, que englobava um conceito que, utilizando teclado, câmeras e monitores (MARTINEZ, 2011, p. 120), permitisse a armazenamento e acesso de dados utilizando microfilme (BUSH, 1945). Após este evento, são listados pioneiros como Ivan Sutherland (citado na seção 2.1.2), que conceituou o *Sketchpad* em 1963 com o conceito de interação de interface gráfica utilizando uma caneta; e Douglas Engelbart, o qual produziu em conjunto com a sua equipe de Stanford uma máquina que possibilitasse a interação utilizando um mouse (MARTINEZ, 2011, p. 120), a qual sua apresentação foi denominada de “*Mother of All Demos*”, demonstrando funcionalidades como edição de textos e exibição de janelas (SRI, 2012).

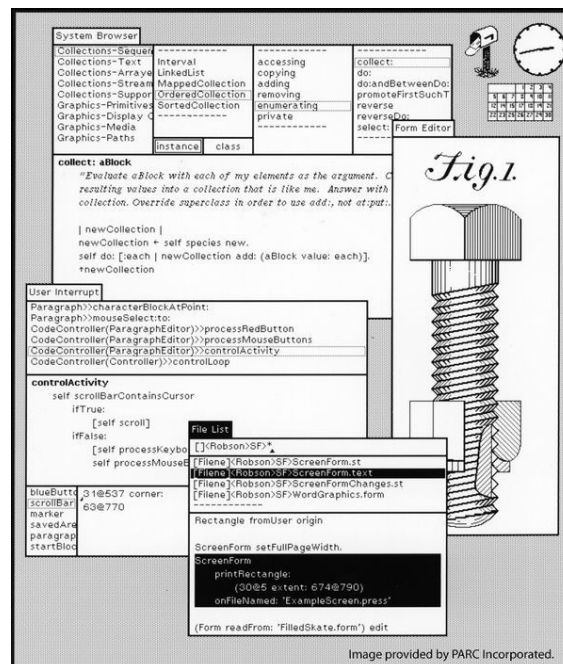
Com a consolidação promovida por invenções anteriores foi desenvolvido, com uma parte do time de Douglas Engelbart, o computador Alto (figura 12) na Xerox Palo Alto Research Center (PARC), concluído em 1973 (BROCK, 2023) e direcionado para o uso na academia (CHM, 2023b), sendo a primeira máquina a possuir uma interface gráfica que permitia a sobreposição de janelas (figura 13), assim como o processamento de texto utilizando o conceito WYSIWYG (*What you see is what you get*) (figura 12) (CHM, 2022b), sendo a primeira aplicação a utilizar uma interface gráfica desenvolvida em 1977 (JANSEN, 1998, p. 23) e, então, com tal paradigma estabelecido, a Xerox lançou em 1981 uma variação comercial do Alto denominada de Xerox Star, mas que não obteve sucesso comercial pelo preço da máquina ser bastante acentuado, além de possuir problemas de performance (JANSEN, 1998, p. 23).

Figura 12: Xerox Alto executando interface gráfica



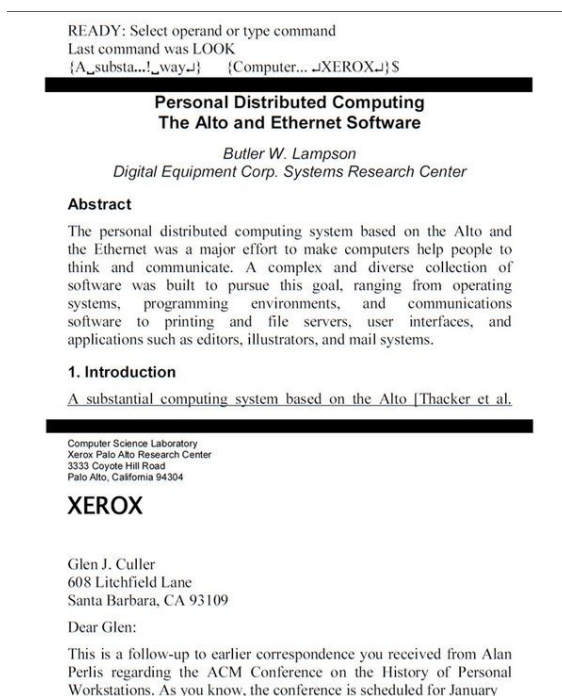
Fonte: Imagem retirada de <http://www.righto.com/2017/10/the-xerox-alto-smalltalk-and-rewriting.html>. Acesso em 21 de Junho de 2023

Figura 13: Interface do Smalltalk para o Xerox Alto



Fonte: PARC (Palo Alto Research Center, Incorporated), c. 1980

Figura 14: Bravo, um processador de texto WYSIWYG para o Xerox Alto



Fonte: PARC (Palo Alto Research Center, Incorporated), c. 1974

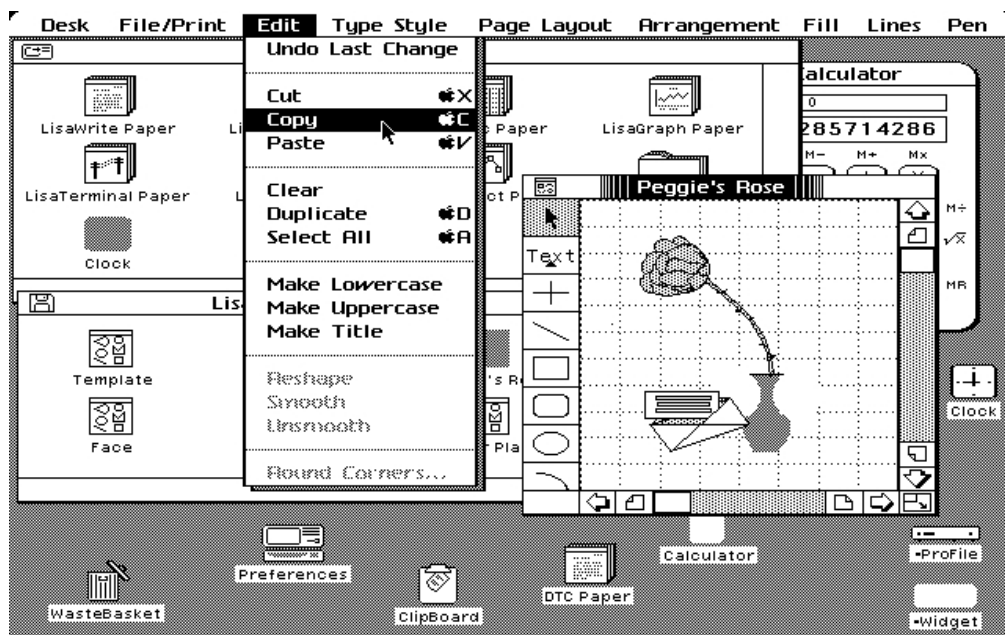
Em 1979, Steve Jobs, em conjunto com a sua equipe da Apple, realizou uma visita ao PARC e, após observar o novo paradigma de interface de usuário para computadores, viu tal como uma oportunidade de negócio (CHM, 2022a) e, desta forma, Jobs decidiu contratar vários pesquisadores da Xerox, que posteriormente trabalharam nos projetos dos computadores Lisa (figuras 15, 16), lançado em 1983, e Macintosh (figuras 17, 18), introduzido em 1984, de maneira a aprimorar o paradigma consolidado na Xerox nos produtos da Apple com a inclusão de novos elementos interativos na interface (CHM, 2022a; MARTINEZ, 2011, p. 120).

Figura 15: Apple Lisa 1



Fonte: Aurora Tucker, Computer History Museum

Figura 16: Interface do Lisa 2



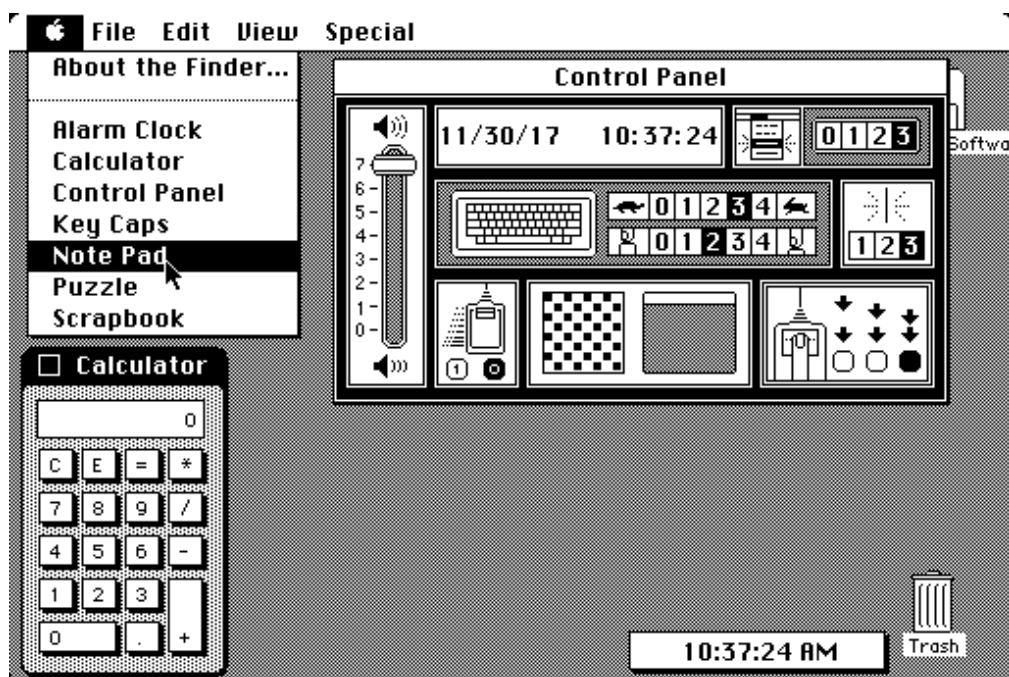
Fonte: David T. Craig, c. 1984

Figura 17: Apple Macintosh



Fonte: Douglas Fairbairn, Computer History Museum

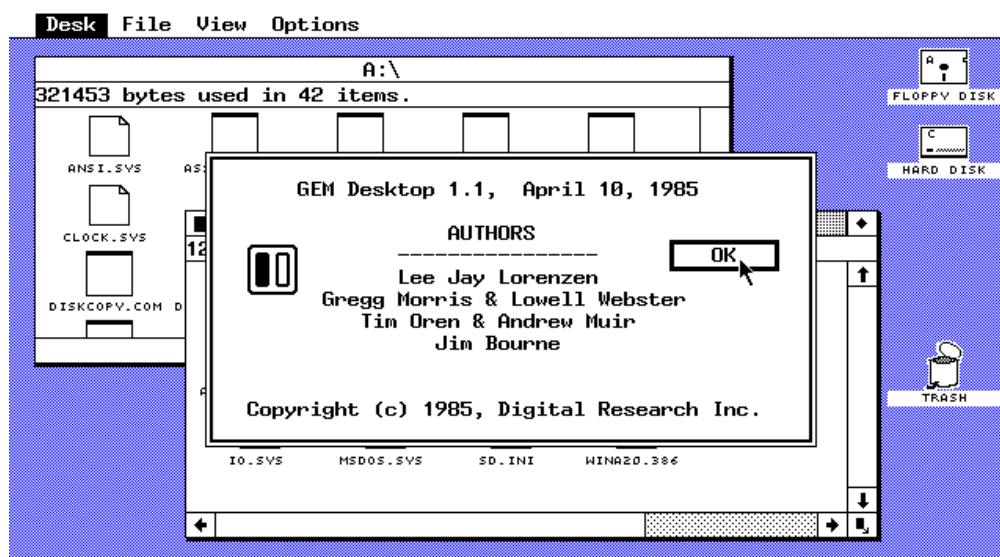
Figura 18: Interface do Mac OS 1.0



Fonte: Imagem retirada de <https://winworldpc.com/product/mac-os-0-6/system-1x>. Acesso em 21 de Junho de 2023

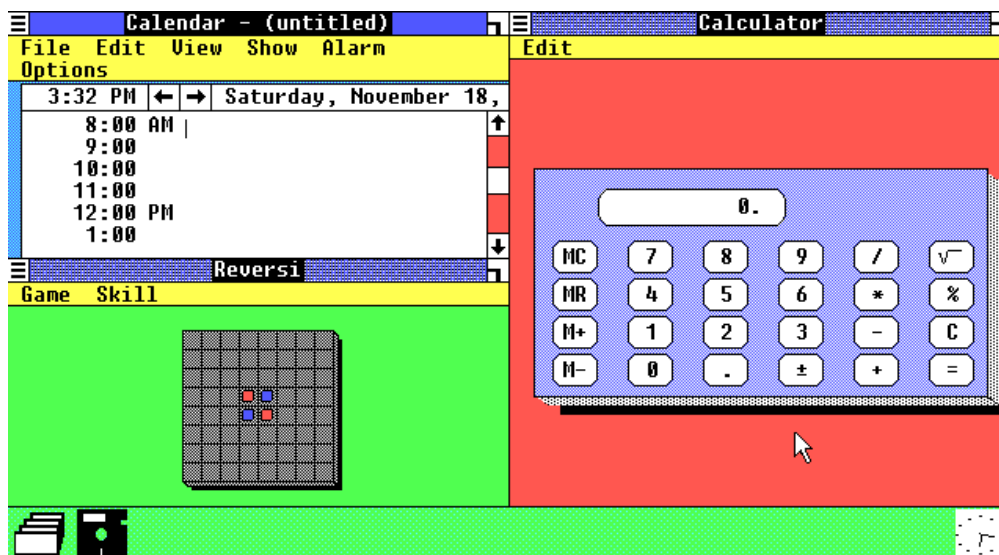
Com a influência da interface gráfica desenvolvida pela Apple, surgiram implementações nos anos 1980 para outras plataformas, sendo o IBM PC um exemplo, utilizando um paradigma similar de interface, como o lançamento do GEM (figura 19) e do Microsoft Windows (figura 20) em 1985 (CHM, 2022a; MARTINEZ, 2011, p. 120) e, mesmo com o aparecimento de várias interfaces alternativas nos anos 1980, se observou a consolidação da Apple e da Microsoft neste ramo ao decorrer do tempo (REIMER, 2005).

Figura 19: Interface do ambiente GEM 1.1



Fonte: Imagem retirada de <https://winworldpc.com/product/gem/1x>. Acesso em 21 de Junho de 2023

Figura 20: Interface do ambiente Microsoft Windows 1.01



Fonte: Imagem retirada de <https://winworldpc.com/product/windows-10/101>. Acesso em 21 de Junho de 2023

2.2.3 Métodos de prototipagem

2.2.3.1 Protótipo de baixa fidelidade

Protótipos de baixa fidelidade, por serem de rápida produção e modificação (PERNICE, 2016), são geralmente produzidos no início do projeto com o intuito de definir requisitos para o produto, estabelecer um layout inicial e apresentar conceitos iniciais de funcionalidade (RUDD; STERN; ISENSEE, 1996, p. 79–80) por meio da concepção de interfaces estáticas que podem se apresentar como *sketches*, desenhados à mão livre, ou *wireframes* (PERNICE, 2016), produzidos em *softwares* de edição vetorial em geral, como o Adobe Illustrator, ou em aplicações especializadas, como o Axure (BROWN, 2011, p. 200). No entanto, este tipo de método pode não ser eficiente para detectar problemas de usabilidade em uma sessão de validação, assim como a implementação direta de uma interface produzida por meio de protótipo de baixa fidelidade ser mais difícil de ser implementada, pois sem diretrizes visuais aplicadas na prototipação, pode ocorrer da equipe de desenvolvimento fazer uso de práticas contraindicadas de Design (RUDD; STERN; ISENSEE, 1996, p. 80–81).

2.2.3.2 Protótipo de alta fidelidade

Protótipos de alta fidelidade, embora possuam produção financeiramente e temporalmente onerosa (RUDD; STERN; ISENSEE, 1996, p. 82), permitem a aplicação de diretrizes de Design definidas no projeto, como paleta de cores, hierarquia de elementos, tipografia, além de componentes de interface avançados (PERNICE, 2016) e, por permitirem interatividade, sendo clicáveis, e possuírem um maior detalhamento visual em comparação com a alternativa anterior, esta é uma solução adequada para testes de usabilidade, de forma com que avaliadores possam detectar problemas de interface mais eficientemente e com antecipação, além de facilitar a elaboração de críticas construtivas à interface, podendo ser documentadas para posterior melhoria (RUDD; STERN; ISENSEE, 1996, p. 81–82). Também, este tipo de prototipação evita que a equipe de desenvolvimento possua dúvidas inerentes à estética da interface, evitando decisões arbitrárias de Design e ocasional retrabalho por parte dos mesmos (RUDD; STERN; ISENSEE, 1996, p. 81).

2.2.4 Metodologias de análise de usabilidade

As seções abaixo listam e especificam as metodologias de validação de usabilidade utilizadas neste trabalho, as quais são citadas na seção 3.1.3.

2.2.4.1 Avaliação heurística de Nielsen

A avaliação heurística de Nielsen foi criada por Jakob Nielsen e Rolf Molich em 1990 (NIELSEN, 2010) com o propósito de servir como um sistema de validação de uma interface gráfica verificando se a mesma respeita as dez diretrizes de usabilidade, ao invés outras ferramentas que possuem uma quantidade muito maior de fatores heurísticos a serem verificados, permitindo que a avaliação de Nielsen possua uma aplicação mais ágil e simplificada (NIELSEN, 1993, p. 155).

Esta avaliação heurística pode ser aplicada utilizando um ou mais avaliadores, sendo a eficiência da examinação da interface de acordo com os padrões estabelecidos diretamente proporcional à quantidade e à qualificação dos examinadores (NIELSEN, 1993, p. 155–156), pois, quando referido teste é aplicado com diversos avaliadores, é possível que uma pessoa consiga detectar mais falhas na interface do que outra, sendo uma ótima solução para otimizar a eficiência da análise caso hajam pessoas integrantes à equipe de projeto disponíveis para isto (NIELSEN, 1993, p. 156). No entanto, deve-se possuir cautela na escolha da quantidade de integrantes disponíveis para a avaliação visto que, embora o aumento de avaliadores possa diminuir a carga de trabalho para cada um e, desta forma, acelerar a execução da examinação, existe a possibilidade de haver uma sobrecarga de documentação gerada no final do teste, fazendo com que isto acarrete em uma maior dificuldade de interpretação dos resultados pela equipe encarregada (NIELSEN, 1993, p. 157).

Em relação às dez diretrizes de usabilidade, citam-se estas: visibilidade do estado no qual o sistema se encontra; correspondência entre o sistema e o mundo real; existência de controle e liberdade do usuário; presença de consistência e padrões; prevenções de erros; reconhecimento em vez de memorização; flexibilidade e eficiência no uso; estética e design minimalista; possibilidade que o usuário reconheça, diagnostique e se recupere de erros; presença de ajuda e documentação (NIELSEN, 2010).

A visibilidade do estado no qual o sistema se encontra (*feedback*) está relacionada à quantidade de informações úteis que uma interface pode prover no momento em que o usuário opta por realizar uma operação no sistema, o qual pode estar relacionado à exibição de informações pré-operação que podem ser indispensáveis para a decisão final do usuário assim como a possibilidade de representação do progresso de tal operação em tempo real, sendo, desta forma, um elemento indispensável para que não hajam erros em operações que podem ser, possivelmente, irreversíveis (NIELSEN, 1993, p. 134–138).

O quesito de correspondência entre o sistema e o mundo real prioriza a utilização de termos reconhecíveis pelo público-alvo do software ao invés de expressões literais encontradas no código-fonte, a internacionalização da linguagem exibida na interface de usuário, o cuidado com a existência de múltiplas interpretações para um tipo de termo ou frase encontrada na interface e a opção de adotar ícones ao invés de palavras para vários termos, possibilitando o uso de metáforas em forma de elementos gráficos para representar determinada ação. Uma interface que respeita este quesito consegue evitar que o usuário possua uma interpretação errada acerca de uma determinada funcionalidade, de maneira a não confundir-lo em sua navegação, fazendo com que o mesmo consiga achar os recursos desejados de maneira mais eficiente (NIELSEN, 1993, p. 123–129).

A existência de controle e liberdade do usuário em um *software* é inerente à presença de recursos que permitam com que o utilizador possa cancelar ou reverter uma operação da maneira mais simples possível. Logo, o programa deve fornecer atalhos e possuir elementos de interface que explicitam a possibilidade de poder contornar algum procedimento no qual o usuário venha a desistir, de forma com que o mesmo possua uma sensação de controle mediante as operações disponibilizadas (NIELSEN, 1993, p. 138–139).

A presença de consistência em uma interface está relacionada à existência do uso padronizado de elementos gráficos, diagramação e estruturação de ações em um software. Logo, para que o *software* não confunda o usuário e o mesmo realize uma ação indesejada em virtude da falta de padronização encontrada, faz-se necessário que a aplicação possua um sistema que dite as regras gráficas que devem ser tomadas na implementação de cada componente da interface, e como todos os componentes conseguem, de maneira visual, possuir associação entre si mesmos (NIELSEN, 1993, p. 132–134).

O fator de prevenções de erros está ligado à implementação de elementos que busquem prever o erro de um usuário ou a confirmação de uma operação irreversível ou perigosa. Desta forma, por meio de estudos de usabilidade e a documentação dos erros que acontecerem durante o uso, é possível identificar quais são as funcionalidades que o utilizador possui maior tendência ao erro, possibilitando a implementação de melhorias para melhorar a experiência do usuário, como o uso moderado de diálogos de confirmação (NIELSEN, 1993, p. 145–146).

O reconhecimento em vez de memorização possui o propósito de evitar com que usuários passem por uma sobrecarga cognitiva tanto pela falta de convenção das funcionalidades quanto pela indisponibilidade de valores padronizados como *input* para um caso de uso específicos. Neste caso, uma boa prática está relacionada a fornecer ao usuário, no momento no qual o mesmo está em um diálogo que necessita fornecer um *input* para o preenchimento de algum dado ou definição de alguma opção do *software*, opções padronizadas no campo relacionado, como o autopreenchimento da data atual em um campo que esteja neste escopo, de maneira que haja a possibilidade de que o usuário não precise redigitar os dados necessários, fazendo com que o utilizador ganhe tempo e não sofra sobrecarga mental ao utilizar a aplicação (NIELSEN, 1993, p. 129–132). Outro fator que participa desta diretriz heurística é a existência de funcionalidades básicas de manipulação de dados (como copiar e colar) que condizem com a método realizado por programas similares tanto na associação de seu atalho como no comportamento da aplicação ao executar tal funcionalidade (NIELSEN, 1993, p. 131–132).

A flexibilidade e eficiência no uso está associada à presença de métodos para acelerar a utilização de uma aplicação dependendo do nível do usuário. Desta forma, o programa pode fornecer um conjunto de atalhos para que pessoas que já possuem conhecimento significativo sobre o *software* possam utilizá-lo de maneira mais eficiente, de forma com que seja possível economizar tempo em várias operações. No entanto, isso deve enaltecer a flexibilidade do uso da aplicação, para que este programa não precise, obrigatoriamente, que o usuário memorize todos os atalhos, pois, também, é uma boa prática tornar o uso de tal *software* acessível a pessoas mais leigas na utilização do mesmo, de maneira a disponibilizar vários métodos que podem ser utilizados para chegar a um mesmo resultado, o qual o usuário pode optar acerca do qual querará utilizar (NIELSEN, 1993, p. 139–142).

A estética e design minimalista está relacionada à aplicação correta de conceitos de ponderação, posicionamento e paleta de cores nas informações presentes em uma interface gráfica. O uso de uma interface em um *software* pode ser feito de maneira muito mais eficiente caso a mesma exiba apenas elementos úteis para o contexto no qual o usuário se encontra, ou seja, sem informações desnecessárias exibidas no módulo relacionado (NIELSEN, 1993, p. 116), pois o excesso de informações pode gerar confusão e demora do tempo de resposta do utilizador, fazendo com que processos envolvendo a aplicação realizados pelo usuário possam ser mais demorados e, por conta disso, gerar uma maior carga cognitiva ao mesmo (NIELSEN, 1993, p. 120–123). Além disso, o posicionamento correto dos elementos é bastante importante para que não hajam interpretações erradas acerca dos dados, na qual o posicionamento dos itens que possuem contextos relacionados, como o nome do índice e o seu valor, devem se encontrar próximos para evitar que haja confusão por parte do visualizar de tais itens (NIELSEN, 1993, p. 118–119). Outro fator importante está relacionado à escolha correta da paleta de cores, a qual deve possuir simplicidade e, além disso, ser acessível aos usuários com algum tipo de deficiência na distinção de cores, sendo também sugerido que haja uma versão monocromática da interface para sanar alguns transtornos relacionados ao daltonismo (NIELSEN, 1993, p. 118–119).

A diretriz da possibilidade que o usuário reconheça, diagnostique e se recupere de erros está ligada à concisão e informações úteis que possam aparecer em mensagens de erro. Logo, esta diretriz avalia se as mensagens conseguem explicar ao usuário um diagnóstico conciso acerca da causa do erro e a possível solução para o mesmo para que o usuário não fique confuso no momento de erro de um programa, assim como tais mensagens de erros possam servir para que o utilizador consiga aprender a correta utilização da aplicação, enquanto frases vagas de diagnóstico não seriam uma boa prática por conta de tais não direcionarem o usuário a uma solução para as exceções ocorridas no *software*, levando a problemas de usabilidade (NIELSEN, 1993, p. 142–145).

A presença de ajuda e documentação possui o intuito de auxiliar o usuário a entender melhor determinada funcionalidade no qual o mesmo possui dúvidas. Logo, tal documentação necessita de uma interface que permita uma busca rápida acerca do tópico no qual o utilizador deseja consultar (NIELSEN, 2010), além de possuir uma redação simples, evitando jargões (NIELSEN, 1993, p. 150–152), e, além disso, como boa prática, a documentação deve oferecer uma estrutura passo-a-passo para procedimentos importantes e, se possível, diagramas acerca dos mesmos (NIELSEN, 1993, p. 152).

2.2.4.2 Questionário

A aplicação de questionários, segundo Nielsen (1993), é uma metodologia de avaliação indireta, visto que, neste caso, a avaliação mensura a satisfação do usuário frente à aplicação ao invés do teste propriamente dito da interface. O questionário, como vantagem, possui uma escalabilidade muito boa em relação à quantidade de testes aplicados, visto que o mesmo é um teste bastante objetivo e padronizado para todos os avaliados, permitindo que o mesmo possa ser realizado com várias pessoas, o que possibilita um espaço amostral maior (NIELSEN, 1993, p. 210–211). Este tipo de avaliação possui elementos como *checklists* (para a seleção de múltiplas opções objetivas como resposta), escalas de classificação (para a seleção de uma opção objetiva como resposta), além de perguntas abertas predefinidas que podem coletar *feedbacks* bastante úteis sobre a funcionalidade da aplicação, o que pode proporcionar uma posterior implementação de melhoria no *software* (NIELSEN, 1993, p. 212–213). No entanto, é importante que haja a ponderação do número de perguntas incluídas no questionário, além da qualidade de elaboração e redação das mesmas para que não haja um grau de rejeição acentuado em relação ao mesmo, o que pode comprometer a interpretação dos dados (NIELSEN, 1993, p. 213).

2.3 Ferramentas de modelo de negócio

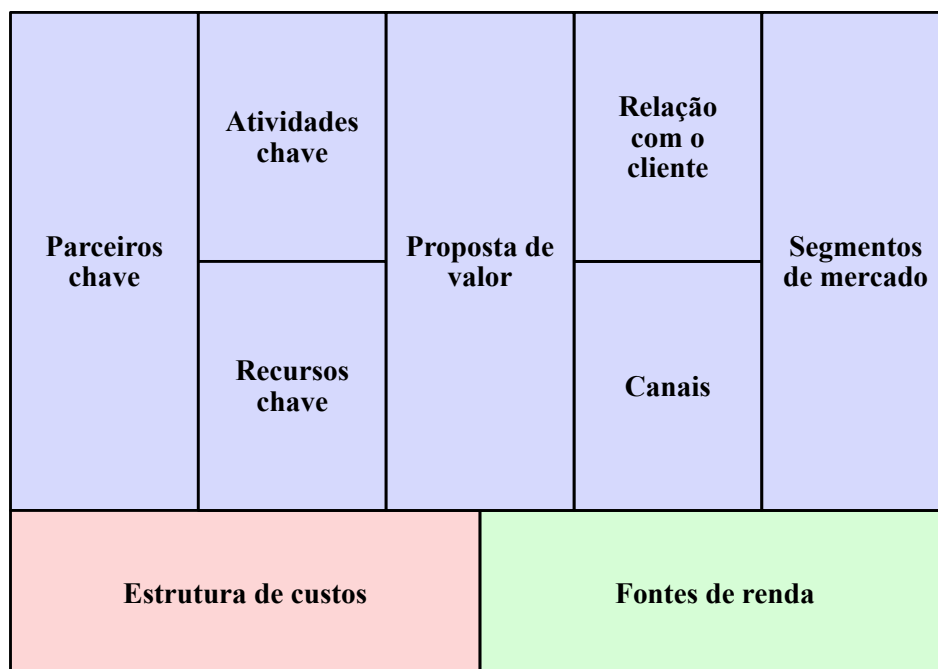
2.3.1 Matriz SWOT

A matriz SWOT é uma ferramenta de análise de Marketing que consiste na listagem de fatores internos e externos ao negócio com o intuito de chegar a uma conclusão sobre a viabilidade da atividade (KOTLER; KELLER, 2016, p. 71–73) e, desta forma, também servir de base para a criação de metas para a empresa (KOTLER; KELLER, 2016, p. 74). Os fatores internos (forças e fraquezas) estão associados à eficiência na qual a corporação executa tal atividade, podendo listar processos que possam estar afetando de maneira positiva ou negativa a rentabilidade da empresa em relação aos seus processos, como o fechamento de parcerias comerciais e o método de vendas utilizado. Já os fatores externos (oportunidades e ameaças) são ligados às questões econômicas no contexto no qual o negócio se insere, podendo, por exemplo, que a quantidade de concorrentes, assim como de matéria-prima ou de demanda por determinado produto possam afetar a lucratividade da empresa (KOTLER; KELLER, 2016, p. 71–73).

2.3.2 Business Model Canvas

O *Business Model Canvas* (figura 21) é uma ferramenta criada por Alexander Osterwalder (SEBRAE, 2021) e utilizada por várias corporações no mundo, possuindo o objetivo de representar fatores de rentabilidade da empresa por meio de uma disposição que promova o fácil entendimento acerca do planejamento desejado, focando em quatro áreas essenciais para o funcionamento de uma organização: clientes, oferta, infraestrutura e viabilidade financeira (OSTERWALDER; PIGNEUR, 2010, p. 15); sendo divididos, no total, em nove fatores que são indispensáveis para um negócio: parcerias-chave, atividades-chave, recursos-chave, proposta de valor, relacionamento com clientes, canais, segmento de clientes, estrutura de custos e fluxo de receita (OSTERWALDER; PIGNEUR, 2010, p. 16–17).

Figura 21: Visão geral do Business Model Canvas



Fonte: Elaborado pelo autor

2.4 Desenvolvimento Web

2.4.1 Front-end

O desenvolvimento *front-end* é inerente à criação de interfaces web utilizando HTML, CSS e Javascript, nos quais as três linguagens possuem finalidades diferentes para a criação do produto final: o HTML é responsável pela estrutura e conteúdo da página, enquanto o CSS realiza a estilização desta página e o Javascript fornece recursos para tornar elementos de interface interativos (COURSERA, 2023a). Além disso, uma das exigências desta área está relacionada à aplicação de boas práticas na escrita de código com o objetivo de maximizar a compatibilidade do site produzido entre diversos navegadores e promover o uso correto de recursos de acessibilidade web já existentes (MDN, 2023b), como a aplicação de tags semânticas ao código, como `<nav>` e `<section>` ao invés de `<div>`, o uso da propriedade `alt`, que pode descrever textualmente uma imagem, e a adoção das propriedades ARIA (*Accessible Rich Internet Applications*), relacionadas à adição de contexto caso o mesmo não possa ser descrito pelas tags semânticas (HENKE, 2019).

Mesmo que o desenvolvedor front-end possua a atribuição de implementação de interfaces, também há o profissional denominado Web Designer, que, embora possa ter o conhecimento sobre linguagens de marcação como HTML e CSS para a construção de páginas, o mesmo possui um maior foco na prototipação com a produção de *sketches*, *wireframes* e *mockups* com base em seus estudos acerca da identidade visual adequada para o site e das pesquisas sobre experiência do usuário para o projeto no qual o mesmo faz parte (SACRAMENTO, 2021).

2.4.2 Back-end

O desenvolvimento *back-end* está associado à construção de código relacionado à renderização dinâmica e tratamento de dados em uma aplicação web, no qual inclui a definição da estrutura de dados, a escrita de lógica, com base na documentação das regras de negócio do projeto, que determina as condições nas quais algum dado deve ser modificado e para qual valor, a elaboração de circunstâncias necessárias para a visualização de uma informação específica no *database* (a obrigatoriedade que o usuário esteja logado para ver seus dados pessoais, por exemplo) (MDN, 2023c) e a implementação de operações básicas de armazenamento dos dados, sendo elas: criar, ler, atualizar e excluir; as quais formam um conjunto denominado de CRUD, acrônimo para *create, read, update and delete* (MDN, 2023a).

Enquanto, no quesito de linguagens disponibilizadas, o desenvolvimento front-end está restrito ao HTML, CSS e Javascript, no back-end é possível utilizar uma grande variedade de linguagens, na qual a escolha pode estar associada às demandas técnicas necessárias para cada tipo de projeto, como Ruby, Python, PHP e C#, havendo também a possibilidade de que possa trabalhar com *frameworks* que agilizam a construção da aplicação (MDN, 2023c), sendo possível automatizar etapas como a criação de páginas e lógica que formam o CRUD associado a uma estrutura de dados, recurso existente no Ruby on Rails, ferramenta utilizada neste projeto, denominada de *scaffolding* (RAILS, 2023). Desta forma, como demonstração, caso seja invocado, no terminal, o comando `rails generate scaffold User name:string email:string`, a *framework* construirá o modelo `User` estruturado com as propriedades `name` e `email`, as quais ambas são do tipo `string`, ou seja, aceitam texto como dado (RUBY-DOC, 2023) e, com isto, a ferramenta irá gerar a página de listagem com todos os usuários (figura 22), além das páginas referentes ao CRUD, com a criação (figura 23), edição (figura 24) e visualização (figura 25) de cada usuário, além da possibilidade de exclusão por meio de um botão disponibilizado.

Figura 22: Rails — Listagem de usuários

The screenshot shows a web application interface for managing users. At the top left, the title 'Users' is displayed in a large, bold, black font. To the right of the title is a blue button with the text 'New user'. Below the title, the details for the first user, Charles, are listed: 'Name: Charles' and 'Email: charles@usuario.com'. Under these details are two light gray buttons: 'Show this user' and 'Edit this user'. A horizontal line separates this section from the details of the second user, João. The details for João are: 'Name: João' and 'Email: joao@usuario.com', also followed by 'Show this user' and 'Edit this user' buttons. Another horizontal line is at the bottom of the visible content.

Fonte: Elaborado pelo autor

Figura 23: Rails — Criação de usuário

The screenshot shows a web application interface for creating a new user. The title 'New user' is displayed in a large, bold, black font at the top. Below the title, there are two input fields. The first is labeled 'Name' and the second is labeled 'Email'. Both labels are in a small, gray font. Below the input fields are two buttons: a blue button labeled 'Create User' and a light gray button labeled 'Back to users'.

Fonte: Elaborado pelo autor

Figura 24: Rails — Edição de usuário

Editing user

Name

Email

[Update User](#)[Show this user](#)[Back to users](#)

Fonte: Elaborado pelo autor

Figura 25: Rails — Visualização de usuário

Name:

Charles

Email:

charles@usuario.com

[Edit this user](#)[Destroy this user](#)[Back to users](#)

Fonte: Elaborado pelo autor

2.5 Unified Modeling Language (UML)

2.5.1 Conceito

UML, hoje em sua versão 2.5², é uma linguagem de modelagem gráfica com a sua versão 1.0 lançada em 1994 criada pela conjunção de métodos de modelagem realizados por Grady Booch, James Rumbaugh e Ivar Jacobson (PILONE; PITMAN, 2005) e coordenada por um consórcio de empresas denominado de *Object Management Group* (OMG) (FOWLER, 2004, p. 1) com o objetivo de padronizar a notação para a concepção de regras de negócio, visto que anteriormente, entre o final dos anos 1980 e o começo dos anos 1990, começaram a surgir várias metodologias de modelagem, logo com notações inconsistentes entre si, de forma a causar dúvidas entre os utilizadores (FOWLER, 2004, p. 7). Desta forma, com a criação da UML, é possível utilizar vários tipos de diagramas a depender das necessidades da atividade de maneira consistente.

Embora a UML seja, predominantemente, utilizada no desenvolvimento e documentação de *softwares*, ela pode ser bastante útil na modelagem de ofícios fora do escopo da Tecnologia da Informação (TI), como na saúde e no comércio (PILONE; PITMAN, 2005) com o uso, em seus processos, de diagramas de atividades, que são similares a fluxogramas, mas que permitem paralelismo de processos (FOWLER, 2004, p. 117).

Em relação à modelagem de projetos de tecnologia por UML, é possível adotar dois tipos de engenharia: progressiva e reversa. No caso da engenharia progressiva, o sistema é modelado antes de ser codificado, enquanto na engenharia reversa são gerados modelos a partir do código-fonte disponível. A finalidade para ambos os casos é a documentação do projeto, porém, na engenharia progressiva, os modelos são utilizados como uma visão geral do sistema ainda não implementado, de maneira a possibilitar a discussão entre a equipe de desenvolvimento acerca dos recursos que estarão presentes na aplicação, assim como orientar o desenvolvedor sobre a estrutura e procedimentos relacionados do *software* a ser implementado. Já na engenharia reversa, a modelagem é gerada por meio de uma ferramenta automatizada, possuindo o propósito de facilitar a visualização, por parte dos desenvolvedores, da estrutura já existente no programa (FOWLER, 2004, p. 2–3). Neste projeto, são utilizados diagramas de atividades e de casos de uso de maneira progressiva, logo, anterior à criação do sistema proposto e servindo para a definição do escopo do projeto.

2.5.2 Diagramas

Os diagramas existentes na UML são classificados em dois tipos: estruturais e comportamentais (PILONE; PITMAN, 2005). Segundo Pilone (2005), os diagramas estruturais estão relacionados à organização de informações ou funcionalidades encontrada em algum componente, como a descrição de quais dados pertencerão a um determinado recurso; enquanto os diagramas comportamentais buscam descrever o que acontece quando uma funcionalidade é executada, como a mudança de estados de entidades que sofrem influência ao decorrer da execução de um processo. A visão geral destes diagramas pode ser consultada na figura 26 e nas tabelas abaixo:

²<http://www.omg.org/spec/UML/>. Acesso em 10 de Junho de 2023

Figura 26: Tipos de diagramas UML

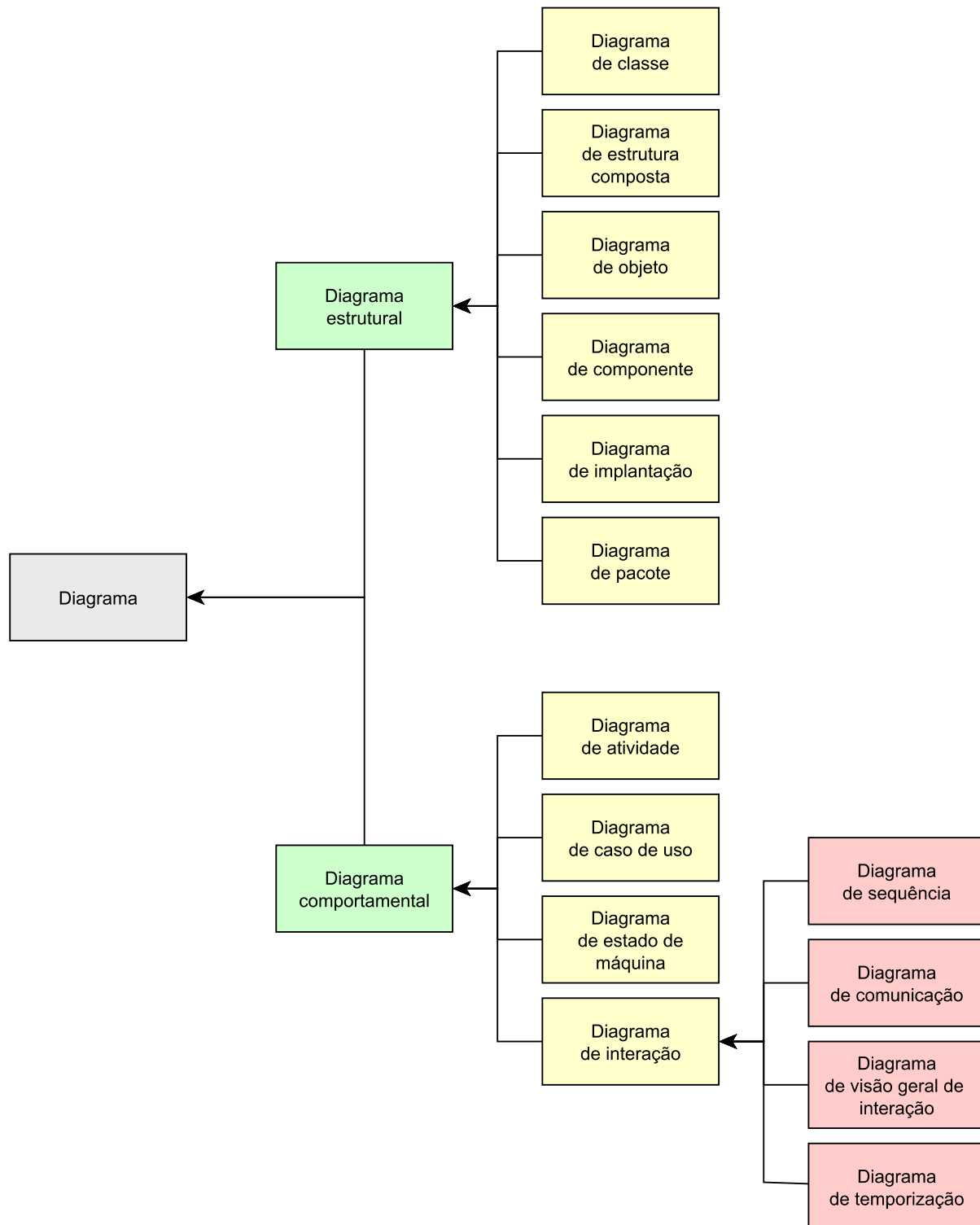


Tabela 1: Diagramas UML estruturais

Diagrama	Propósito
Classe	Representação de classes, recursos e relações.
Componente	Estruturação e conexão de componentes.
Estrutura composta	Decomposição de uma classe em tempo de execução.
Implantação	Implantação de artefatos em nós estruturais.
Objeto	Exemplificar configurações de instâncias.
Pacote	Estruturação hierárquica em tempo de compilação.

Fonte: FOWLER, 2004, p. 11 (Tradução nossa)

Tabela 2: Diagramas UML comportamentais

Diagrama	Propósito
Atividade	Representação de comportamento procedural e paralelo.
Caso de uso	Representação de como usuários interagem com o sistema.
Comunicação	Interação entre objetos, com ênfase na representação por conexões.
Estado de máquina	Representação de como eventos modificam objetos ao decorrer do tempo.
Sequência	Interação entre objetos, com ênfase na representação sequencial.
Temporização	Interação entre objetos, com ênfase na representação por tempo.
Visão geral de interação	Representação mista de diagrama de sequência e de atividade.

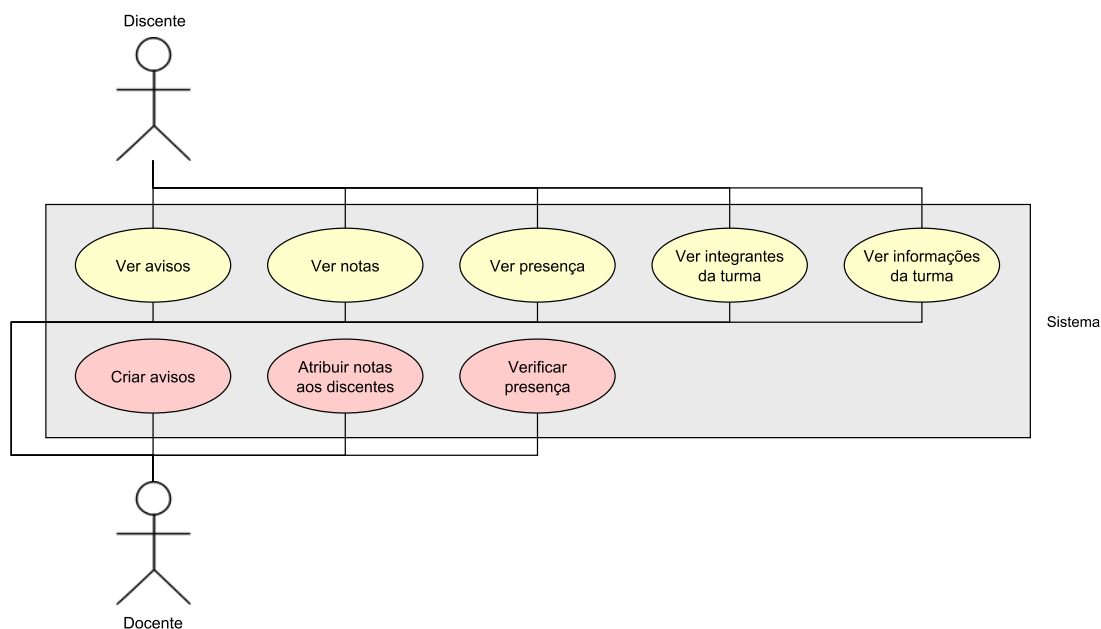
Fonte: FOWLER, 2004, p. 11 (Tradução nossa)

2.5.3 Diagramas utilizados

2.5.3.1 Diagrama de casos de uso

O diagrama de casos de uso (figura 27) é uma representação gráfica das relações em que cada caso de uso possui em relação a si mesmos e aos seus atores, que são usuários que interagem com o sistema, logo determinando qual conjunto de funcionalidades cada utilizador irá usufruir (FOWLER, 2004, p. 102–103). Além disso, pode haver uma relação de dependência entre cada caso de uso, podendo ser do tipo <<extend>>, que é uma dependência opcional de um caso executada dependendo das condições existentes durante o uso, ou <<include>>, que é uma dependência obrigatória de um caso de uso que e deve ser executada para qualquer situação na qual o dependente de tal módulo esteja em execução (PILONE; PITMAN, 2005). Logo, tal diagrama é uma ferramenta auxiliar na documentação textual de casos de uso, que é importante para a definição dos requisitos funcionais (FOWLER, 2004, p. 99).

Figura 27: Diagrama de casos de uso para um sistema de turmas



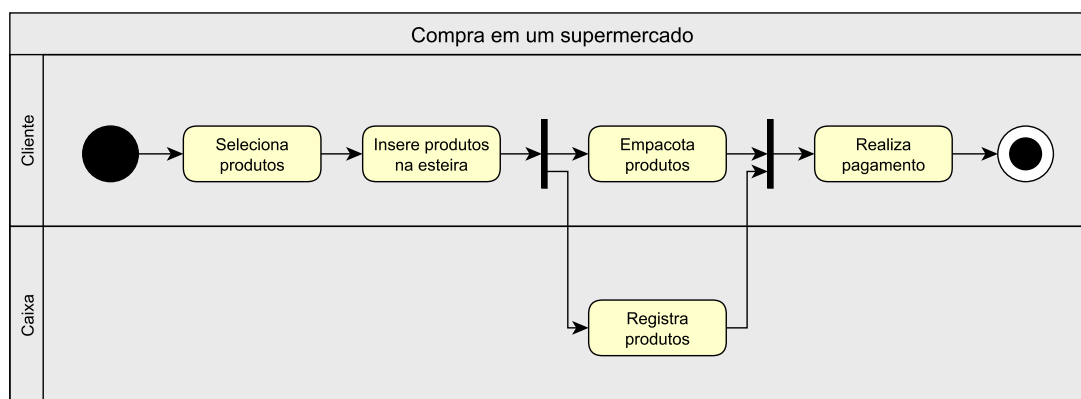
Fonte: Elaborado pelo autor

Logo, o diagrama representado na figura 27 expõe as funcionalidades que um sistema de gestão de turmas possui e quais tipos de usuários possuem acesso ao mesmo. Neste caso, o discente possui acesso a todas as funções, com exceção daquelas que necessitam de um privilégio maior de uso, como a atribuição de nota a cada aluno, no qual tal acesso é permitido apenas ao docente.

2.5.3.2 Diagrama de atividades

O diagrama de atividades (figura 28) possui a finalidade de representar processos de uma operação de maneira similar a um fluxograma, porém, este diagrama possui outros recursos que permitem uma representação mais avançada, como paralelismo, na qual uma notação permite a divisão de processos após uma determinada atividade e tornando-os concorrentes, e particionamento, sendo possível associar cada atividade às unidades envolvidas no processo de uma operação, como usuário e sistema, ou comprador e vendedor, por exemplo (FOWLER, 2004, p. 117–122). Logo, este tipo de representação permite que sejam feitos diagramas de jornada do usuário durante a utilização do sistema mais detalhados em comparação aos fluxogramas por permitirem explicitar o comportamento do sistema dependendo do *input* do usuário e como esta aplicação responde a tal evento.

Figura 28: Diagrama de atividades com particionamento e paralelismo



Fonte: Elaborado pelo autor

Então, o diagrama exposto na figura 28 apresenta uma situação muito comum no cotidiano, que é a ida ao supermercado. Este evento se inicia a partir do momento que o usuário seleciona os produtos desejados, o que leva, ao final deste primeiro processo, ao cliente inserir tais produtos na esteira. Neste momento, após a inserção dos produtos, existe uma condição de paralelismo, na qual o caixa registra os produtos da esteira e o cliente realiza o empacotamento dos mesmos. Após isso, o usuário realiza o pagamento, assim encerrando a atividade de compra no supermercado.

2.6 Diagrama de Entidade e Relacionamento

O diagrama de entidade e relacionamento é uma ferramenta utilizada no processo de Engenharia de Software que busca representar graficamente a estrutura e a relação de dados de uma aplicação (BAGUI; EARP, 2023, p. 72). Tal ferramenta possui variações de representação, sendo a mais utilizada o modelo criado por Richard Baker em 1990 e adaptado posteriormente pela Oracle, logo denominada de notação Baker/Oracle (BAGUI; EARP, 2023, p. 325).

O elemento mais importante do diagrama é chamado de entidade, que descreve uma estrutura de dados que identifique alguma funcionalidade ou autor, como um usuário (BAGUI; EARP, 2023, p. 73), possuindo elementos denominados de atributos, que são propriedades, como nome ou telefone, que serão associados a este autor (BAGUI; EARP, 2023, p. 73 e p. 326).

A outra parte essencial deste diagrama é inerente ao relacionamento entre as entidades, que define a dependência e a quantidade na qual uma entidade pode ser relacionada a outra, sendo tal chamada de cardinalidade (BAGUI; EARP, 2023, p. 129). A cardinalidade entre entidades pode ser definida de acordo com a tabela abaixo:

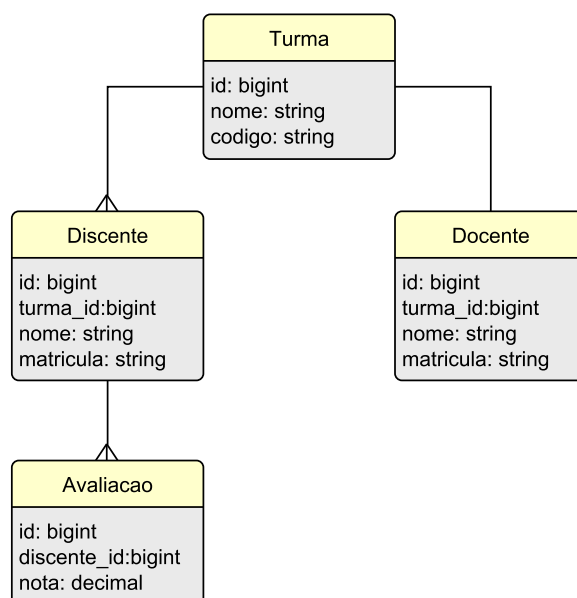
Tabela 3: Cardinalidade no diagrama de entidade e relacionamento

Cardinalidade	Nome	Descrição
1:1	Um para um	Uma entidade pai é relacionada a uma entidade filha.
M:1	Muitos para um	Várias entidades pais são relacionadas a uma entidade filha.
1:M	Um para muitos	Uma entidade pai é relacionada a muitas entidades filhas.
M:N	Muitos para muitos	Muitas entidades pais são relacionadas a muitas entidades filhas.

Fonte: BAGUI, 2023, p. 130-132

Abaixo, é mostrado um exemplo de representação simplificada para uma turma (figura 29), possuindo um docente e vários discentes, nos quais os últimos possuem várias avaliações. Cada entidade listada possui os atributos apropriados para cada uma, na qual uma turma, por exemplo, possui o nome da disciplina e o código da mesma, enquanto discentes e docentes possuem atributos associados às suas matrículas e seus nomes. Além disso, há um atributo que representa a nota de cada avaliação pertencente a um docente.

Figura 29: Diagrama de entidade e relacionamento para uma turma



Fonte: Elaborado pelo autor

Logo, os relacionamentos entre entidades são classificadas conforme a tabela abaixo:

Tabela 4: Cardinalidade para a representação simplificada de um sistema de turma

Relacionamento	Cardinalidade
Turma – Docente	1:1
Turma – Discente	1:N
Discente – Avaliacao	1:N

Fonte: Elaborado pelo autor

3 METODOLOGIA

3.1 Pesquisa

3.1.1 *Propósito*

O propósito desta pesquisa é de natureza exploratória, por conta de buscar um maior entendimento acerca de um assunto menos difundido (KOTHARI, 1990, p. 2) como apresentado pela contextualização (ver seção 1.1), visto que o presente trabalho tem o objetivo de buscar saber se é possível sanar a problemática exposta por meio da ferramenta proposta via estudos e obtenção de conclusões acerca dos levantamentos obtidos com usuários inseridos nas áreas de Arte e Design através de validações acerca do funcionamento da ferramenta, da eficiência da interface gráfica no uso da referida plataforma e da confirmação da viabilidade de tal como modelo de negócio com o objetivo de validar a hipótese de que as atuais ferramentas de arte generativa por interface gráfica são muito limitadas e que seria possível realizar um modelo rentável caso houvesse uma plataforma web que gerasse uma grande variedade de formas e que, com isto, os criadores pudessem comercializar tais por meio da mesma.

3.1.2 *Abordagem*

As pesquisas aplicadas neste trabalho referem-se à usabilidade por meio da utilização de um questionário UEQ simplificado (ver seção 5.6), fornecendo dados quantitativos mediante perguntas com gradações e dados qualitativos por meio de perguntas opcionais de sugestões. Além disso, durante a implementação, são utilizadas as heurísticas de Nielsen (ver seção 2.2.4.1) como guia para a avaliação de similares e para a criação de uma boa interface de usuário para o projeto.

3.1.3 Procedimentos

Tabela 5: Procedimentos

Tipo	Abordagem da pesquisa	Descrição
Questionários	Quantitativa e qualitativa	Uso de questionários online para mensurar satisfação em relação à navegação e às utilidades oferecidas pela plataforma.
Avaliação heurística	Qualitativa	Produção de interface do projeto e avaliação de similares de acordo com as heurísticas de Nielsen (seção 2.2.4.1), de forma a interpretar falhas de usabilidade das alternativas encontradas para que o referido projeto possua uma boa interface de usuário.

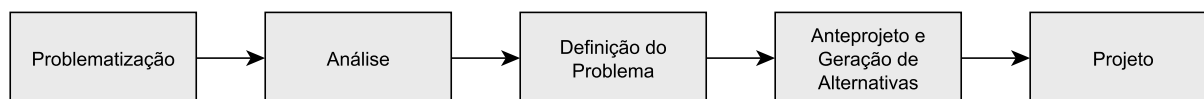
Fonte: Elaborado pelo autor

3.2 Projeto

3.2.1 Design

No âmbito do Design, é utilizada a metodologia do Gui Bonsiepe (BONSIEPE; KELLNER; POESSNECKER, 1984) pelo fato do desenvolvimento do projeto ser melhor acomodado em uma organização temporal linear (ver figura 30). Logo, este projeto busca, primeiramente, inicializar com uma problemática e o contexto envolvido (Problematização), para então ser feito um levantamento acerca de produtos e serviços com certa similaridade à temática trabalhada (Análise) que, em seguida, tais dados fornecem embasamento para a criação de uma lista de requisitos funcionais e não funcionais do projeto, além de suas restrições (Definição do Problema), sendo então possível trabalhar em *sketches* iniciais para melhor entender como a interface da plataforma pode ser melhor apresentada ao usuário (Anteprojeto e Geração de Alternativas), possibilitando, após esta etapa, a implementação do sistema em si (Projeto).

Figura 30: Diagrama da metodologia de Design de Gui Bonsiepe

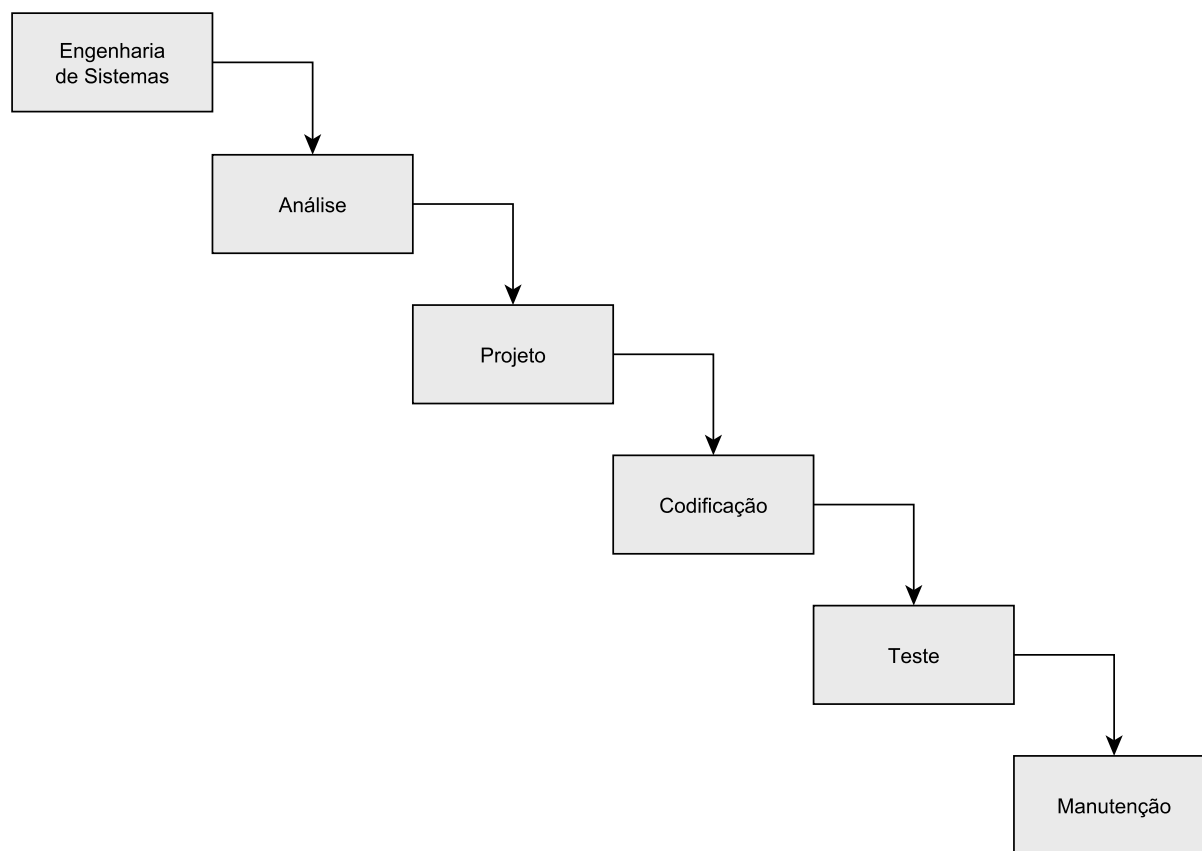


Fonte: BONSIEPE, 1984, p. 35

3.2.2 Computação

Em relação à Engenharia de Software, o desenvolvimento do sistema busca seguir o modelo em Cascata (ver figura 31), visto que este trabalho seguirá uma execução sequencial, se assemelhando em alguns pontos à metodologia do Gui Bonsiepe, e necessitando que as entregas em cada etapa sejam consistentes (HIRAMA, 2011, p. 25–28), com a primeira etapa destinada à concepção das necessidades, em uma visão macro, para o funcionamento do negócio (Engenharia de Sistemas) e, logo após, a definição dos requisitos, com maiores especificidades técnicas, que serão essenciais para o funcionamento da plataforma (Análise), as quais as duas primeiras etapas fornecem documentação suficiente para a estruturação do sistema (Projeto) e implementação da plataforma (Codificação). Logo após a conclusão de um MVP, com todas as funcionalidades planejadas implementadas, serão feitas, conforme descrito na seção 3.1, pesquisas com os usuários acerca das opiniões sobre as funcionalidades e a interface oferecida pelo sistema (Teste), de forma com que os *feedbacks* dos usuários sirvam para uma futura versão da plataforma com a implementação das melhorias sugeridas pelos mesmos (Manutenção).

Figura 31: Diagrama da metodologia de desenvolvimento em Cascata



Fonte: HIRAMA, 2011, p. 26

4 ANÁLISE DE SIMILARES

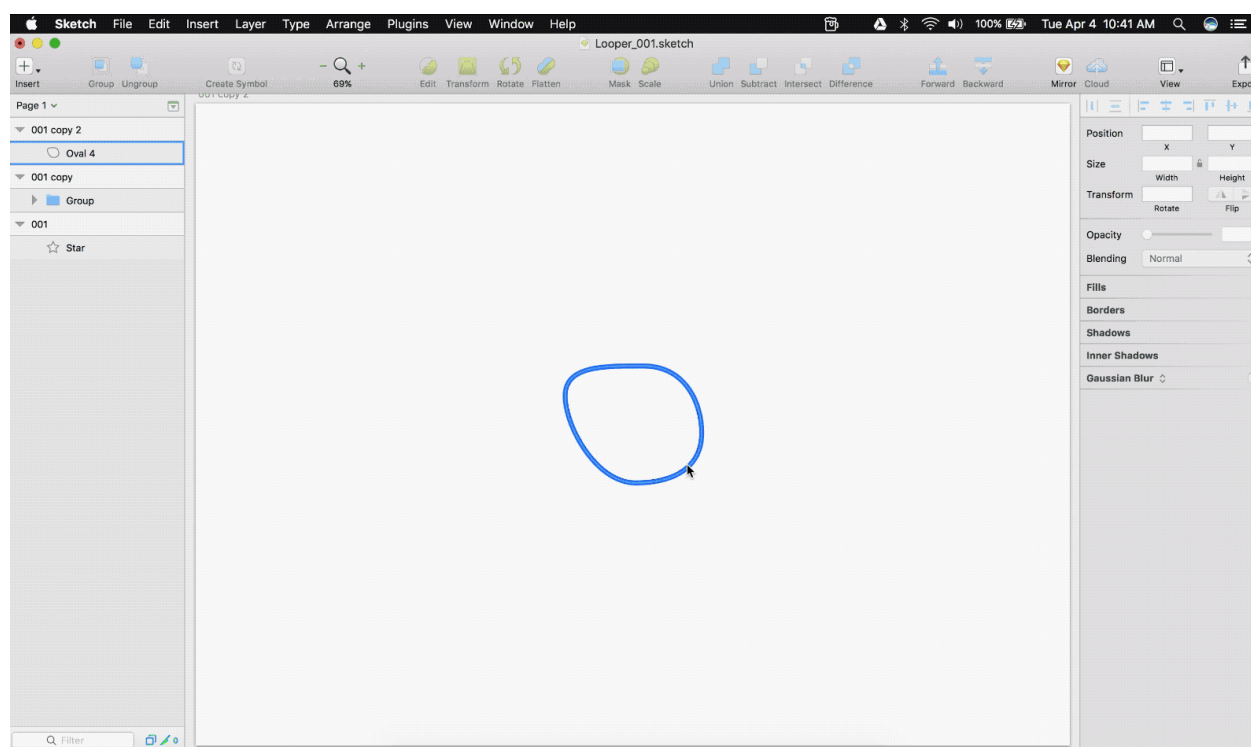
4.1 Geradores

4.1.1 Por interface de usuário

4.1.1.1 Looper

Looper³ é um *plugin* para o *software* Sketch⁴, disponível apenas para macOS, no qual o mesmo gera repetições a partir de uma figura criada pelo próprio usuário (figura 32), mediante customização de parâmetros em uma interface gráfica (figura 33), que possuem a utilidade de variar a rotação, escala e opacidade durante cada laço de repetição. Logo, é possível ter resultados que têm a estética de arte generativa (figura 34), porém há a limitação oriunda do *software* não gerar automaticamente tais formas, dependendo que o usuário realize tal processo.

Figura 32: Looper — Criação da figura

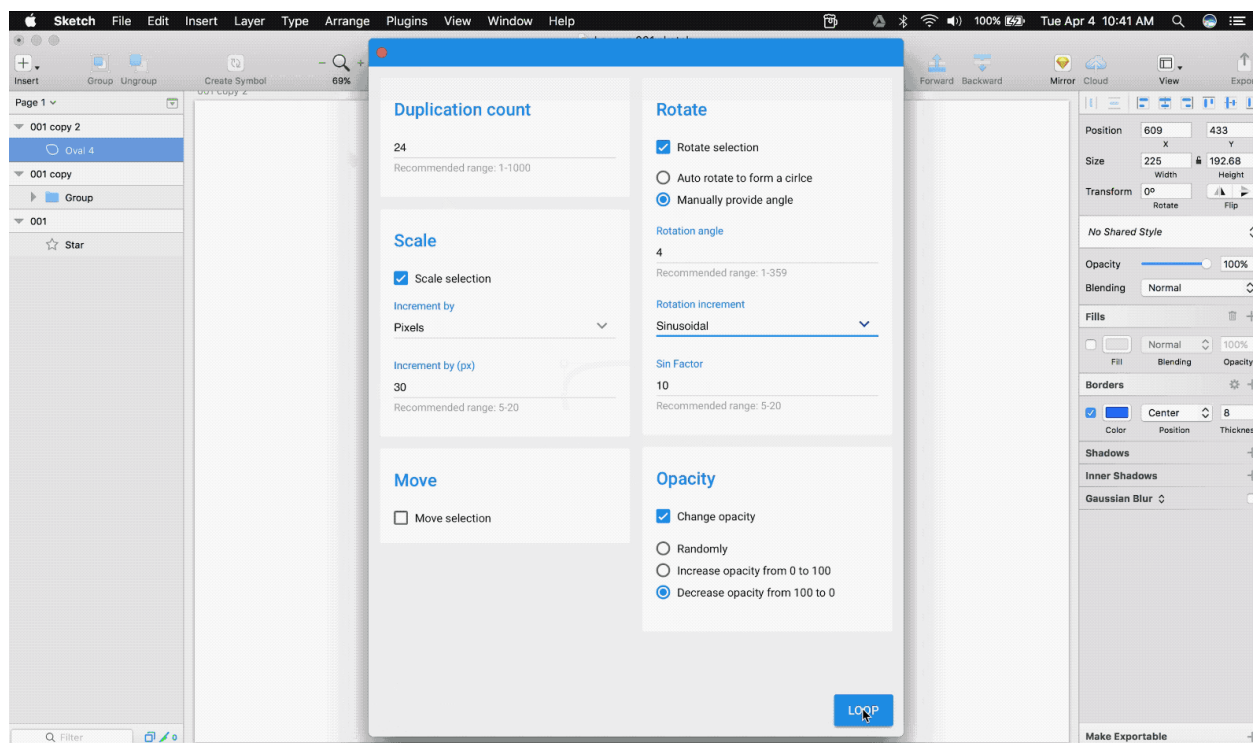


Fonte: Imagem retirada de <https://github.com/sureskumar/Looper> em 21 de Abril de 2023

³<https://github.com/sureskumar/Looper>

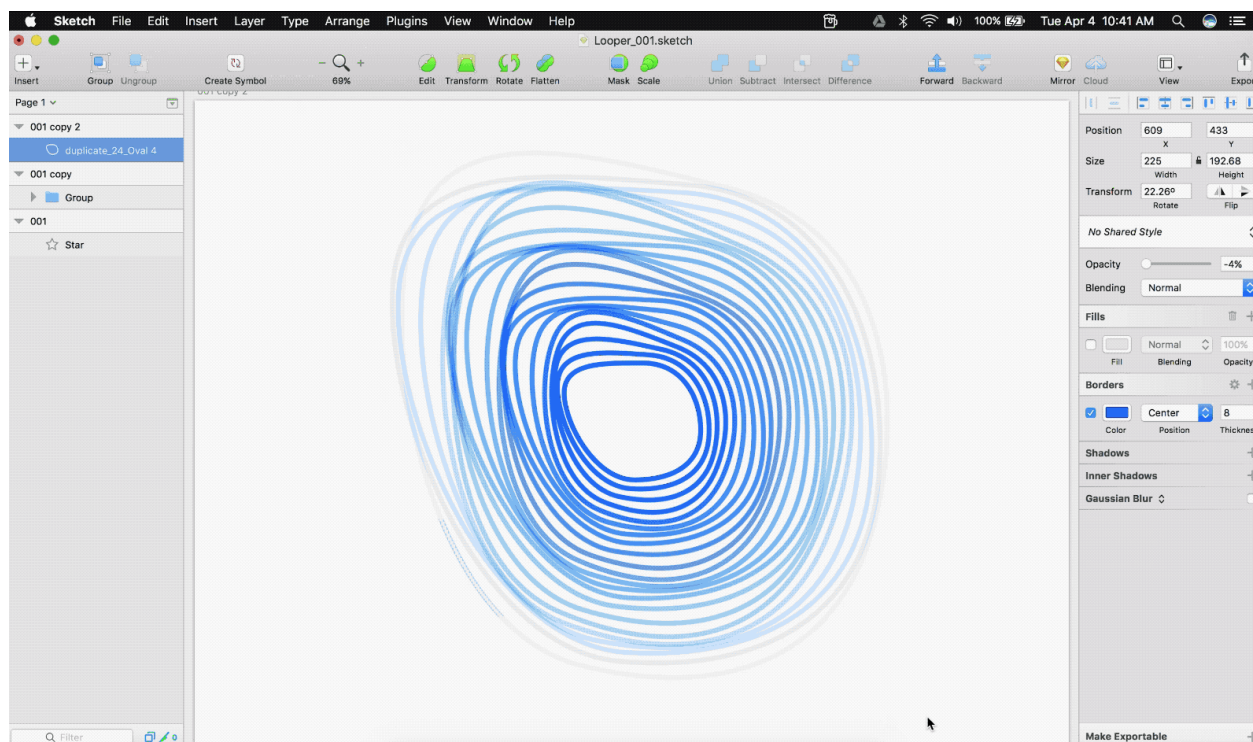
⁴<https://www.sketch.com/>

Figura 33: Looper — Seleção de parâmetros



Fonte: Imagem retirada de <https://github.com/sureskumar/Looper> em 21 de Abril de 2023

Figura 34: Looper — Resultado final

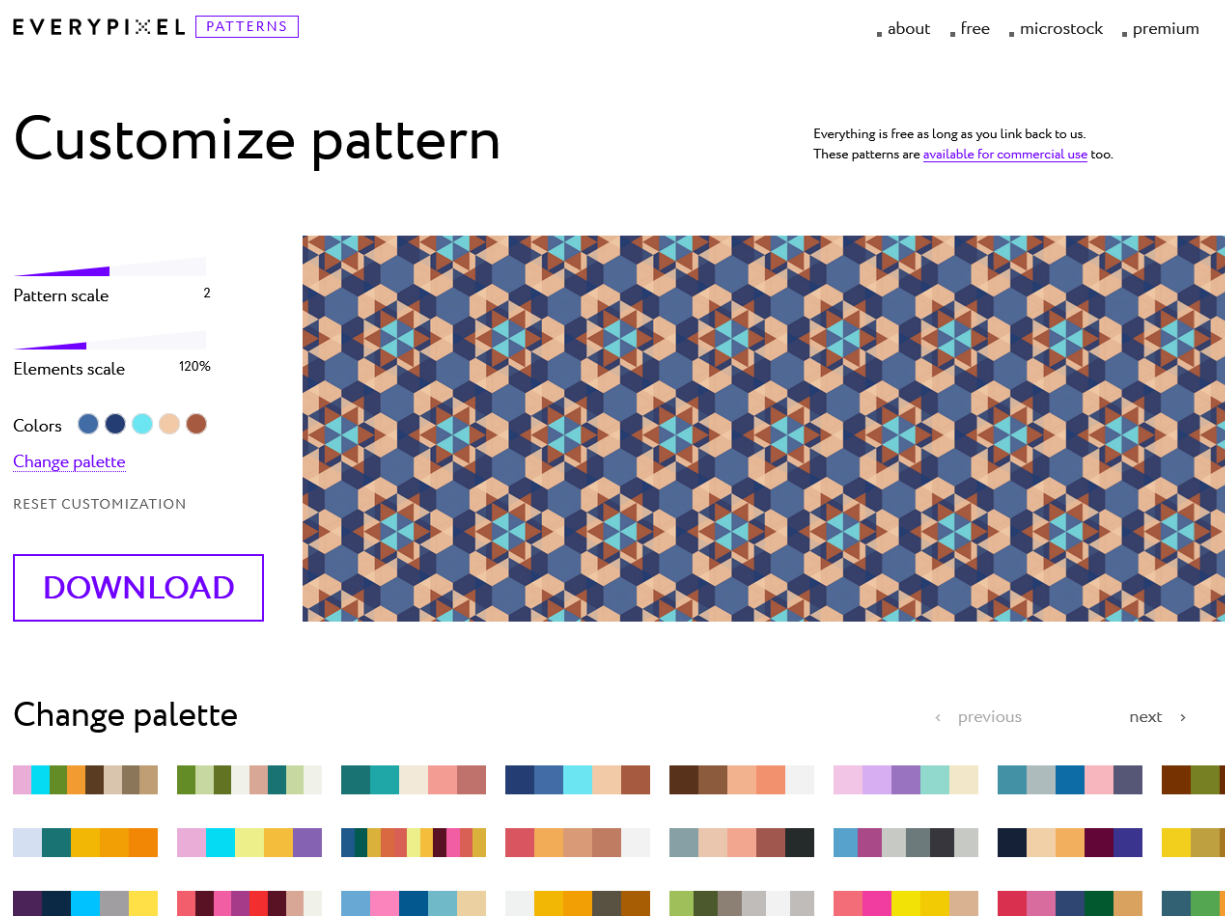


Fonte: Imagem retirada de <https://github.com/sureskumar/Looper> em 21 de Abril de 2023

4.1.1.2 Everypixel Patterns

O serviço Everypixel Patterns⁵ disponibiliza vários padrões predefinidos para o usuário customizá-los (figura 35), sendo possível configurar parâmetros como escala do padrão, dos elementos, assim como a paleta de cores, possibilitando, no final do processo, que o utilizador faça o *download* do conjunto de figuras nos formatos .svg ou .png. Portanto, mesmo que haja uma certa automatização em relação à customização do padrão escolhido, não apresenta resultados similares à arte generativa.

Figura 35: Everypixel Patterns — Customização do padrão selecionado



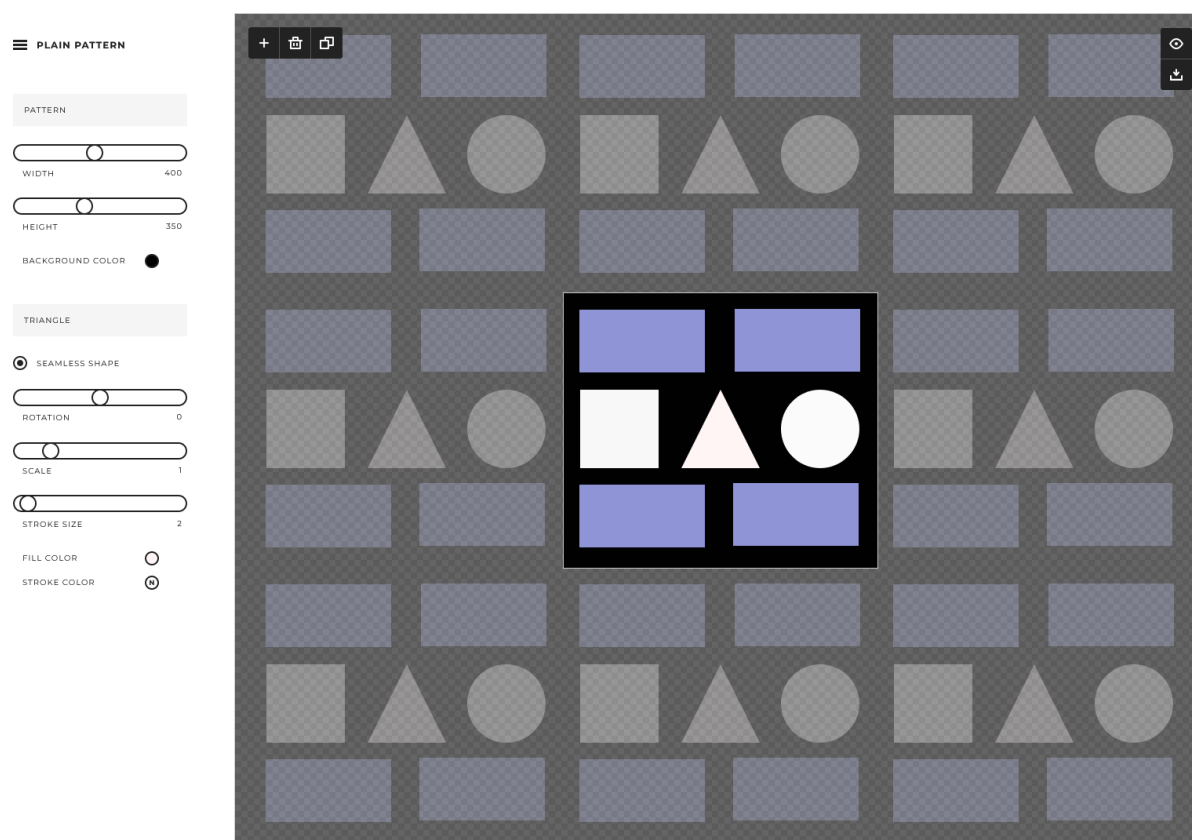
Fonte: Imagem retirada de <https://www.everypixel.com/patterns/customize> em 21 de Abril de 2023

⁵<https://www.everypixel.com/patterns>

4.1.1.3 Plain Pattern

A aplicação Plain Pattern⁶, a qual ainda está em estágio de desenvolvimento, permite que o usuário selecione algumas formas geométricas básicas, além de alguns ícones provenientes do Noun Project⁷ ou, até, mesmo, realizar o *upload* de um arquivo .svg e realizar a edição do padrão pela interface (figura 36).

Figura 36: Plain Pattern — Criação de padrão



Fonte: Imagem retirada de <http://www.kennethcachia.com/plain-pattern/app/> em 21 de Abril de 2023

4.1.1.4 Korpus

A incubadora russa Korpus⁸, focada no desenvolvimento de projetos com Inteligência Artificial (KORPUS, 2021), desenvolveu uma identidade visual⁹ baseada em autômatos celulares com a criação de uma aplicação¹⁰ que gera padrões com o conceito citado tanto aleatoriamente quanto por meio de um texto, o qual pode ser definido em sua interface (figuras 37, 38). Logo, tal alternativa possui, mesmo que focada em apenas um tipo, geração de arte por meio de interface de usuário, embora a mesma possua limitações.

⁶<http://www.kennethcachia.com/plain-pattern/app/>

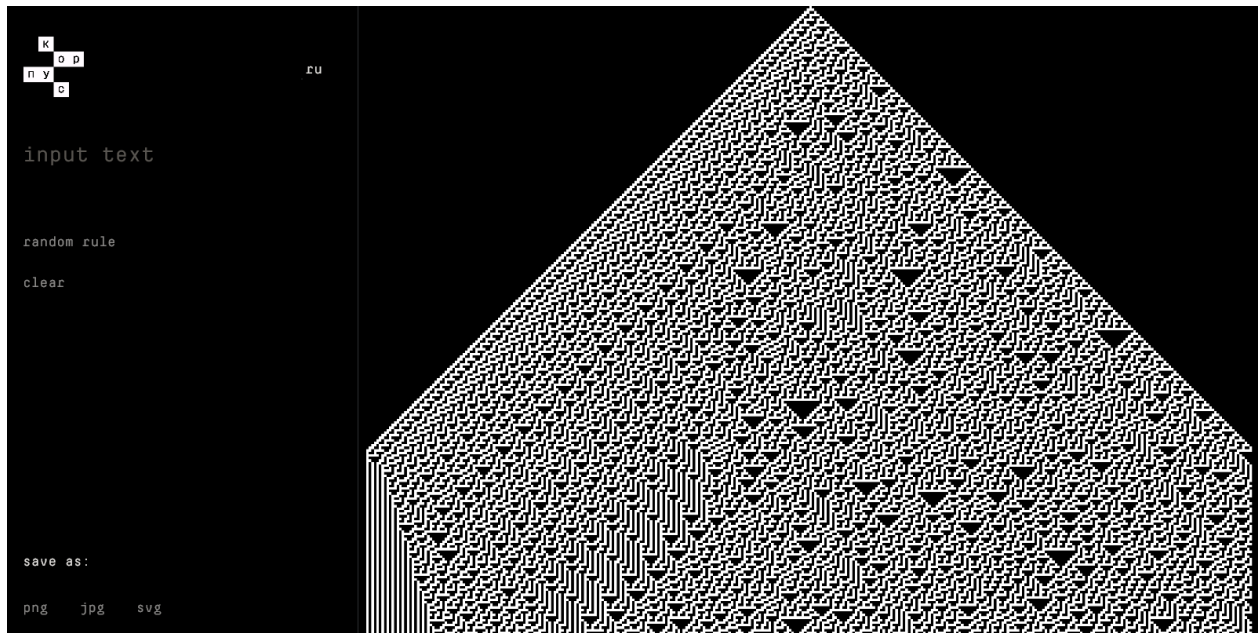
⁷<https://thenounproject.com/>

⁸<https://korpus.io/>

⁹<https://www.behance.net/gallery/64306153/Corpus>

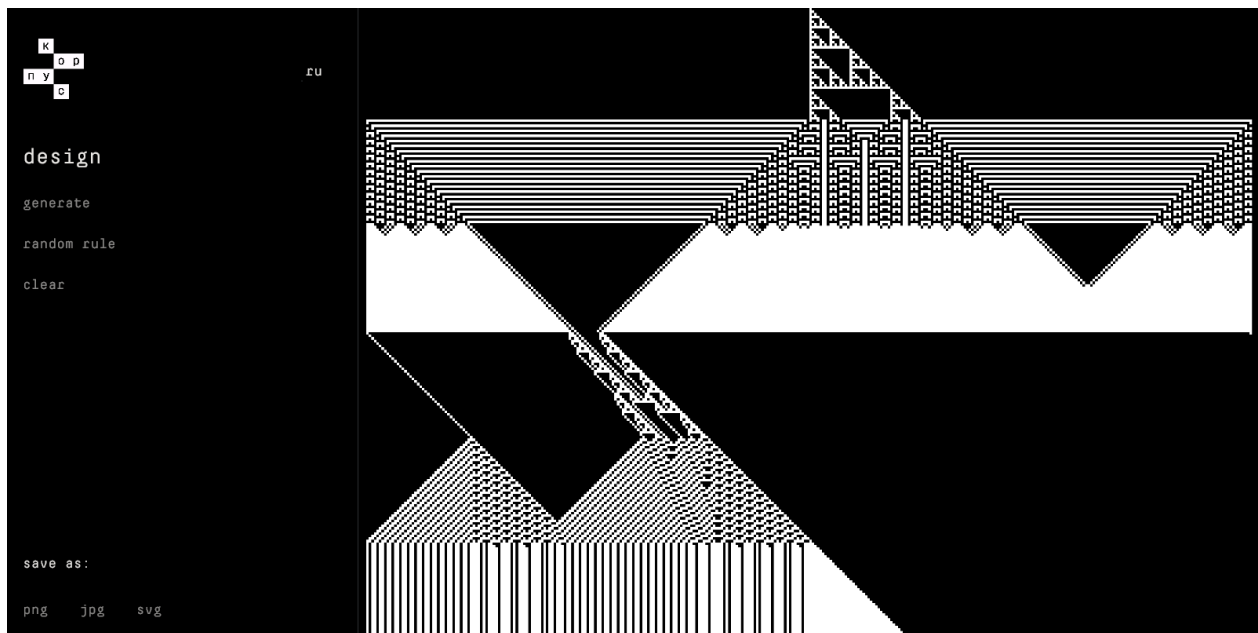
¹⁰<https://korpus.io/en/app/>

Figura 37: Korpus — Padrão aleatório



Fonte: Imagem retirada de <https://korpus.io/en/app/> em 21 de Abril de 2023

Figura 38: Korpus — Geração de autômato celular com a palavra “Design”

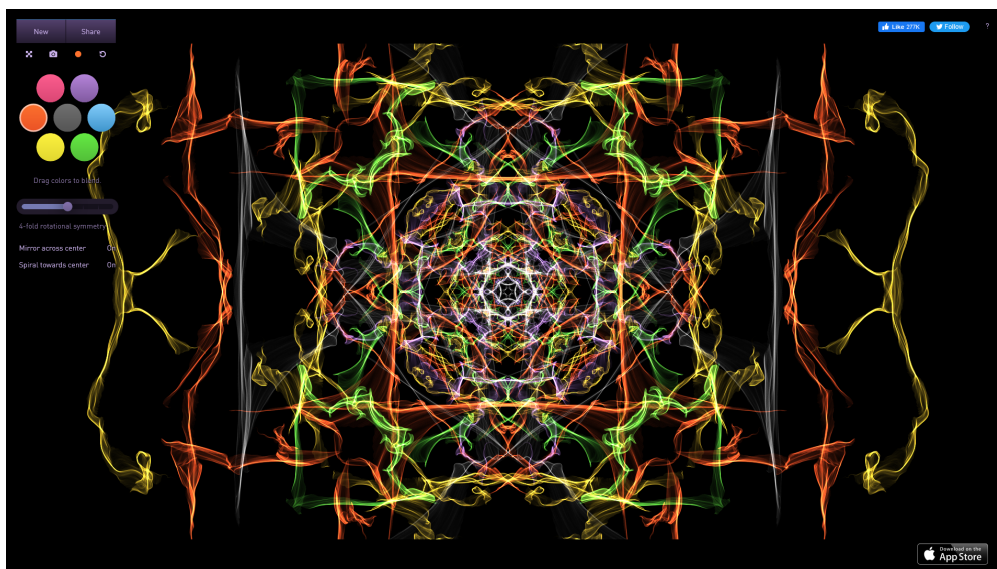


Fonte: Imagem retirada de <https://korpus.io/en/app/> em 21 de Abril de 2023

4.1.1.5 Silk

Silk¹¹ é uma aplicação para web e iOS que permite ao usuário desenhar qualquer tipo de forma desejada por meio de uma tela interativa (figura 39), fazendo com que o *software* gere o padrão programado à medida que o utilizador desenha na aplicação, de forma a produzir formas variadas com o uso da técnica adotada pelo programa. Além disso, o Silk oferece algumas opções de simetria e de cores que buscam aprimorar a experiência e o resultado do desenho do usuário, permitindo que o mesmo possua mais possibilidades de padrões no ato de desenhar. Além disso, a aplicação permite ao usuário salvar e compartilhar a imagem diretamente pela interface, o que facilita a utilização pelo público.

Figura 39: Silk — Imagem realizada por usuário



Fonte: Imagem retirada de <http://weavesilk.com/> em 03 de Maio de 2023

4.1.1.6 GPU Fluid

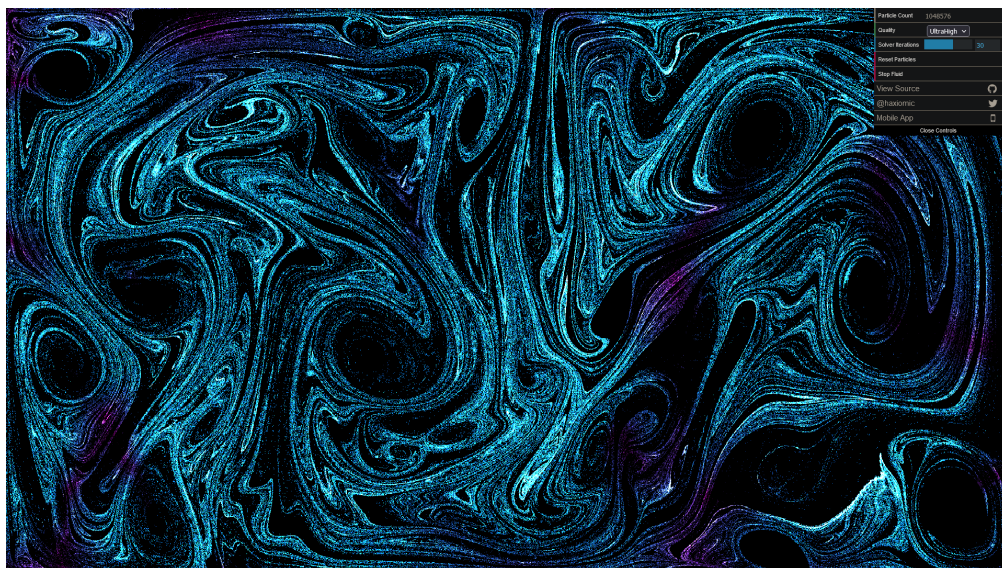
A aplicação GPU Fluid¹² é um experimento interativo para a web em código aberto¹³ realizado pelo desenvolvedor George Corney que permite ao usuário, de acordo com o movimento do mouse clicado, gerar padrões de fluidos na tela, sendo possível também ter acesso a opções de qualidade de renderização, como o número de partículas exibidas e as iterações geradas pelo *input* do usuário, para aprimorar o resultado gerado. No entanto, o programa não oferece, diretamente pela interface, opções para que o usuário salve ou compartilhe a imagem gerada.

¹¹<http://weavesilk.com/>

¹²<https://haxiomic.github.io/GPU-Fluid-Experiments/html5/>

¹³<https://github.com/haxiomic/GPU-Fluid-Experiments>

Figura 40: GPU Fluid — Imagem realizada por usuário

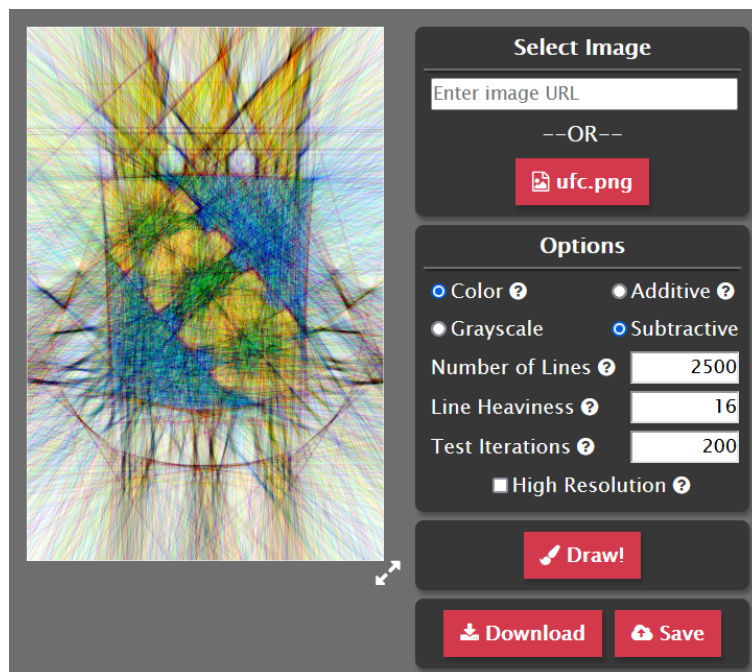


Fonte: Imagem retirada de <https://haxiomic.github.io/GPU-Fluid-Experiments/html5/> em 03 de Maio de 2023

4.1.1.7 Linify

O serviço Linify¹⁴ converte imagens fornecidas pelo usuário por um estilo que utiliza linhas automaticamente por meio de um algoritmo. Na interface, é possível selecionar um link para a imagem ou realizar o upload, além de ser possível alterar alguns parâmetros que podem influenciar diretamente na qualidade ou no estilo da imagem, como o número de linhas e de iterações a serem realizadas (figura 41).

Figura 41: Linify — Geração de padrão utilizando logotipo da Universidade Federal do Ceará



¹⁴<http://linify.me/>

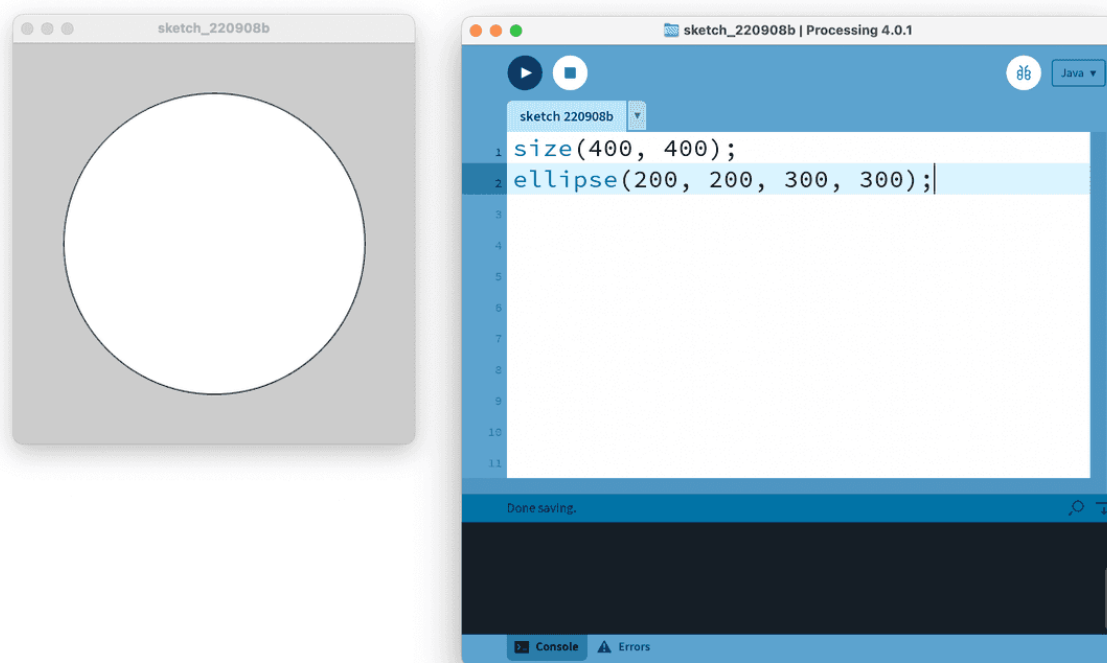
Fonte: Imagem gerada em <http://linify.me/> em 06 de Maio de 2023

4.1.2 Por algoritmo

4.1.2.1 Processing

Processing¹⁵ (figura 42) é uma ferramenta em código aberto baseada na linguagem Java com desenvolvimento iniciado em 2001 por Casey Reas e Benjamin Fry, então estudantes do MIT, para a criação de arte generativa com complexidade por sua ampla gama de funcionalidades, mas sendo de aprendizagem simplificada para o usuário por conta de sua sintaxe e abstrações (PEARSON, 2011, p. 14). A criação do Processing possui inspiração direta no *software Design By Numbers*¹⁶ (figura 43), criado em 1999 por John Maeda, o qual, como destaque, possuía um fácil aprendizado por conta de sua simples sintaxe, porém, o mesmo possuía limitações técnicas, não sendo possível gerar artes muito complexas por trabalhar apenas em escala de cinza e não ter um *canvas* muito extenso (REAS; FRY, 2018).

Figura 42: Processing — Interface do programa e renderização de elipse

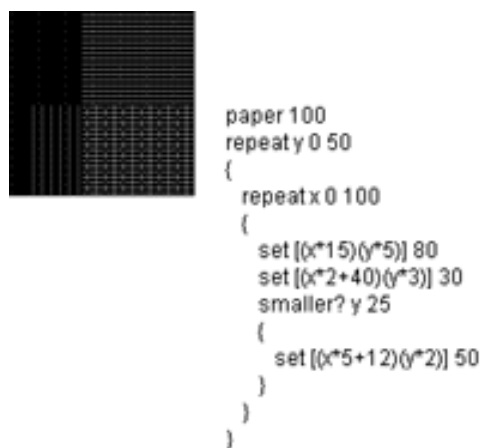


Fonte: Imagem retirada de <https://processing.org/tutorials/gettingstarted> em 06 de Maio de 2023

¹⁵<https://processing.org/>

¹⁶<https://dbn.media.mit.edu/>

Figura 43: Design By Numbers — Exemplo de código e sua renderização



Fonte: Imagem retirada de <https://dbn.media.mit.edu/whatisdbn.html> em 06 de Maio de 2023

O Processing, de acordo com a sua documentação de referência¹⁷, conta com várias funções, sendo possível, além da abstração de fórmulas matemáticas para elementos geométricos e a renderização dos mesmos, utilizar vários *inputs* que auxiliam na visualização das formas desejadas, como mouse ou teclado. No entanto, também são disponibilizadas bibliotecas que buscam ampliar a funcionalidade da ferramenta, permitindo, além disso, a utilização de *inputs* de áudio e vídeo, criando novas possibilidades para o uso desta tecnologia.

Desta forma, ao decorrer do tempo, o Processing, por suas características, se consolidou como uma das principais ferramentas na comunidade de Arte Generativa, ampliando a tecnologia para outras plataformas além do Java, como o Javascript¹⁸ (Web), Android¹⁹ e Python²⁰, também criando uma diversa comunidade pelo mundo, na qual a Processing Foundation²¹ auxilia na promoção de eventos e de treinamentos com a ferramenta.

4.1.2.2 Pts.js

Pts.js²² é uma biblioteca em Javascript que possui o mesmo propósito do p5.js em relação à integração de elementos visuais generativos na web. No entanto, em comparação à biblioteca fornecida pela Processing Foundation, o Pts.js possui a vantagem de ser bastante leve, possuindo aproximadamente 100KB²³ em relação aos 3.8MB²⁴ do p5.js, o que é bastante benéfico para diminuir o tempo de carregamento e o uso de recursos ao carregar uma página que busca aplicar o conceito de arte generativa. Logo, como o Pts.js possui menos módulos de funcionalidades em comparação à biblioteca Javascript do Processing, é prudente ponderar, de maneira preemptiva, quais funcionalidades a aplicação web necessitará para estudar a viabilidade de se utilizar uma biblioteca com menos recursos, mas que pode beneficiar a performance da referida aplicação.

¹⁷<https://processing.org/reference>

¹⁸p5.js — <https://p5js.org/>

¹⁹<https://android.processing.org/>

²⁰<https://py.processing.org/>

²¹<https://processingfoundation.org/>

²²<https://ptsjs.org/>

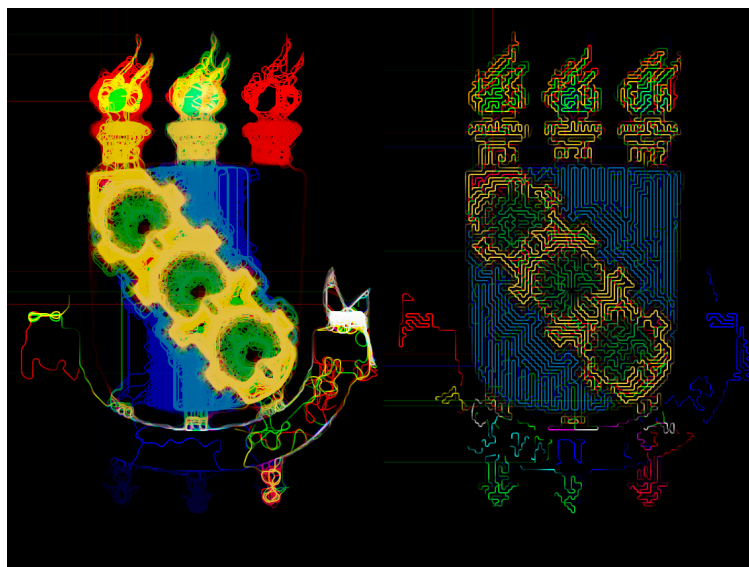
²³<https://github.com/williamngan/pts>

²⁴<https://github.com/processing/p5.js/releases/tag/v1.6.0>

4.1.2.3 Chromata

Chromata²⁵ (figura 44) é uma biblioteca Javascript que disponibiliza, por meio da inserção de um arquivo de imagem como *input*, efeitos de arte generativa com animação, sendo possível alterar, no código da página no qual a ferramenta é integrada, parâmetros disponibilizados para modificar o comportamento do gerador, como ângulo de curva, número de linhas e velocidade de geração, dispensando a criação da lógica da geração da imagem, sendo tal abstraída na própria biblioteca, facilitando o manuseio desta. Em contrapartida, este tipo de abordagem limita as possibilidades disponíveis em comparação a uma ferramenta como o Processing (ver seção 4.1.2.1), que permite com que o usuário crie todo o algoritmo de arte generativa.

Figura 44: Chromata — Logotipo da Universidade Federal do Ceará com duas variações de geração



Fonte: Imagens geradas em <https://www.michaelbromley.co.uk/experiments/chromata/> em 06 de Maio de 2023

4.1.2.4 Nannou

Nannou²⁶ é uma biblioteca para a linguagem de programação Rust²⁷ que possui funcionalidades semelhantes ao Processing, no entanto, tal linguagem utilizada pela *framework* enfatiza a portabilidade entre a muitos sistemas operacionais, além da performance na execução do código, possuindo uma eficiência similar ao C e C++ (NANNOU, 2022). Além das funcionalidades de gráficos, a biblioteca conta com a programação de LEDs, instruções para máquinas de corte à laser, análise de áudio, além da construção de interface de usuário para os *sketches* realizados pelo autor.

4.1.2.5 OPENRNRDR

OPENRNRDR²⁸ é uma *framework* multiplataforma de computação criativa escrita em Kotlin focada na performance da exibição dos resultados desejados com o uso da biblioteca LWJGL²⁹, criada em Java e voltada à renderização de jogos e que pode utilizar APIs como o OpenGL ou o Vulkan (LWJGL, 2023). Um dos recursos que esta ferramenta possui como diferencial é o módulo de machine learning chamado ORML³⁰, permitindo experimentos criativos de computação gráfica utilizando inteligência

²⁵<https://www.michaelbromley.co.uk/experiments/chromata/>

²⁶<https://nannou.cc/>

²⁷<https://www.rust-lang.org/>

²⁸<https://openrndr.org/>

²⁹<https://www.lwjgl.org/>

³⁰<https://orml.openrndr.org/>

artificial.

4.1.2.6 C4iOS

C4iOS³¹ é uma ferramenta de computação criativa que utiliza a linguagem Swift e é voltada para a execução de experiências interativas de design computacional em dispositivos iOS. Embora a *framework* conte com recursos que são possíveis serem achados em outras ferramentas, tal biblioteca encontra-se com o seu desenvolvimento estagnado, sendo, conforme seu repositório no GitHub, a última versão lançada em 2017 (C4IOS, 2018).

4.1.2.7 Canvas Sketch

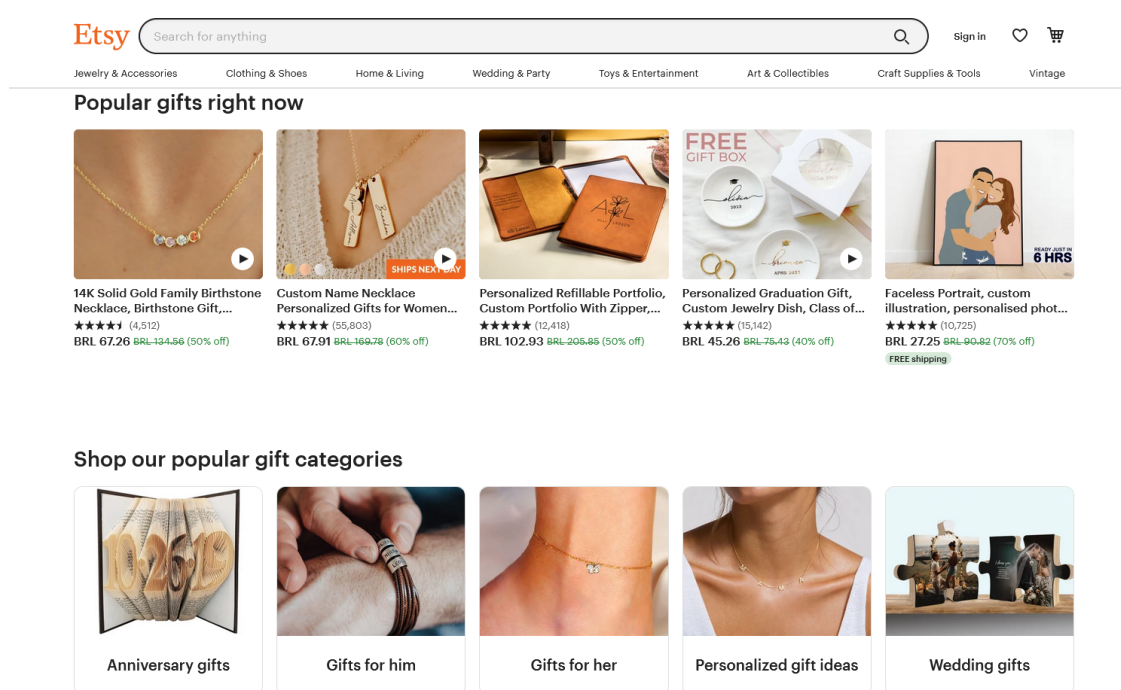
Canvas Sketch³² é uma biblioteca Javascript que se encontra em fase beta focada em arte generativa, porém não possuindo recursos mais avançados como o aceite de *inputs* audiovisuais. Embora haja este tipo de ausência, a referida ferramenta permite salvar as animações resultantes do código implementado em formatos de vídeo ou de imagem, além de permitir ser utilizada concomitantemente com outras *frameworks* Javascript de computação gráfica, como o p5.js, a biblioteca web do Processing (ver seção 4.1.2.1).

4.2 Modelos de negócio

4.2.1 Serviços

4.2.1.1 Etsy

Figura 45: Página da Etsy



Fonte: Imagem retirada de <https://www.etsy.com/> em 16 de Junho de 2023.

³¹<https://c4ios.com>

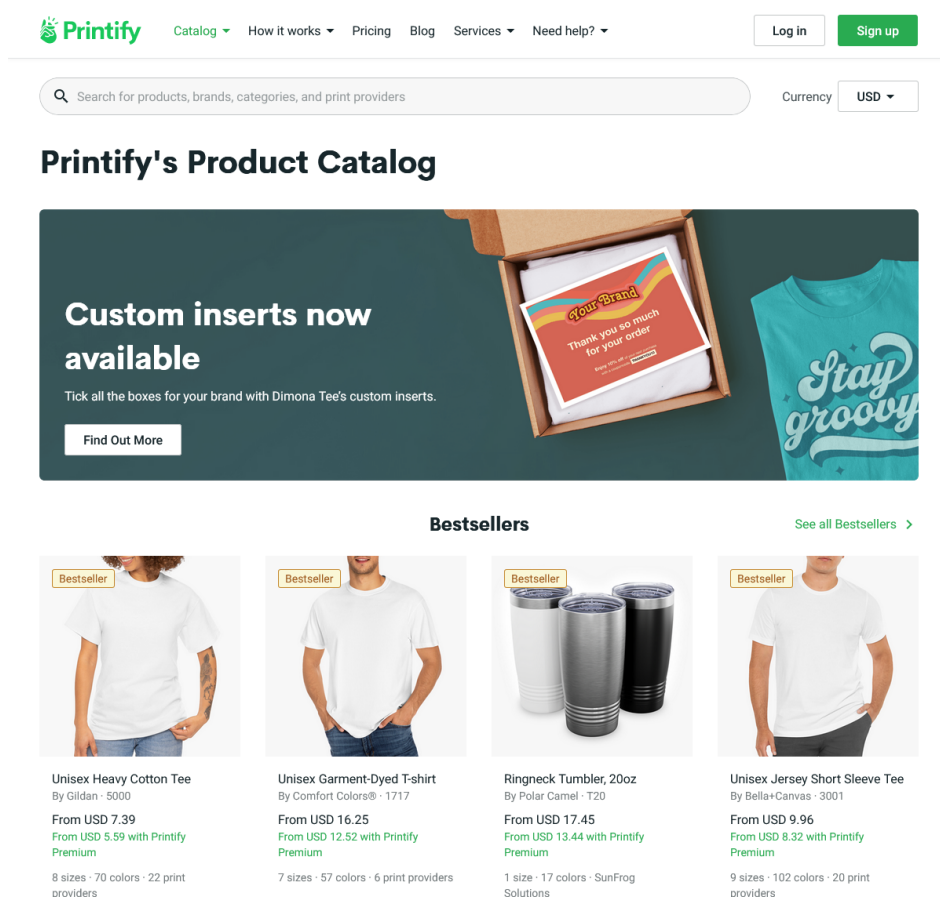
³²<https://github.com/mattdesl/canvas-sketch>

Etsy³³ (figura 45) é um *marketplace* no qual seus usuários podem vender uma vasta gama de artigos, como joias, roupas, artigos de decoração, pinturas, esculturas, brinquedos e ferramentas. Por cada postagem de produto efetivada, o site cobra 20 cents de dólar americano, além de uma taxa de 6,5% sobre o valor do produto com base em cada venda realizada (ETSY, 2023).

Porém, mesmo que o site possua moderação e imponha regras nos quais os produtos devem ser de procedência artesanal ou, então, antiguidades, a plataforma atualmente sofre críticas por possuir postagens de produtos nos quais seus usuários reivindicam que são peças originais ou antigas, mas que, na verdade, são artigos provenientes de grandes redes de varejo, como a Amazon, Alibaba, ASDA e B&M, sendo vendidos na Etsy por um preço muito acima comparado às lojas nas quais tais produtos são originários (WOOD, 2023).

4.2.1.2 Printify

Figura 46: Página da Printify



Fonte: Imagem retirada de <https://www.printify.com/app/products> em 16 de Junho de 2023.

Printify³⁴ (figura 46) é um serviço de impressão sob demanda de estampas em artigos como roupas, copos e bolsas com a utilização de artes produzidas por usuários registrados na plataforma, os quais possuem a possibilidade de assinar um dos três planos oferecidos: gratuito, possuindo limite de criação de 5 lojas; *premium*, com preço de 24,99 dólares ao mês e possuindo

³³<https://www.etsy.com/>

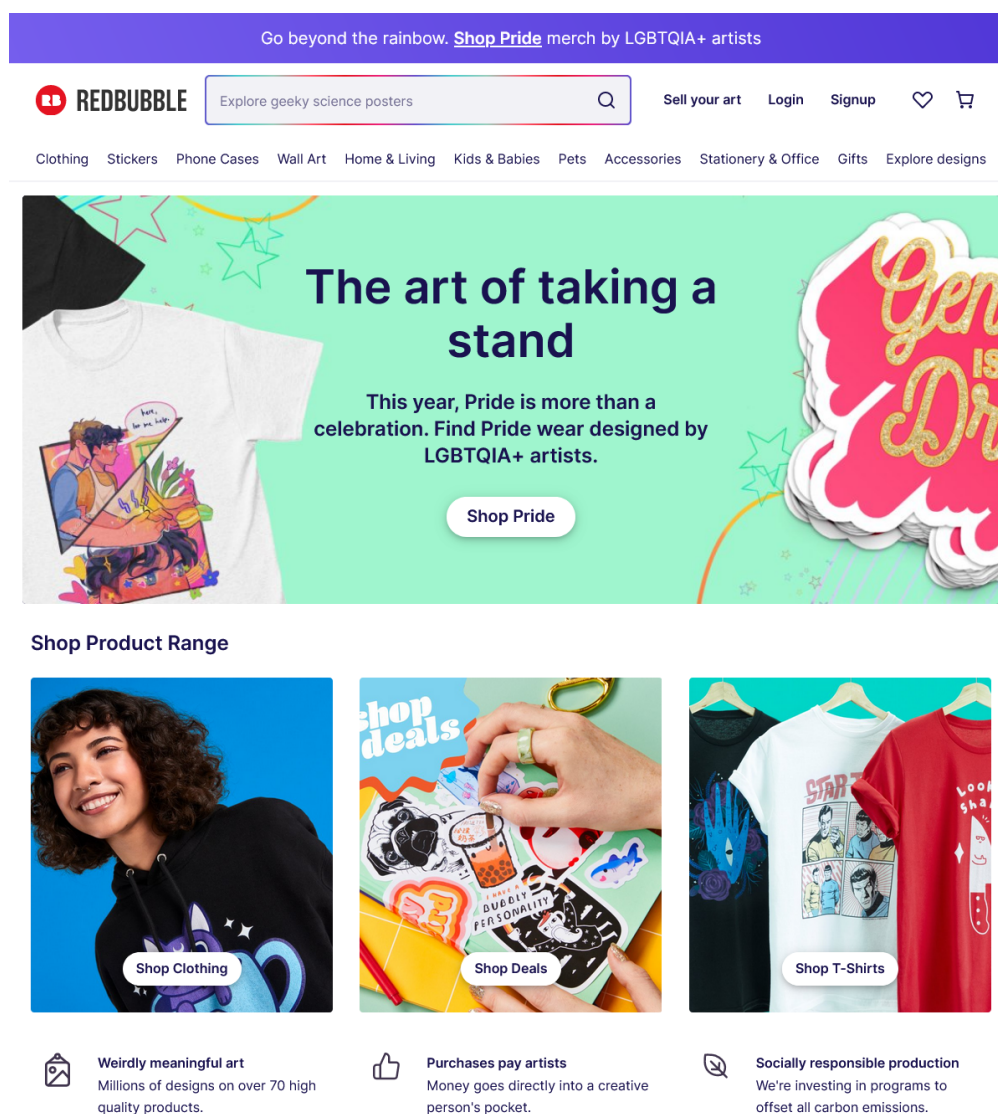
³⁴<https://printify.com/>

limite de criação de 10 lojas, 20% de desconto em todos os produtos e serviço de terceirização de demandas no pós-venda, como reembolso e reimpressão; e empresarial, que não possui preço definido e conta com a criação ilimitada de lojas, além de ter as vantagens apresentadas no plano *premium*. Outro recurso importante oferecido pela plataforma refere-se à ferramenta de integração dos produtos que estão na Printify com outros *marketplaces*, como eBay e Etsy (PRINTIFY, 2023), maximizando as vendas.

Em relação às tarifas impostas sobre a venda de um produto, as mesmas variam de acordo com o preço do frete e da escolha da empresa de impressão, que deve ser escolhida de acordo com a lista de impressores associados à Printify³⁵, possuindo preços variáveis de produção. Logo, o usuário necessita realizar cálculos de margem de lucro, na qual a plataforma indica que os vendedores visem ganhos de 40%, com base nas variáveis expostas para chegar a uma precificação adequada (BARNES, 2023).

4.2.1.3 Redbubble

Figura 47: Página da Redbubble



Fonte: Imagem retirada de <https://www.redbubble.com/> em 16 de Junho de 2023.

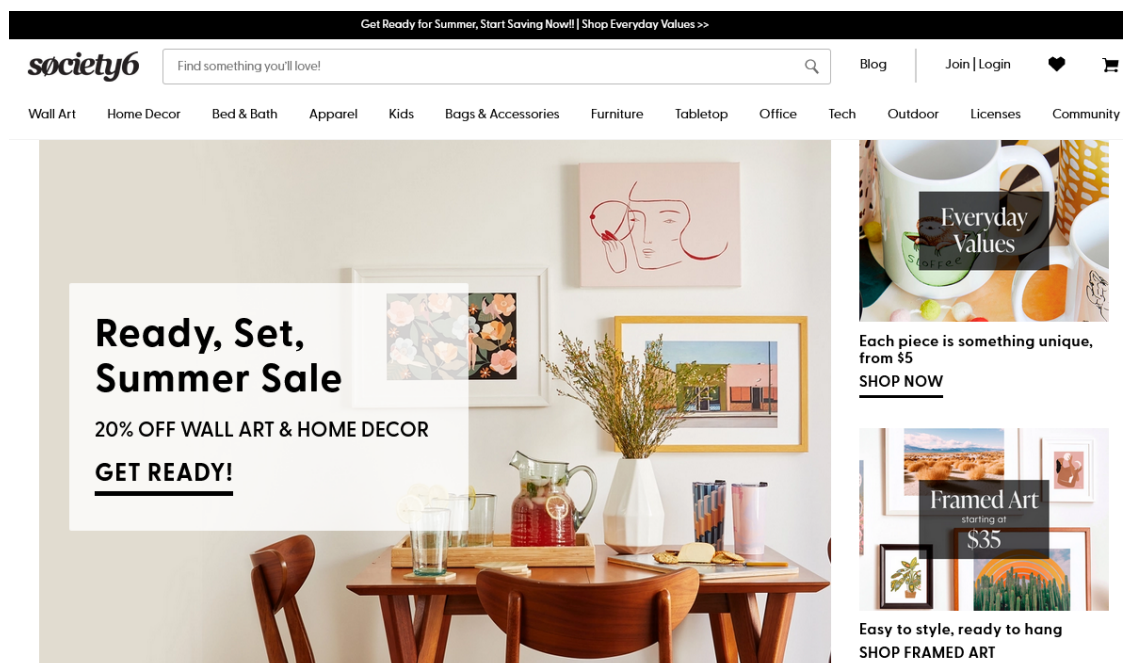
³⁵<https://printify.com/app/print-providers>

Redbubble³⁶(figura 47) é um site que, assim como as outras alternativas já listadas, permite que usuários possam vender suas produções, possuindo um maior foco na venda de roupas, pinturas, papelaria, acessórios e artigos de decoração. Em relação às taxas, são aplicadas cobranças tabeladas em relação à receita mensal do usuário que possui uma conta *standard*, classificação inicial de vendedor, enquanto contas *premium* e *pro* são isentas de tais cobranças por possuírem um maior engajamento na plataforma (REDBUBBLE, 2023).

A Redbubble também funciona de maneira similar à Printify por conta de utilizar produção sob demanda terceirizada, de maneira que um pedido é redirecionado a uma fábrica associada, a qual realiza a materialização da peça criada pelo artista e, então, envia o produto para a transportadora, sendo destinado à residência do comprador (REDBUBBLE, 2022).

4.2.1.4 Society6

Figura 48: Página da Society6



Designed by artists, made by us, just for you.



Fonte: Imagem retirada de <https://www.society6.com/> em 16 de Junho de 2023.

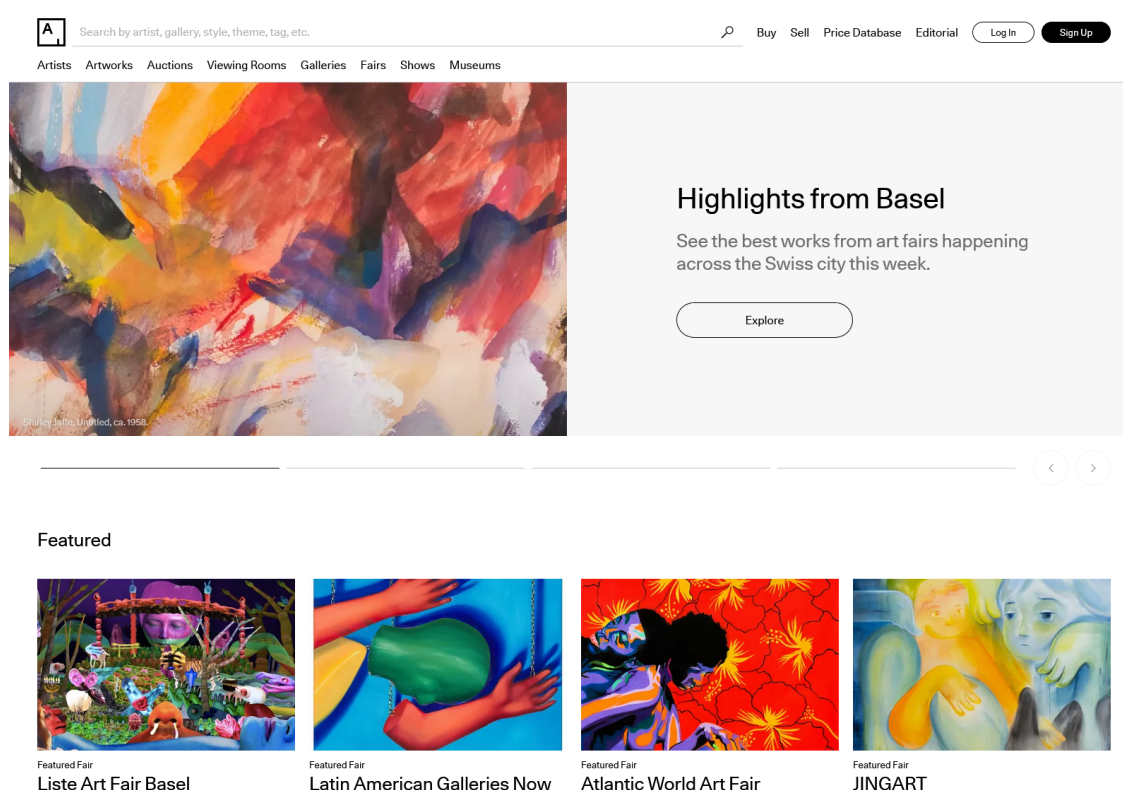
Society6³⁷(figura 48) é uma plataforma destinada à venda de pinturas, vestuário, artigos de decoração e alguns móveis. O site também busca trabalhar com o conceito de produção terceirizada para desonerar o criador, o qual não necessita pagar uma mensalidade para manter uma conta e recebe um percentil fixado em 10% do valor das vendas de todas os itens que não forem associadas a pinturas, permitindo, para esta última categoria, que o artista possa configurar a sua própria margem de lucro (GRAM, 2017).

³⁶<https://www.redbubble.com/>

³⁷<https://society6.com/>

4.2.1.5 Artsy

Figura 49: Página da Artsy



Fonte: Imagem retirada de <https://www.artsy.com/> em 16 de Junho de 2023.

Artsy³⁸ (figura 49) é um site exclusivamente dedicado à venda de pinturas, contando com parcerias com galerias, casas de leilão e museus no mundo inteiro, mas sendo vedada a venda de obras de arte por parte de um criador independente (ARTSY, 2023c). Em relação à forma de venda, a mesma pode se caracterizar tanto por ter um preço já definido pelo parceiro da plataforma, mas permitindo que o comprador possa realizar uma oferta, ou então pode se identificar como um leilão, no qual o comprador deve dar um lance para ter a possibilidade de compra de uma determinada obra de arte. O processo de precificação é realizado com base na análise de dados já existentes na plataforma e em históricos de vendas de obras em leilões, além de levantamento acerca das condições nas quais uma determinada arte se encontra (ARTSY, 2023b). Caso o vendedor opte por anunciar um preço fixado ao invés de leilão, será cobrada uma comissão de 4% sobre o valor de venda da peça (ARTSY, 2023a).

³⁸<https://www.artsy.net/>

5 PROJETO

5.1 Links úteis

- **Plataforma:** <https://facili.art/>
- **Código fonte:** <https://github.com/lunagcezar/faciliart>
- **Devlogs:**
 - **16/08/2023:** <https://youtu.be/F5FTMOxwGxM>
 - **23/08/2023:** <https://youtu.be/LXRfbhdAnq4>
 - **31/08/2023:** <https://youtu.be/4tec-NPmRas>
- **Teaser (25/09/2023):** <https://youtu.be/5KhWicsJb90>
- **Demonstração final (20/11/2023):** <https://youtu.be/TIHB5s3bWMM>

5.2 Escopo do projeto

5.2.1 Requisitos

5.2.1.1 Funcionais

- Cadastrar e realizar o login de usuários.
- Controlar a geração de padrões e figuras geométricas utilizando parâmetros associados às mesmas.
- Determinar que, para cada adição de padrão ou figura geométrica, seja gerada uma camada, facilitando a edição da arte.
- Exibir peças de todos os usuários na página principal.
- Exibir perfil do usuário logado, permitindo a exibição das peças geradas pelo mesmo.
- Gerar arte com a seleção de padrões e figuras geométricas.
- Permitir que o autor de uma arte generativa possa receber uma porcentagem do valor de cada pedido de quadro ou blusa com estampa de sua arte
- Permitir que o usuário classifique uma peça.
- Permitir que o usuário comente uma peça.
- Permitir que o usuário logado possa encomendar um quadro ou blusa com estampa de arte generativa escolhida por meio do pagamento de um valor estipulado pela plataforma.
- Permitir que o usuário logado possa retirar a publicação ou excluir as artes geradas pelo mesmo.
- Salvar a arte gerada no banco de dados para que a mesma seja publicada.

5.2.1.2 Não funcionais

- Armazenamento seguro de credenciais utilizando a biblioteca Devise, de maneira a evitar o comprometimento de dados.
- Uso de navegador atualizado, de forma a evitar falhas de renderização do *front-end*.
- Disponibilidade do servidor o qual hospeda o sistema, permitindo um serviço quase que ininterrupto.
- *Front-end* compatível com os navegadores mais populares, como Google Chrome e Mozilla Firefox.
- Hospedagem em uma plataforma que possua compatibilidade com Rails.

5.3 Design do projeto

5.3.1 Identidade visual

A identidade visual da plataforma buscou transmitir um aspecto de modernidade, passando por processos de criação de uma marca em conjunto com a definição de uma paleta de cores e tipografia. Em relação ao nome, foi cogitado, a princípio, a marca Geniverse (figura 50), uma junção entre *Generative* e *Universe*, de maneira a associar *Generative* à arte generativa e *Universe* às inúmeras possibilidades que a plataforma poderia promover, com alusão ao universo. No entanto, já havia outra plataforma denominada Geniverse³⁹, que consistia em um portal de arte por inteligência artificial. Logo, foi necessária uma mudança de nome para Faciliart (figura 51), uma junção entre a palavra em latim *Facilis* com a palavra *Art*, em inglês. Com esta ideia, foi realizada uma nova sondagem de marca, sendo um melhor nome para o propósito e, então, comprou-se o domínio *facili.art*, possuindo boas características para acesso, visto que se trata de uma URL curta.

Figura 50: Geniverse — Logotipo



Fonte: Elaborado pelo autor

Figura 51: Faciliart — Logotipo

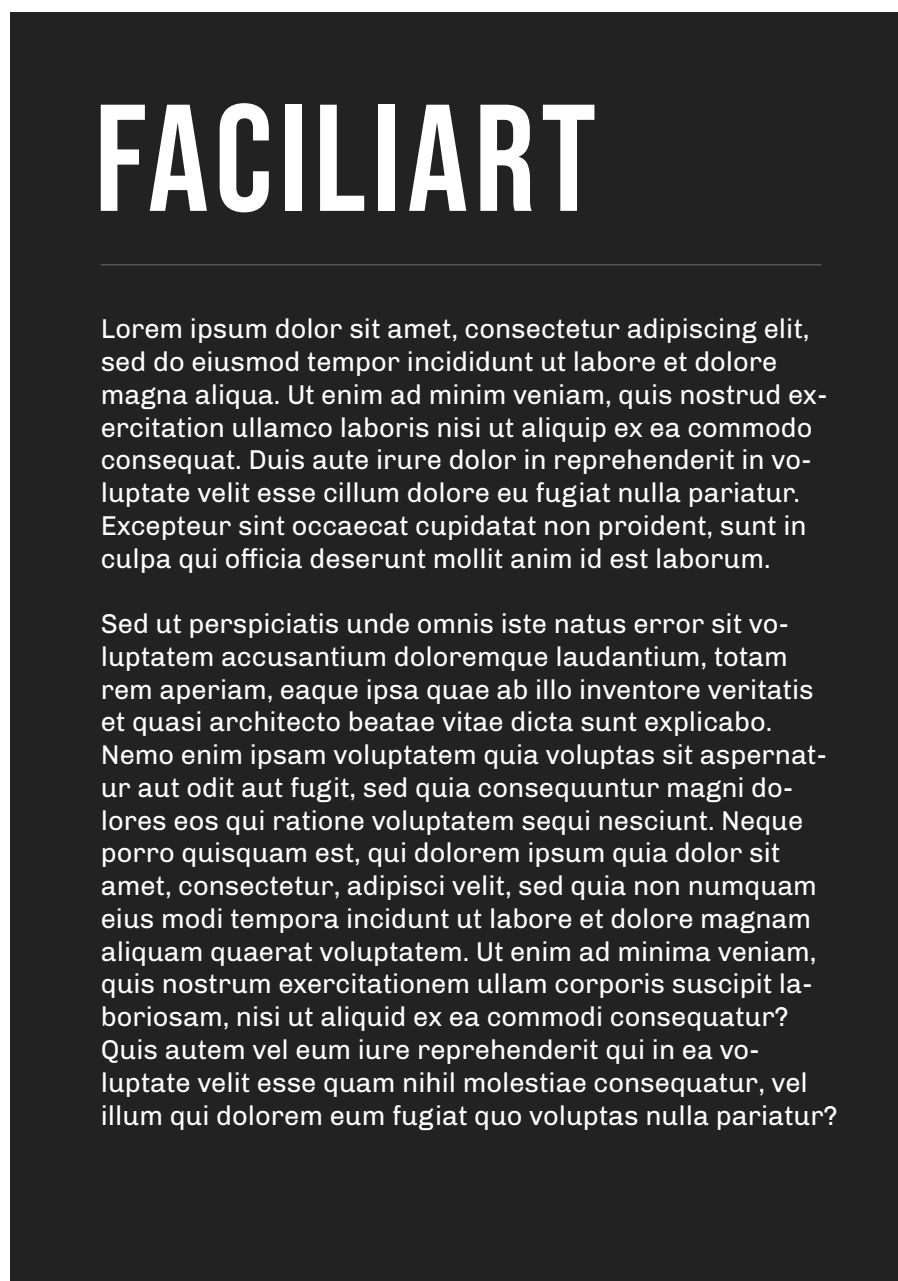


Fonte: Elaborado pelo autor

³⁹<https://geniverse.co/>

Em relação à tipografia (figura 52), foi utilizada a fonte Bebas Neue Regular⁴⁰, da Dharma Type, no logotipo por não possuir serifa, ser condensada e possuir um traçado mais espesso, a fim de transmitir a ideia de algo moderno e imediato. No entanto, como o espaçamento entre caracteres era muito curto, foi aplicado um *tracking* de 40 unidades ao logotipo. No corpo de texto, foi adotada a fonte Chivo⁴¹, da Omnibus Type, que tem um foco na legibilidade, assim como possui semelhanças com a fonte do logotipo, nas quais ambas são neo-grotescas.

Figura 52: Faciliart — Tipografia do projeto



Fonte: Elaborado pelo autor

⁴⁰<https://github.com/dharmatype/Bebas-Neue>

⁴¹<https://www.omnibus-type.com/fonts/chivo/>

A paleta de cores da plataforma (figura 53) possuiu com o objetivo de representar o contexto de modernidade, sendo utilizada um esquema de cores escuro para elementos como o *background* da marca e de conteúdo. Outras cores na paleta incluem uma neutra para linhas e *background* da área de edição, além do branco para o corpo de texto.

Figura 53: Faciliart — Paleta de Cores

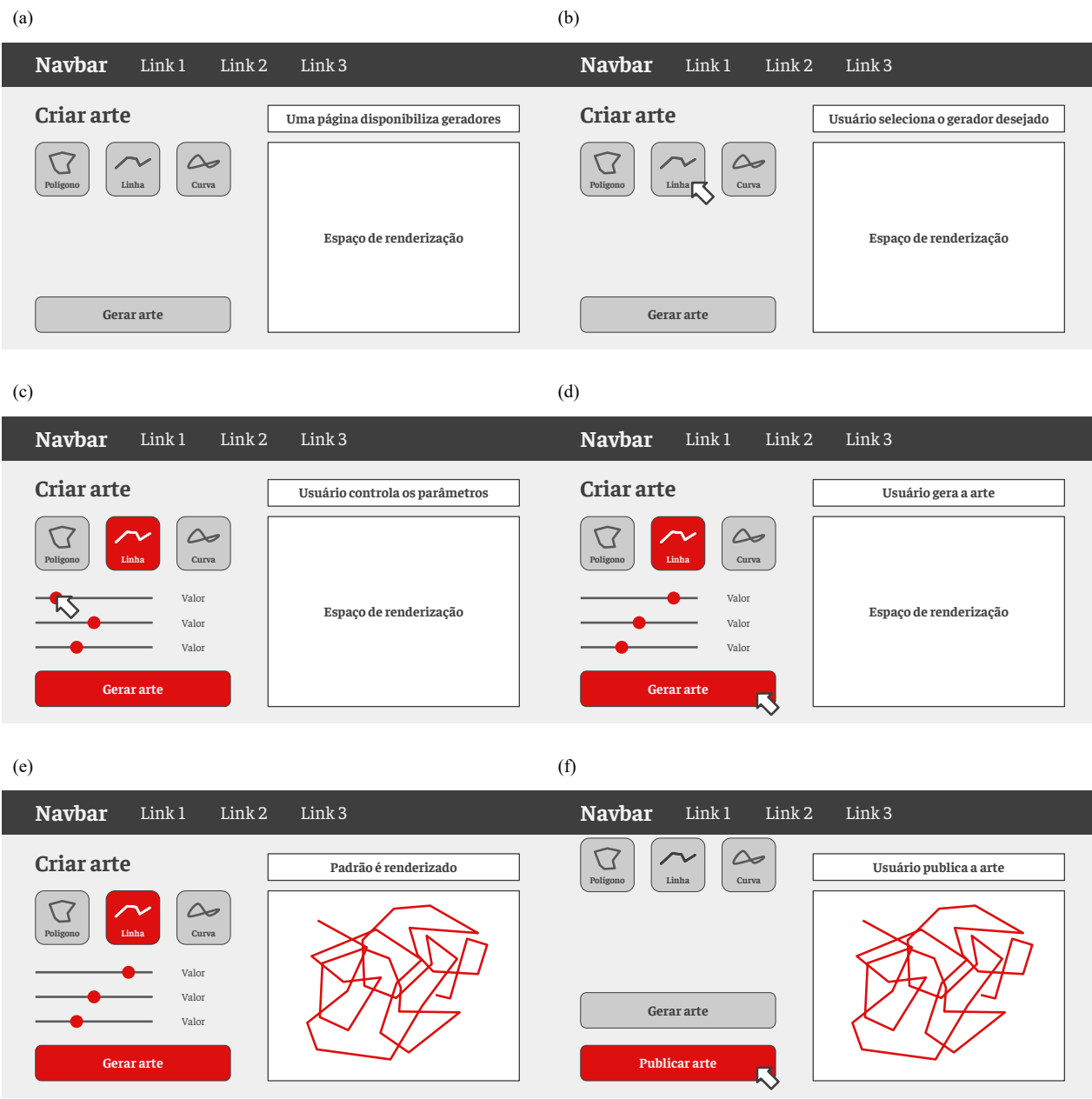
Background da marca	base-100
#0f0f0f	
Background de conteúdo	base-300
#222222	
Linhas e background de editor	neutral
#555555	
Texto	text
#FFFFFF	

Fonte: Elaborado pelo autor

5.3.2 Interface de usuário

A criação da interface de usuário iniciou-se com a apresentação de um *mockup* na apresentação de qualificação deste trabalho em 2023.1, sendo proposta, para a tela de edição (figura 54), uma página com algumas similaridades em relação a softwares de edição de imagem, apresentando seleção de geradores e parâmetros à esquerda, e renderização da arte à direita. Nesta versão, tanto a seleção de geradores quanto a de parâmetros, exibida após a seleção de um gerador, coexistiam na mesma região da tela. Em relação à tela de listagem de artes (figura 55), foi definida a exibição das mesmas por meio de *thumbnails*, as quais teriam links embutidos para a página de visualização da referida arte. E, então, com a tela de visualização de tal arte (figura 56), foi proposta a ideia de comercialização da mesma por meio da exibição de opções de itens disponíveis, assim como o tamanho desejado para o artigo escolhido.

Figura 54: Página de edição de arte — Mockup



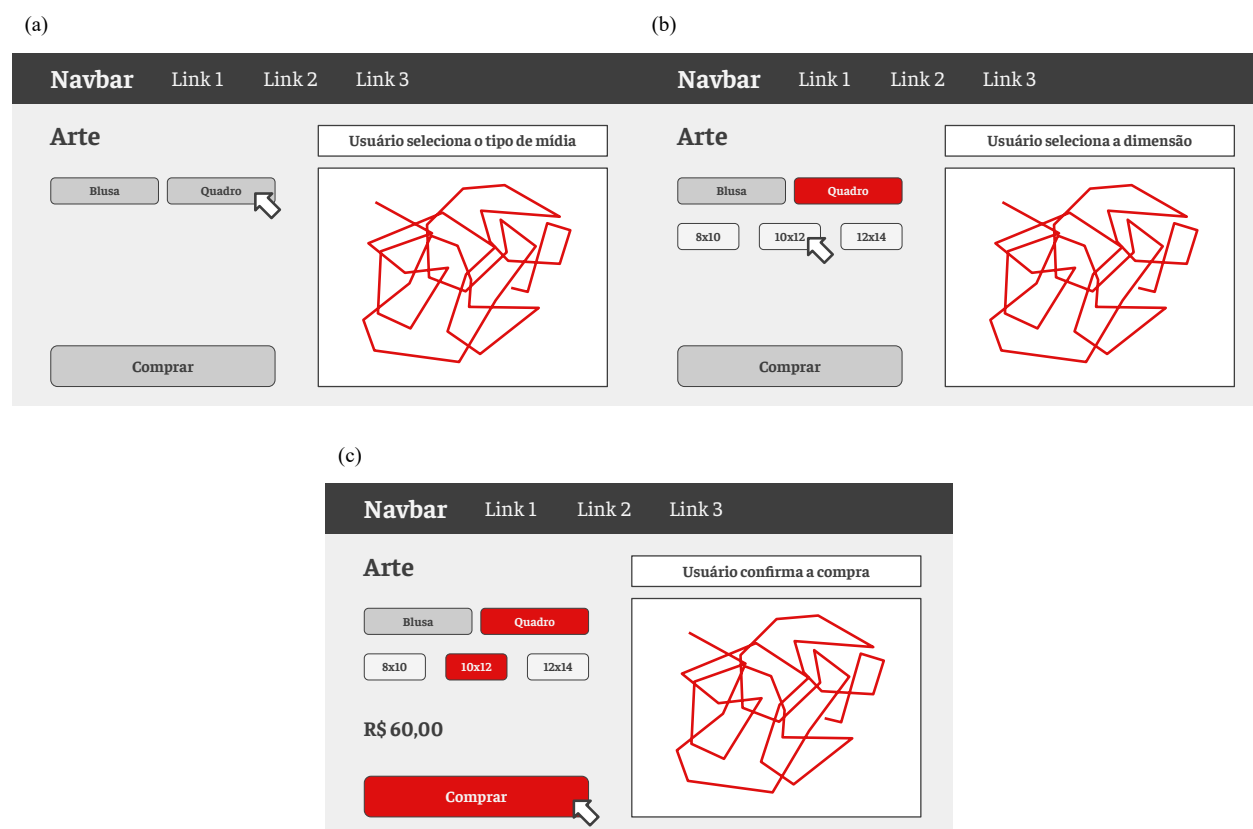
Fonte: Elaborado pelo autor

Figura 55: Página de listagem de artes — Mockup



Fonte: Elaborado pelo autor

Figura 56: Página de visualização de arte — Mockup

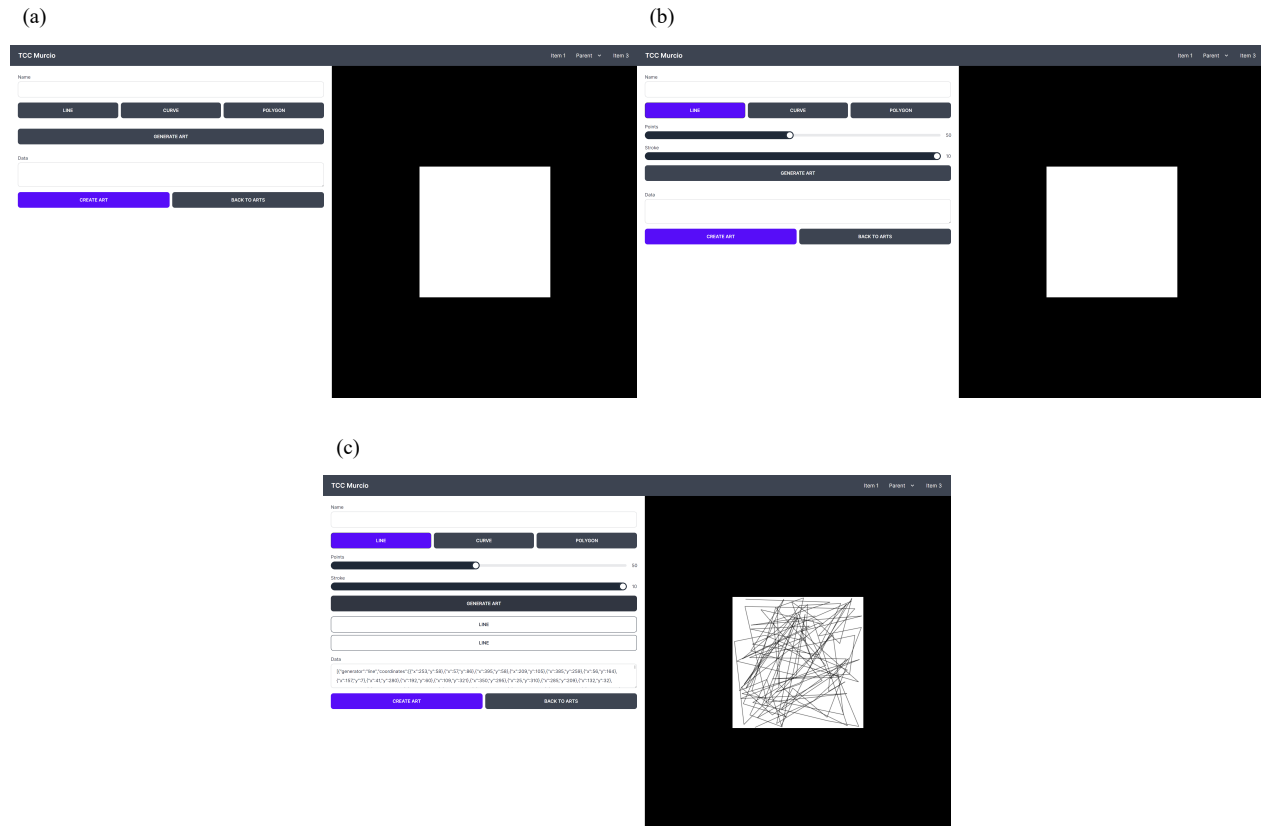


Fonte: Elaborado pelo autor

Com isto, iniciou-se a diagramação da interface por meio do desenvolvimento *front-end* da tela de edição de arte (figura 57), a qual possuía uma proposta de interface similar à apresentada antes, com a utilização de seletores de geradores em uma região da tela que, ao serem clicados exibiam seus parâmetros. No entanto, após avaliações, noticiou-se que esta ideia poderia não ser uma proposta escalável por não existir muito espaço reservado para todo conteúdo, visto que poderiam ser criados novos

geradores e parâmetros, ocupando muito espaço em uma seção na tela, podendo gerar um *overflow* vertical indesejado.

Figura 57: Página de edição de arte — Primeira implementação



Fonte: Elaborado pelo autor

Desta forma, foi idealizado um novo tipo de conceito para a tela de edição (figura 58), no qual consistia em dividir tais etapas de criação de camadas em uma sequência de exibição de elementos, necessitando que o usuário clicasse, inicialmente, no botão “New Layer”, para depois o utilizador ter acesso aos geradores disponíveis, e então, ao selecionar o gerador, houvesse a exibição dos parâmetros inerentes ao mesmo que, após a edição destes, o usuário pudesse criar a camada clicando em “Create Layer”, de maneira a resolver o problema de falta de espaço em um caso de haver muitos geradores ou parâmetros.

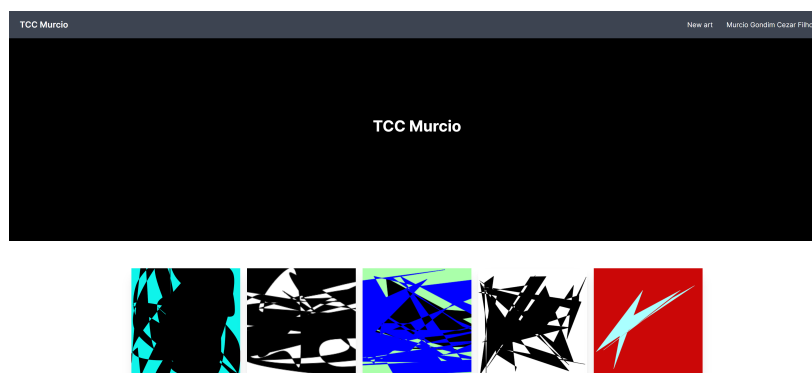
Figura 58: Página de edição de arte — Segunda implementação



Fonte: Elaborado pelo autor

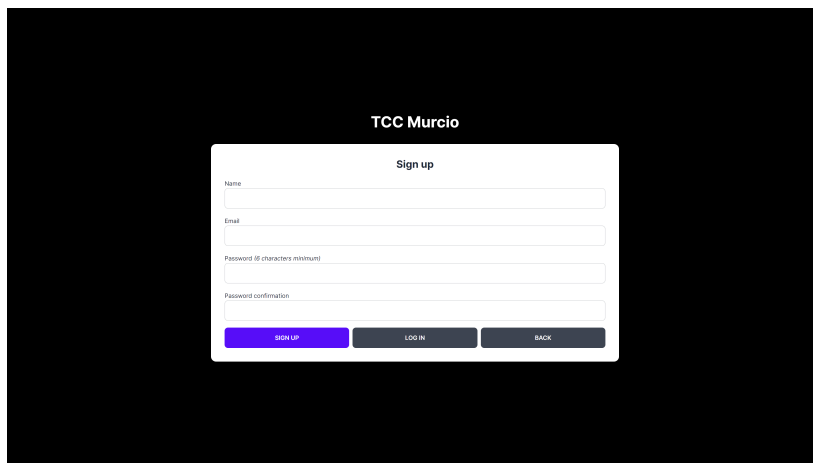
Logo, realizou-se a diagramação da página inicial (figura 59), que seguiu parcialmente o que tinha sido conceituado na apresentação de qualificação, possuindo uma grade de *thumbnails* que serviriam de atalhos para a visualização completa das artes. No entanto, foi adicionado um *hero* com uma frase como slogan, ocupando 50% de altura da tela. Como, aproximadamente nesta etapa, houve a adição de funcionalidade de registro de usuários, também foi feito um *layout* (figura 60) para tal caso de uso.

Figura 59: Página inicial — Primeira implementação



Fonte: Elaborado pelo autor

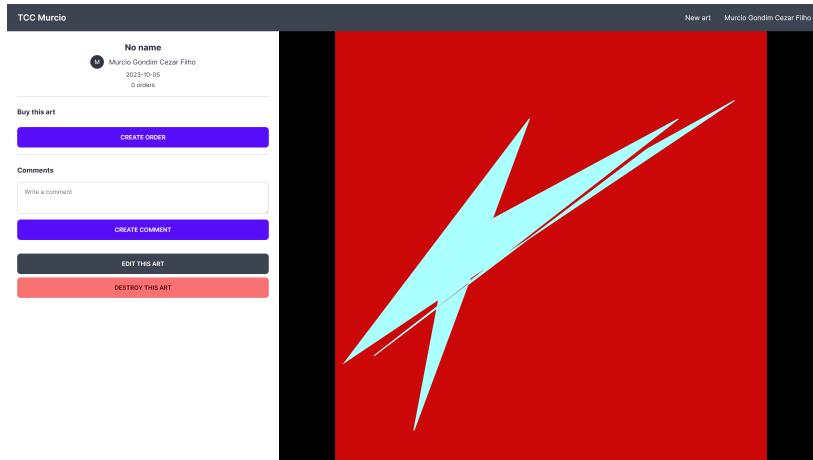
Figura 60: Layout de registro — Primeira implementação



Fonte: Elaborado pelo autor

Então, seguiu-se para a concepção da tela de visualização de arte (figura 61), a qual possuiria dados como o nome da obra e do autor, além de espaço para compra de artigos com a arte em questão, assim como local para adição e exibição de comentários. Neste caso, o layout, em geral, permaneceu como algo similar com o que tinha sido desenvolvido na tela de edição, com duas colunas, sendo a menor, da esquerda, contendo os dados da obra, e a maior, da direita, possuindo a obra propriamente dita.

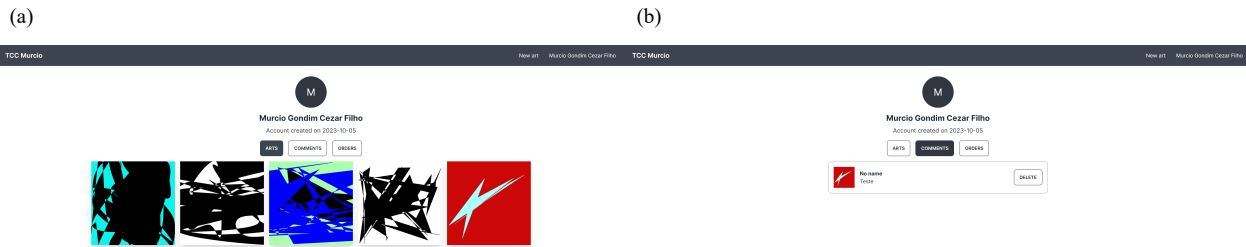
Figura 61: Página de visualização de arte — Primeira implementação



Fonte: Elaborado pelo autor

Portanto, foi iniciada a diagramação do perfil de usuário (figura 62), que permitia ao mesmo visualizar todas as suas obras, assim como os comentários e pedidos de maneira simples. Para isto, foram utilizados botões com a funcionalidade de abas, na qual, ao clicar, ocultava a seção na qual o utilizador estava, e exibia a que foi selecionada. A ideia de alternância de itens em apenas uma página é benéfica para a usabilidade da plataforma, visto que tal proporciona maior performance do que a realização de seções de informações do usuário em páginas separadas, visto que, neste último, necessita um carregamento de uma nova página a cada ação, prejudicando a performance da aplicação, enquanto, na alternância de itens, não há um novo *refresh*, visto que todas as informações já estão na mesma página, apenas estão ocultas.

Figura 62: Página de perfil do usuário — Primeira implementação



Assim, durante este tempo, foi realizado um conjunto de *sketches* que buscavam justificar as escolhas de interface para o projeto, no qual o mesmo buscou seguir decisões de interface conservadoras em relação a softwares de edição e alguns sites de arte similares, como o Behance, Dribbble e DeviantArt. Buscou haver certa padronização de conceitos de interface em relação às alternativas nas páginas do projeto para que não houvesse confusão em seu uso. Logo, as páginas de visualização e listagem de artes, além da de perfil de usuário buscaram ficar próximas às outras opções de portais, enquanto a página de edição de arte teve um maior viés em relação aos atuais softwares de edição de imagem, que, geralmente, possuem uma coluna destinada à edição de parâmetros do arquivo.

GENIVERSE

GENERATIVE UNIVERSE

→ FONTE: BEBAS NEVE REGULAR

→ TRACKING: 40

Fontes de
Lorem Ipsum Dolor ... → CORPO DE TEXTO: CHIVO

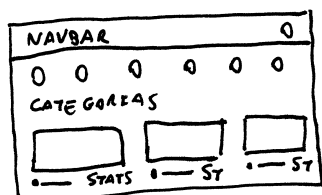
SKETCH

DOCUMENTAÇÃO

05/09/2023

HOME → INSPIRAÇÕES: BEHANCE / DEVIANT ART / OPENPROCESSING / Dribbble

BEHANCE



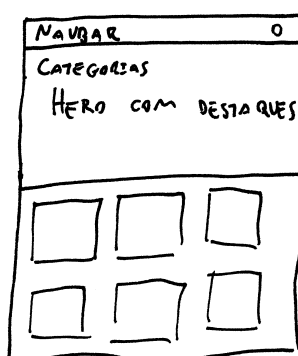
- NOME EXIBIDO EM HOVER
- USUÁRIO E ESTATÍSTICAS FORA DA ÁREA DE ARTE

DEVIANT ART



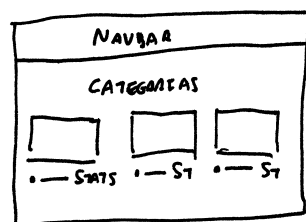
- TODAS INFORMAÇÕES EM HOVER

OPENPROCESSING



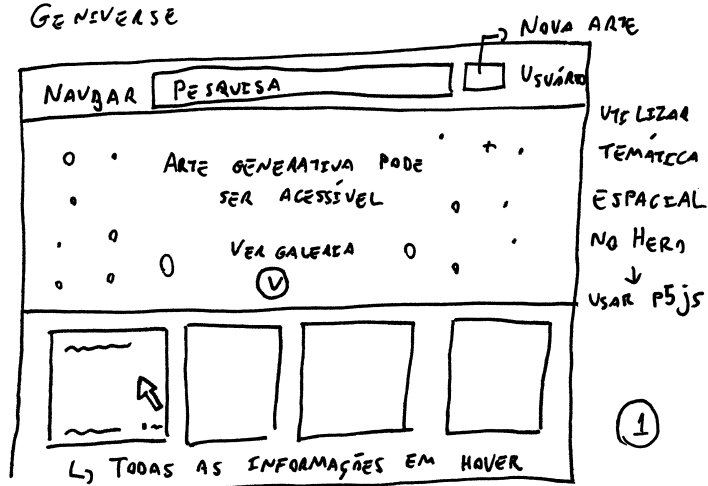
- TODAS INFORMAÇÕES EM HOVER

Dribbble



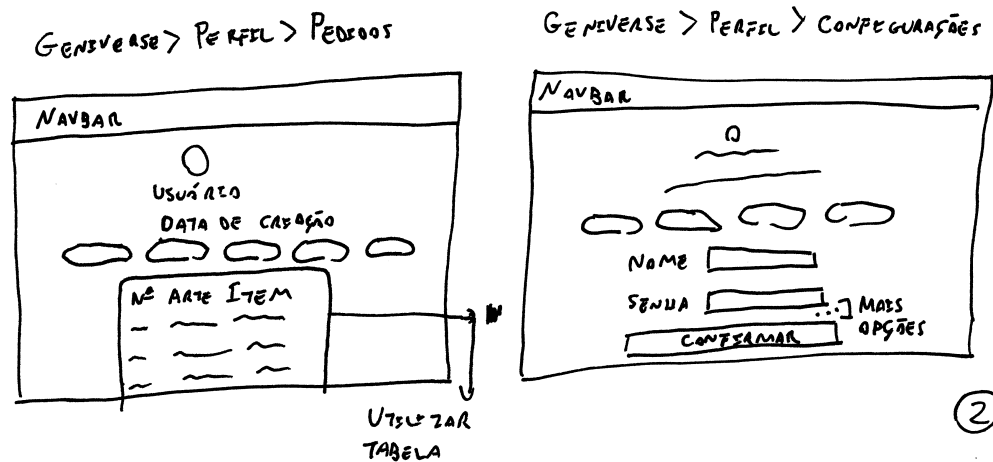
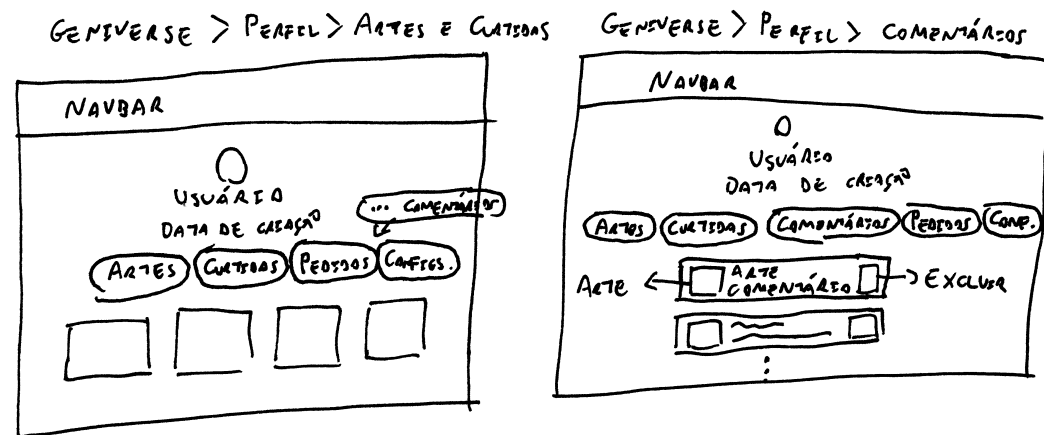
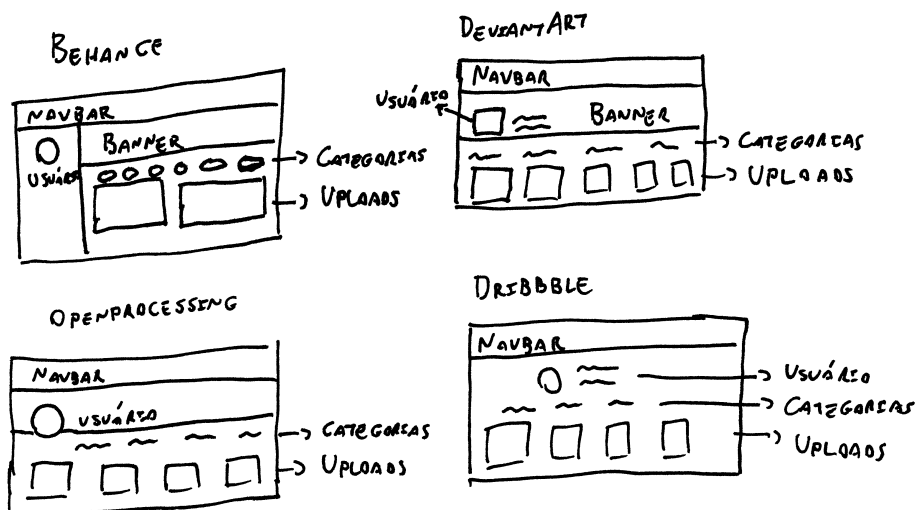
- NOME EXIBIDO EM HOVER
- USUÁRIO E ESTATÍSTICAS FORA DA ÁREA DA ARTE

GENIVERSE

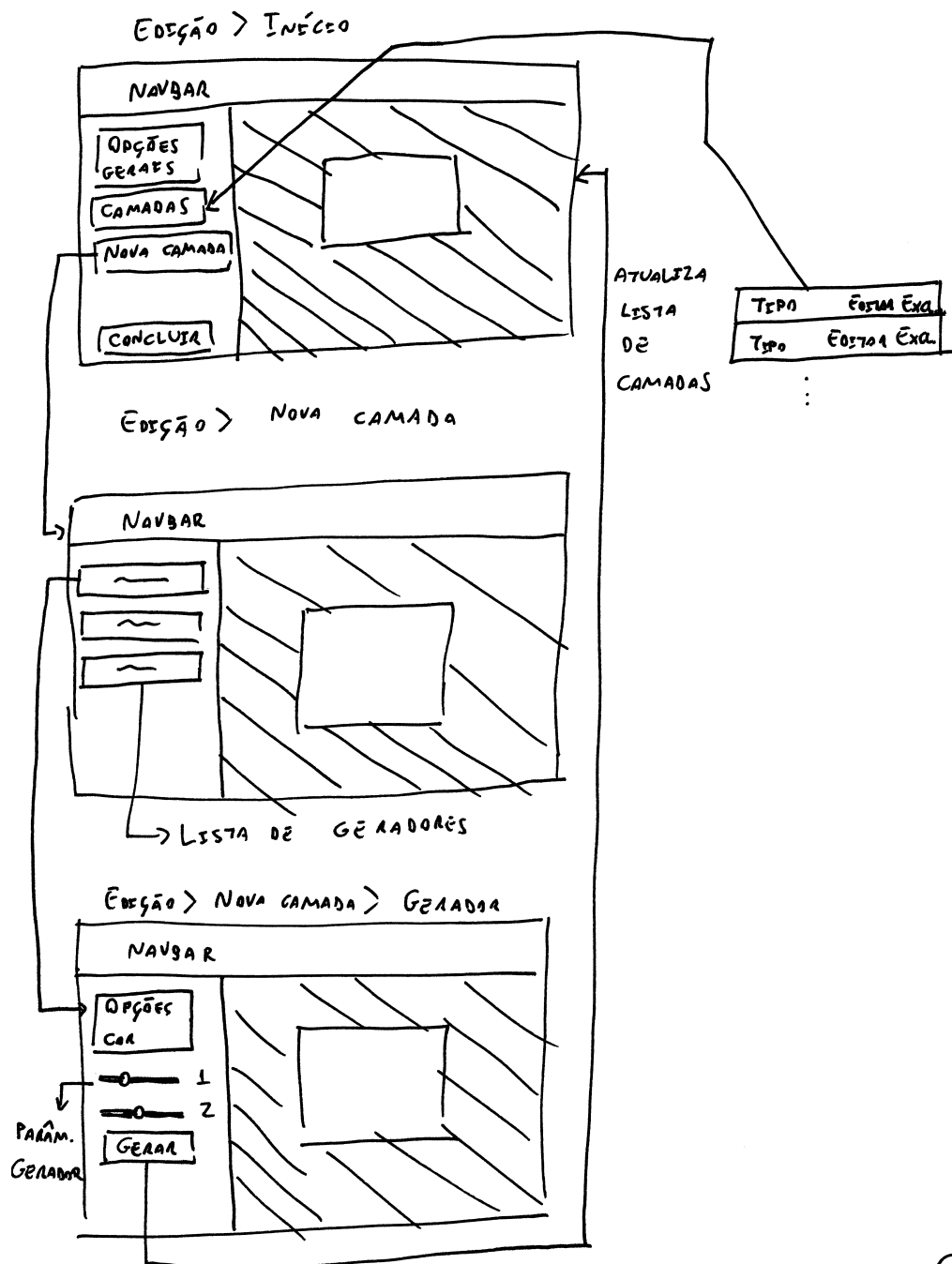


SE POSSÍVEL, LAZY LOADING (SCROLL INFINITO)

PERFIL → INSPIRAÇÕES: BEHANCE / DEVIANART / OPENPROCESSING / DRIBBBLE

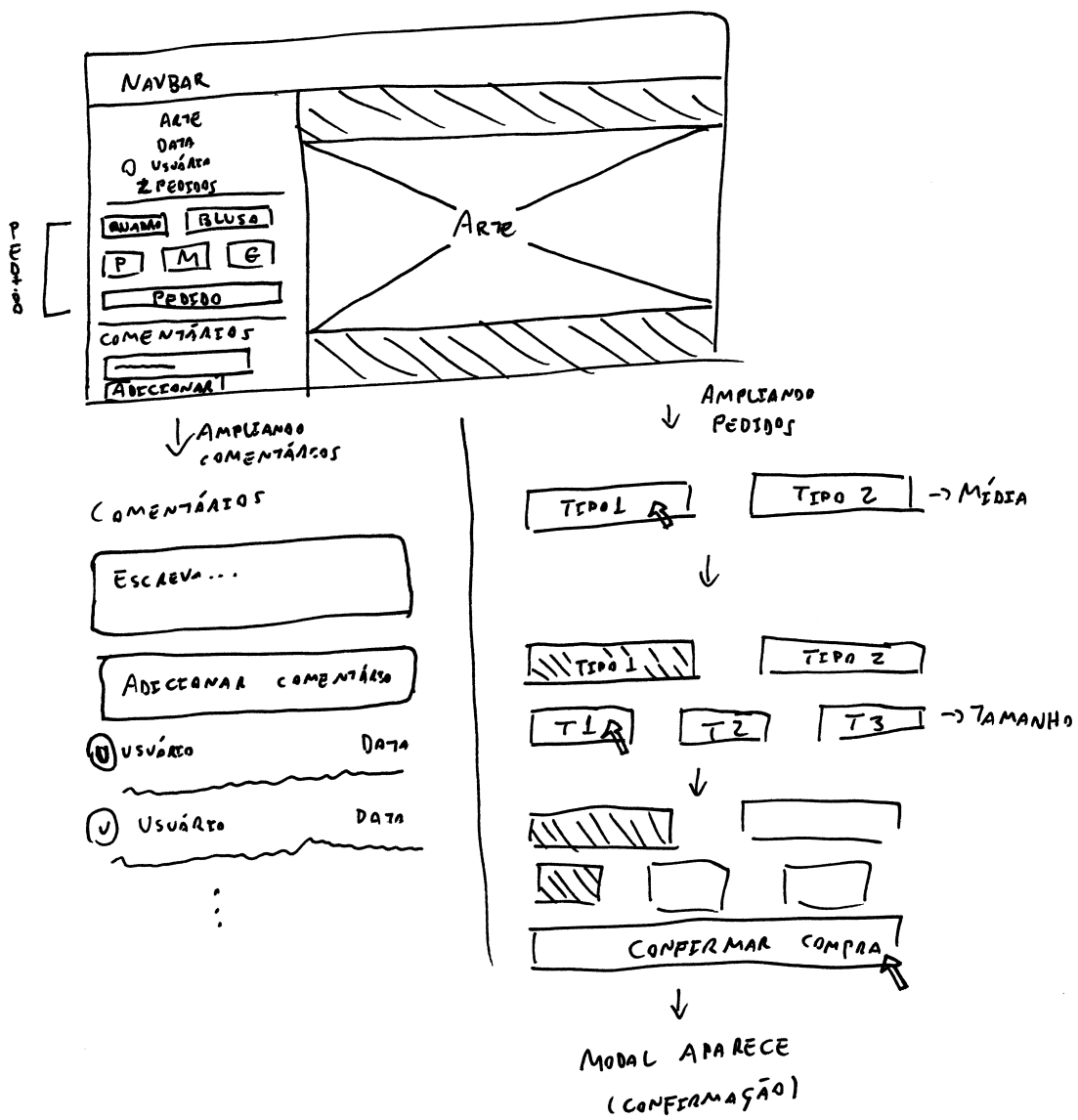


EDIÇÃO → INSPIRAÇÕES: SOFTWARES DE EDIÇÃO EM GERAL

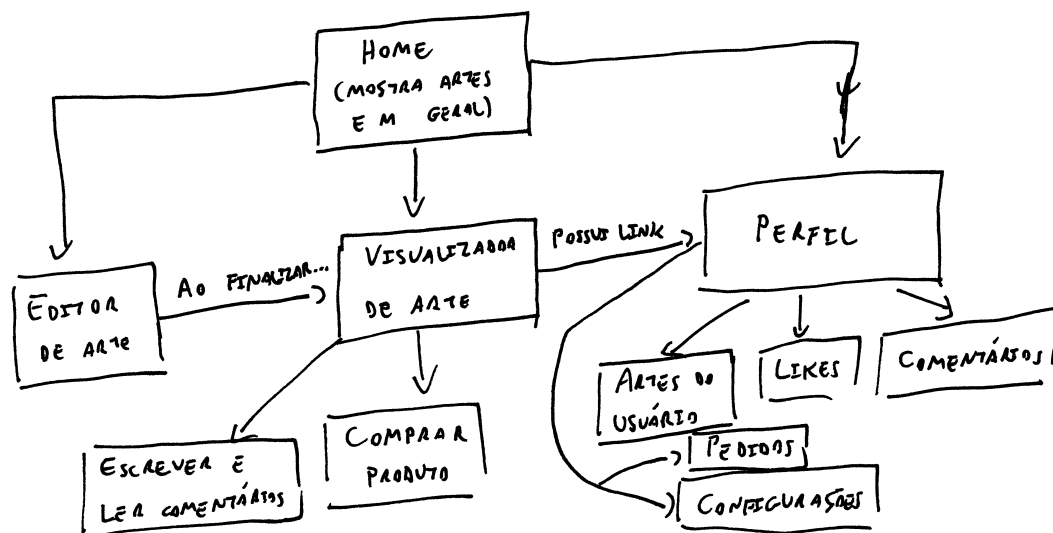


(3)

VISUALIZAÇÃO DE ARTE : DIAGRAMAÇÃO DERIVADA DA PÁGINA DE EDIÇÃO



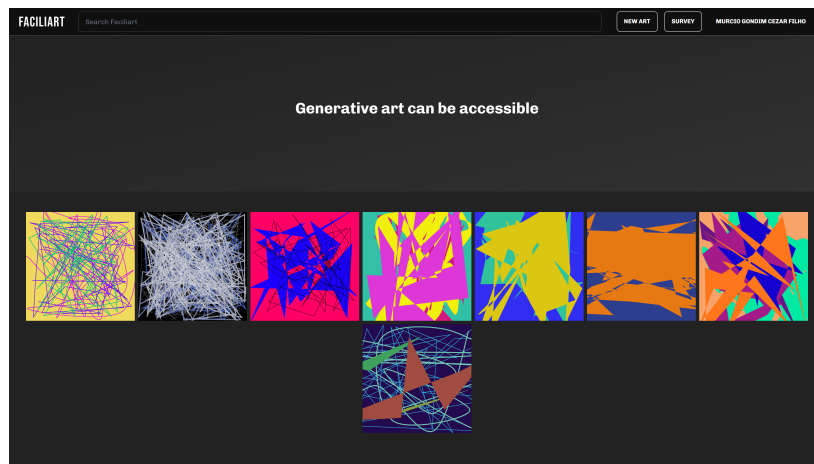
MAPA DA PLATAFORMA



(5)

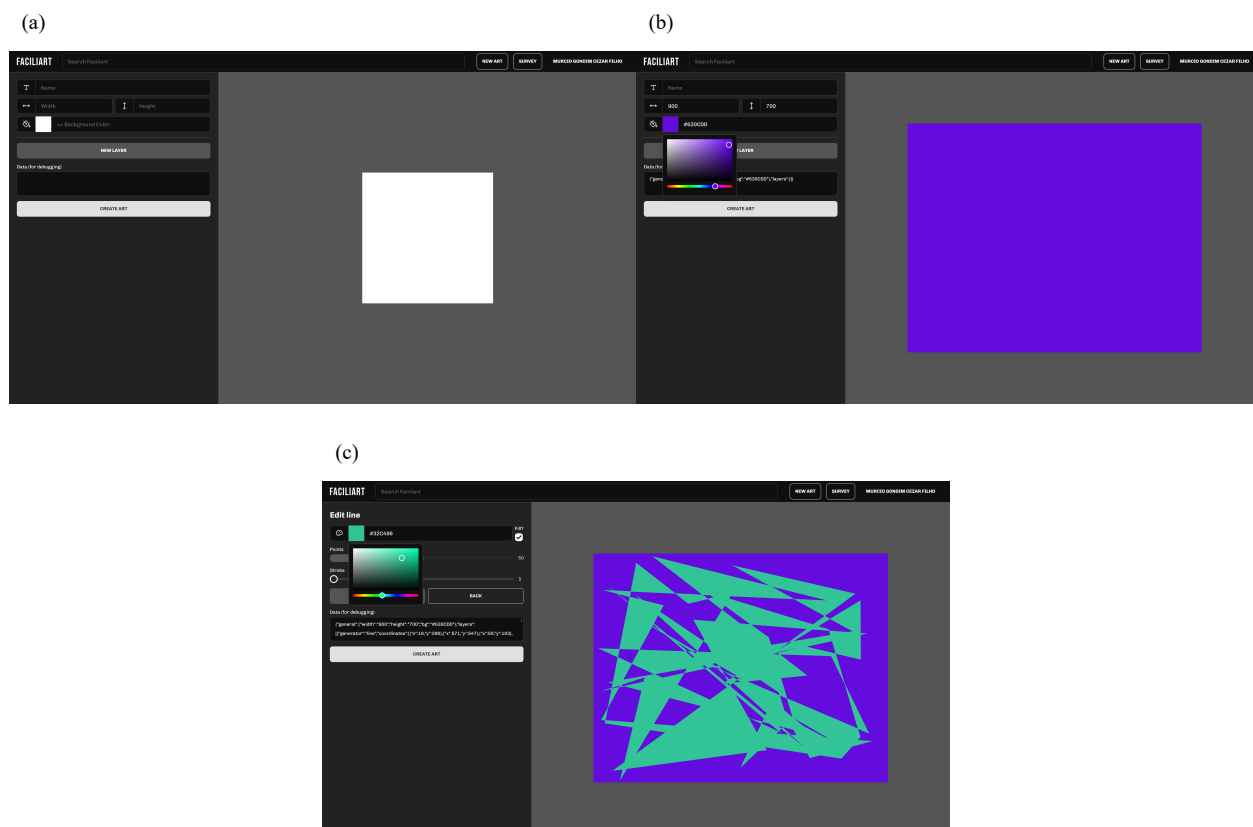
Com isto, começou a ser desenvolvido o aprimoramento de visual da interface gráfica com a aplicação das cores e tipografias definidas no *branding* do projeto para a maioria dos componentes de interface existentes, resultando em uma experiência muito mais coerente e consistente com a proposta do projeto. Além disso, houve a reformulação dos *inputs* existentes nas páginas da plataforma para um formato mais simples, retirando o *label* de cada campo, localizado acima do *input*, para um conjunto unificado de ícone com *input*, sendo melhor detalhado com a adição de um *placeholder* no local de edição.

Figura 63: Página inicial — Implementação final



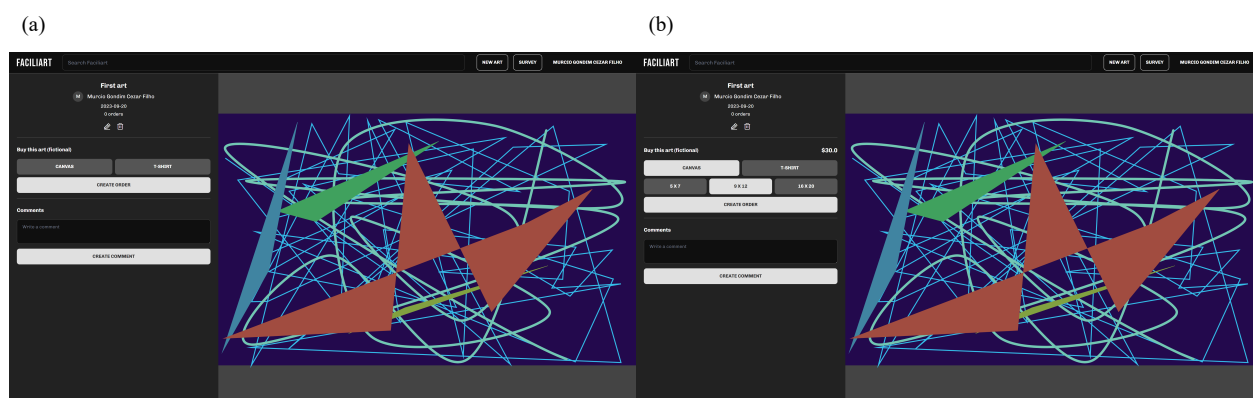
Fonte: Elaborado pelo autor

Figura 64: Página de edição de arte — Implementação final



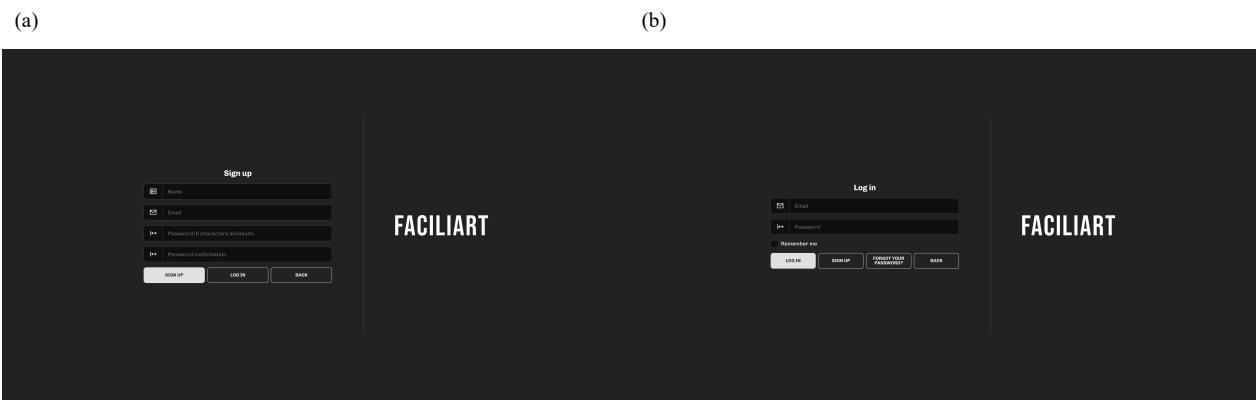
Fonte: Elaborado pelo autor

Figura 65: Página de visualização de arte — Implementação final



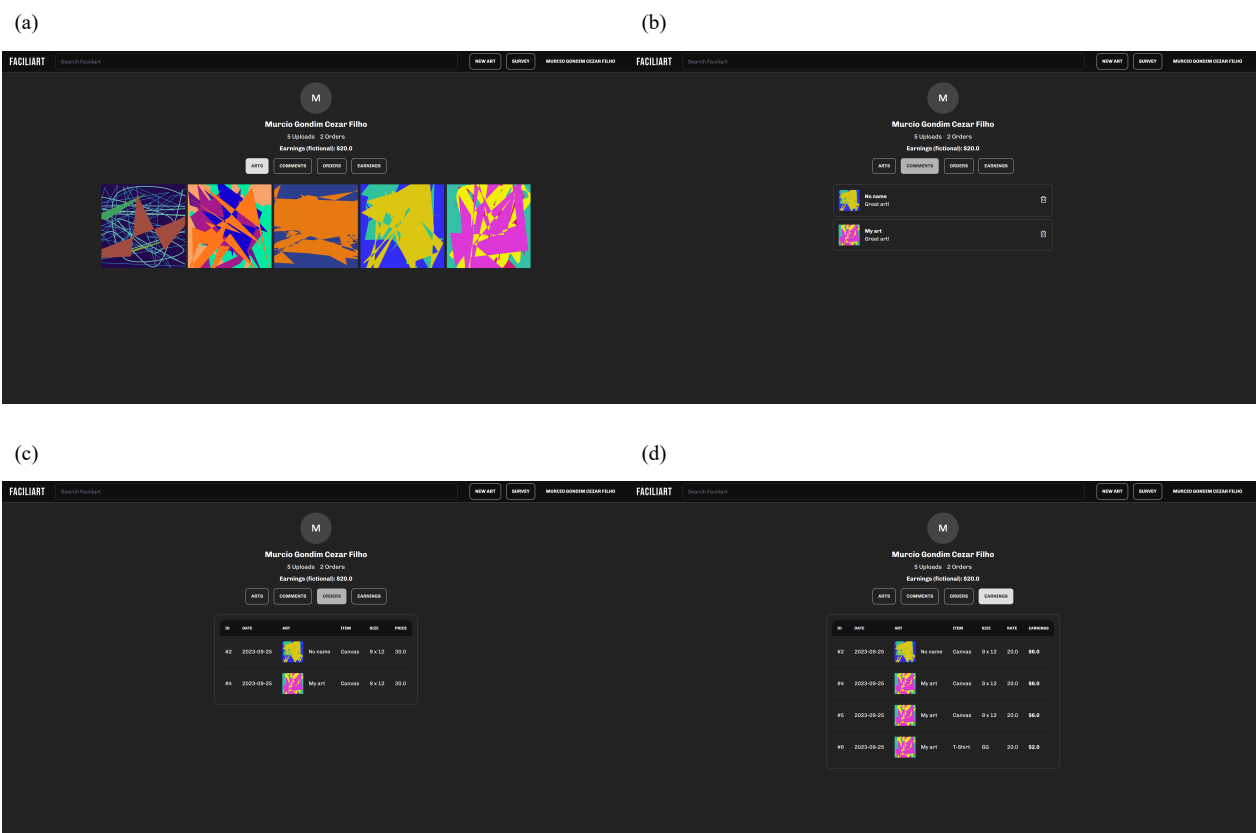
Fonte: Elaborado pelo autor

Figura 66: Páginas de registro e login — Implementação final



Fonte: Elaborado pelo autor

Figura 67: Página de perfil de usuário — Implementação final

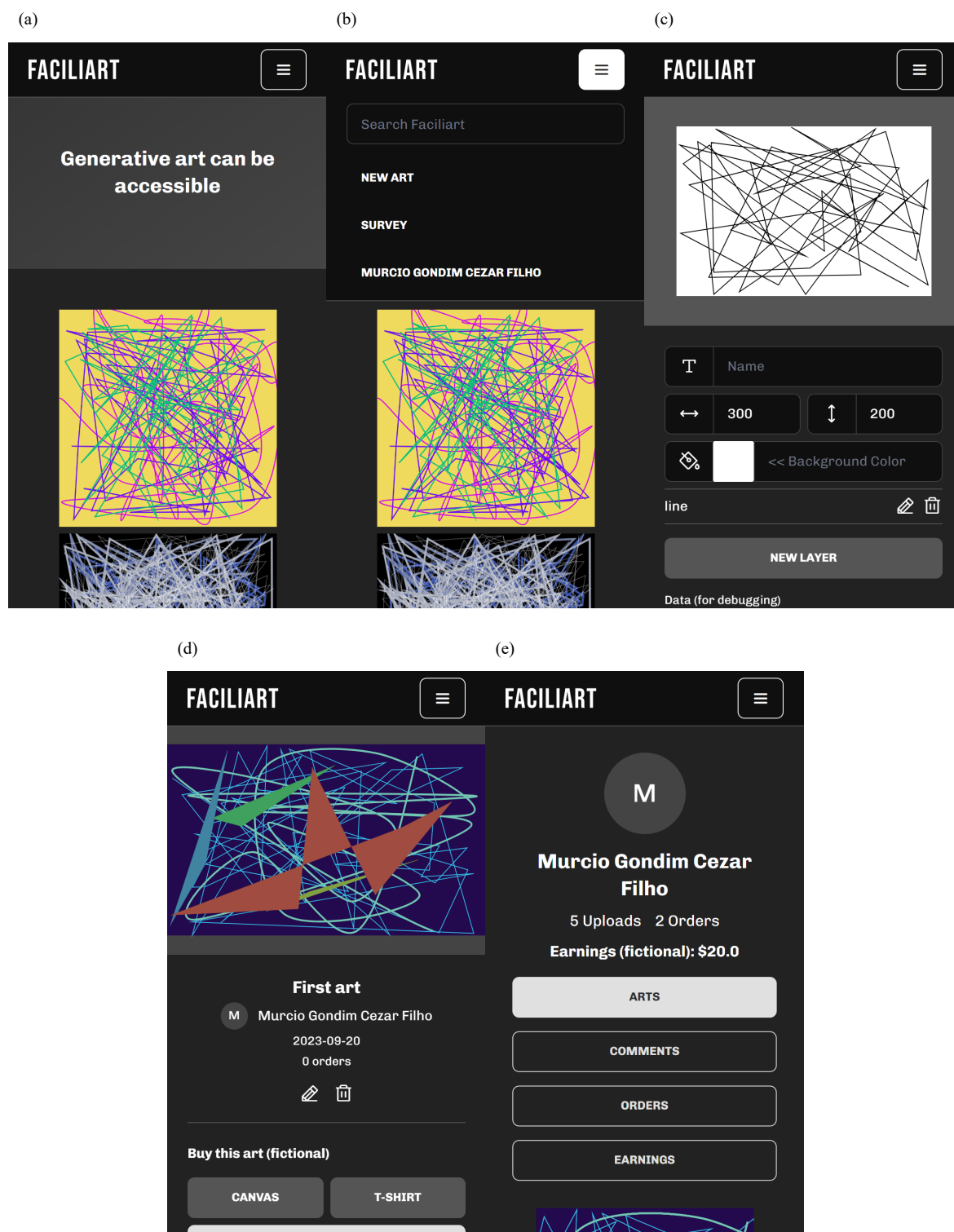


Fonte: Elaborado pelo autor

Com estas modificações realizadas, prosseguiu-se para a implementação de uma interface responsiva figura 68, visto que, até então, o projeto não contemplava dispositivos móveis. Desta forma, para as páginas de exibição e edição de artes de artes, além da de login e registro, foi necessária a conversão de colunas para linhas a partir de um *breakpoint* determinado, de maneira que fosse possível promover uma melhora no uso para este tipo de caso. Também foi adicionado um menu hambúrguer para expandir a

barra de navegação em casos de telas pequenas.

Figura 68: Versão *mobile* da plataforma



Fonte: Elaborado pelo autor

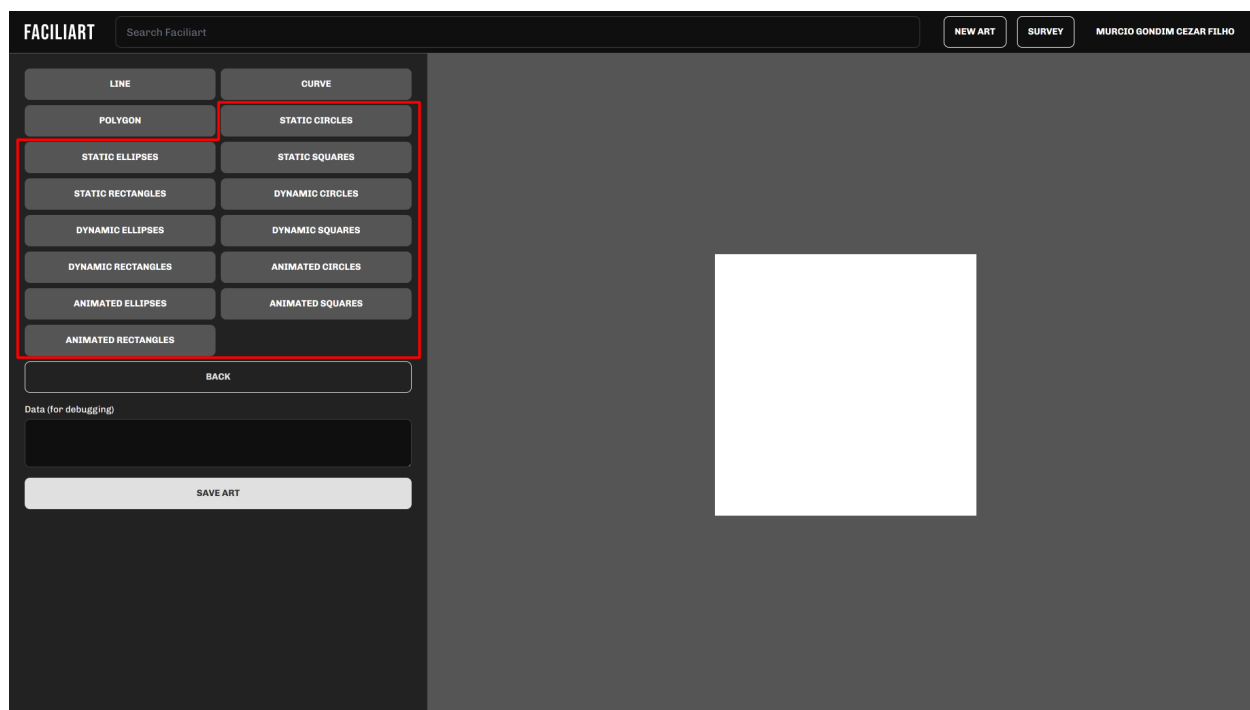
Após isso, o projeto atingiu o marco do MVP (Mínimo Produto Viável), possibilitando que a aplicação fosse submetida a testes por meio da hospedagem do projeto, possibilitando o acesso público, e a disponibilização de um teste de usabilidade (ver seção 5.6), sendo possível compilar métricas e *feedbacks* de participantes.

Com isto, foram realizadas modificações (figura 69) focadas no editor de arte que melhoravam, de maneira significativa, a usabilidade da plataforma, sendo citadas abaixo:

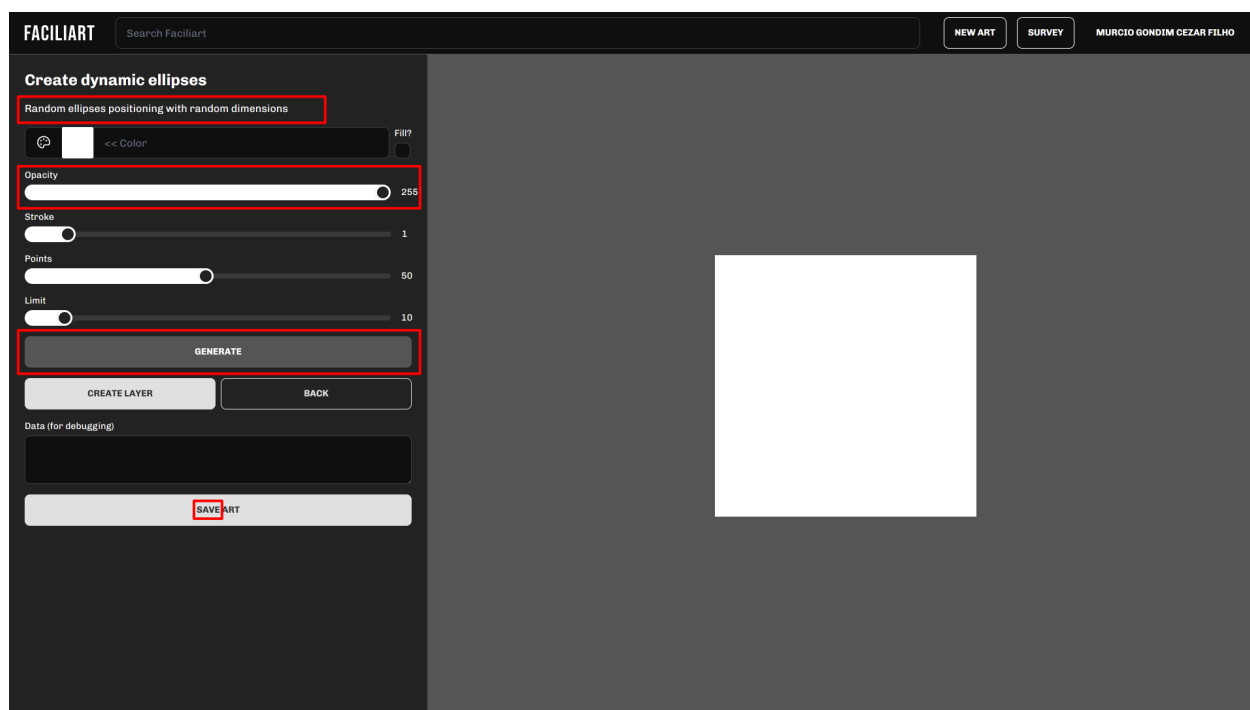
- Adição de descrição de gerador de arte no momento em que os parâmetros do mesmo são exibidos, permitindo que o usuário não possua dúvidas acerca do conceito de um gerador;
- Adição de seletor de opacidade para cada camada, permitindo maiores possibilidades de estética;
- Adição de novos geradores, como de quadrados, retângulos, círculos e elipses;
- Adição de um botão que permita com que o usuário gere o padrão desejado antes de salvar a camada, permitindo que o usuário veja o formato gerado antes de salvar em uma camada e, caso o utilizador não esteja satisfeito com a geração, apenas clicar no mesmo botão;
- Mudança do botão “*Create art*” para “*Save art*”, com maior adequação ao contexto.

Figura 69: Alterações finais na tela de edição

(a) Lista de camadas



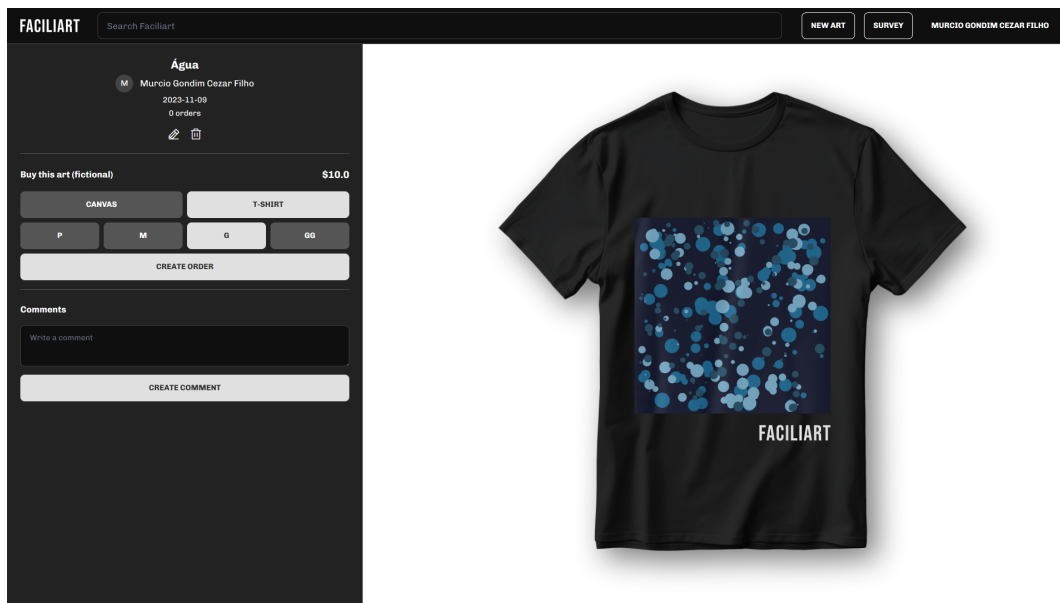
(b) Edição de camada



Fonte: Elaborado pelo autor

Após isso, por meio de sugestões, foi elaborado um *mockup* (figura 70), ou seja, um conceito de um recurso ainda não implementado, de uma funcionalidade de *preview* da peça a ser comprada com a arte escolhida. Neste caso, tal *feature* poderia ser implementada após o usuário clicar no tipo de artigo desejado, como um blusa, de maneira a exibir uma prévia de como ficaria a impressão de tal arte na blusa, aprimorando a experiência do usuário de maneira significativa.

Figura 70: Mockup de compra de blusa na plataforma Faciliart



Fonte: Interface e arte elaboradas pelo autor. Template de blusa de autoria de vector_corp

5.4 Implementação do projeto

5.4.1 Tecnologias utilizadas

Para a concepção do projeto, foi utilizado um conjunto de ferramentas de desenvolvimento web que permitiram que fosse realizado um desenvolvimento ágil do referido projeto. Logo, foi utilizado, no *back-end*, o Ruby on Rails, que permitiu uma rápida construção da estrutura da aplicação e de dados, incluindo a lógica necessária para o armazenamento das informações no banco de dados, permitindo a persistência de tais dados na aplicação. Também, em relação à mesma área de desenvolvimento, foi adotada a biblioteca Devise⁴² para que fosse permitida o registro e login de usuários de uma maneira segura por meio da criptografia por *default* das senhas.

Em relação ao *front-end*, foi utilizado o TailwindCSS⁴³ para permitir a rápida prototipação de elementos na tela com a utilização de classes nomeadas de acordo com a propriedade CSS a ser aplicada. Além disso, a *framework* DaisyUI⁴⁴ foi utilizada como auxiliar ao Tailwind, fornecendo vários componentes de interface prontos, como barras de navegação e botões. Também, para a renderização das artes, foram utilizadas as bibliotecas p5.js⁴⁵, responsável pela renderização das formas geométricas que compõem cada arte, e p5.js-svg⁴⁶, utilizado para a conversão das artes realizadas em vetores, permitindo uma exibição nítida na página de visualização de arte.

⁴²<https://github.com/heartcombo/devise>

⁴³<https://tailwindcss.com/>

⁴⁴<https://daisyui.com/>

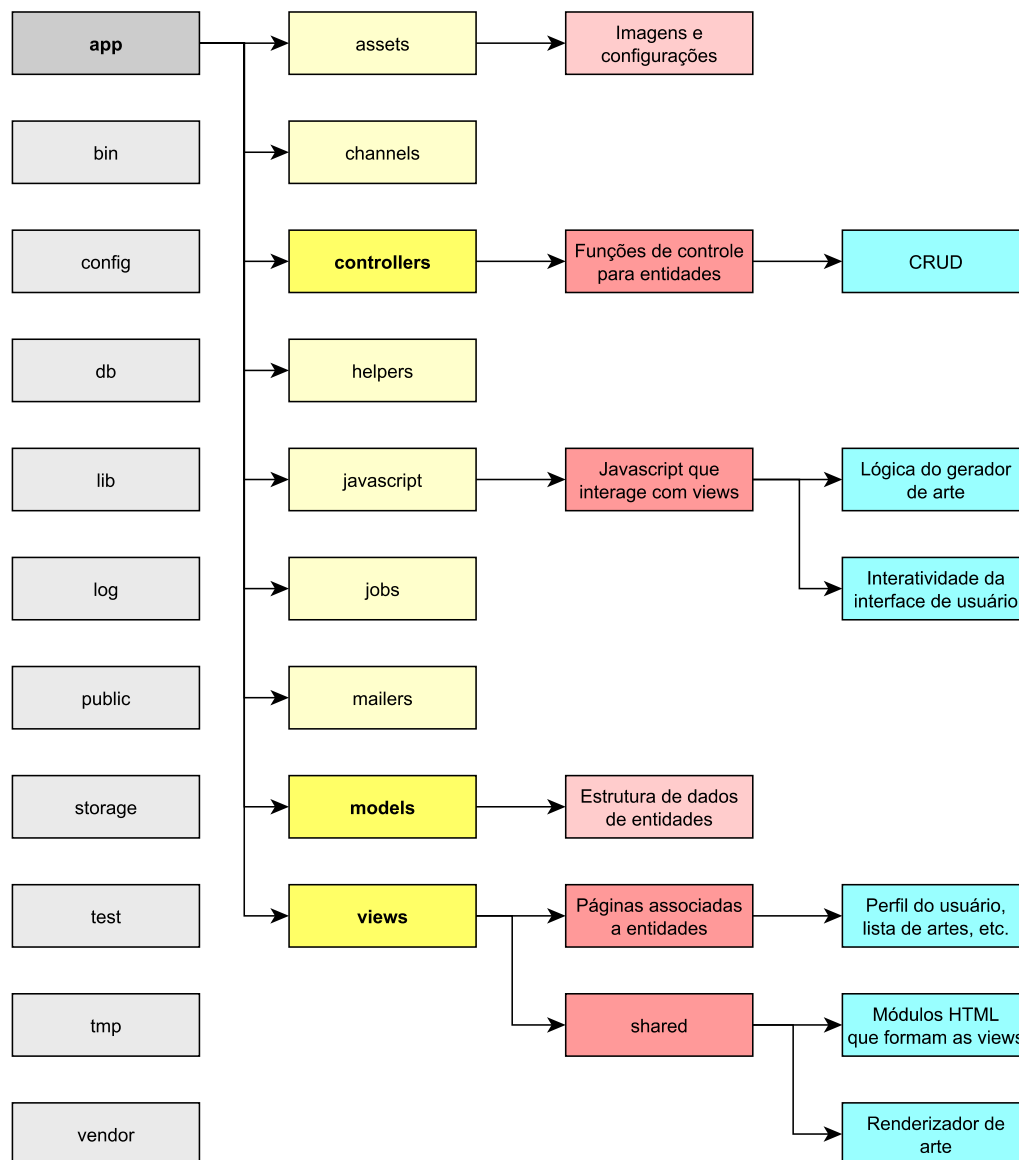
⁴⁵<https://p5js.org/>

⁴⁶<https://github.com/zenozeng/p5.js-svg>

5.4.2 Processo de desenvolvimento

O projeto seguiu a arquitetura MVC (Model-View-Controller) de acordo com a convenção adotada pela *framework* Ruby on Rails, sendo a estrutura final mostrada na figura 71.

Figura 71: Estrutura do projeto Faciliart



Fonte: Elaborado pelo autor

Inicialmente, para a construção das páginas relacionadas à criação, edição e exibição de artes, foi realizado um comando de rails `generate scaffold` para a geração automática de tais elementos. Com isto, realizou-se, primeiramente, a formulação da página de edição de arte e a implementação inicial da funcionalidade de renderização de arte em um canvas utilizando lógica em Javascript em conjunto com a biblioteca p5.js. Além disso, foi necessário realizar a concepção da estrutura de dados da arte em JSON, sendo exibido um exemplo abaixo, para que a peça pudesse ser armazenada em banco de dados com todos os detalhes necessários, como cor de *background* e o detalhamento de cada camada, incluindo suas coordenadas e o gerador selecionado.

```

1 {
2   "general": { "width": "800", "height": "600", "bg": "#B31616" },
3   "layers": [
4     {
5       "generator": "line",
6       "coordinates": [
7         { "x": 230, "y": 529 },
8         { "x": 301, "y": 523 },
9         { "x": 506, "y": 277 },
10        { "x": 439, "y": 122 },
11        { "x": 4, "y": 221 },
12        { "x": 359, "y": 262 },
13        { "x": 522, "y": 564 },
14        { "x": 496, "y": 274 },
15        { "x": 726, "y": 171 },
16        { "x": 541, "y": 361 },
17        { "x": 473, "y": 14 },
18        { "x": 176, "y": 382 }
19      ],
20      "parameters": {
21        "color": "#67C3D7",
22        "fill": true,
23        "points": "12",
24        "stroke": "1"
25      }
26    },
27    {
28      "generator": "polygon",
29      "coordinates": [
30        { "x": 47, "y": 14 },
31        { "x": 645, "y": 312 },
32        { "x": 359, "y": 160 },
33        { "x": 361, "y": 32 },
34        { "x": 298, "y": 300 },
35        { "x": 570, "y": 454 }
36      ],
37      "parameters": {
38        "color": "#1ED747",
39        "fill": true,
40        "points": "6",
41        "stroke": "1"
42      }
43    }
44  ]
45 }

```

Em paralelo, foi realizada a geração da entidade `User` e a integração inicial da biblioteca `Devise` ao projeto, permitindo registro e login de usuários, de maneira a permitir a implementação de que a arte criada fosse atribuída ao usuário logado.

Logo após possuir uma versão inicial do editor de artes, foi realizada uma primeira versão da página inicial, o qual foi necessário realizar uma lógica para a renderização dos *thumbnails* referentes a cada arte utilizando uma conversão do canvas do `p5.js` para o formato `png`, sendo possível realizar operações de centralização destas artes na área reservada a cada *thumbnail*.

Nesta etapa, foi necessário refatorar o renderizador para permitir que o mesmo pudesse criar várias instâncias do `p5.js` dependendo da quantidade de artes a serem exibidas, visto que antes a implementação de renderizador apenas contemplava uma arte por página.

Então, para que a possibilidade de listar itens disponíveis para compra, incluindo tamanhos associados, realizou-se a criação de mais duas entidades, intituladas de `Item`, que fornece apenas o nome para tal, e `ItemSize`, que está associado a um `Item`, de forma a indicar um tamanho e um preço para tal.

Com isto, foi iniciado o desenvolvimento da página de visualização da arte, podendo ser acessada pelo link fornecido por cada *thumbnail* e possuindo um layout parecido com o da edição de arte. A distinção entre esta página e a de edição é inerente às informações exibidas, como dados do autor da arte, possibilidade de compra, adição e visualização de comentários, além do método de renderização da arte que, ao invés de exibir um canvas rasterizado, as camadas pertencentes eram convertidas para `svg`, permitindo uma visualização nítida da mesma. Isto necessitou que fosse realizado um incremento no código do renderizador, precisando que fosse possível a passagem dos dados em `svg` para um elemento `HTML` específico de uma forma que ficasse centralizado e escalonado corretamente em relação ao elemento `HTML` pai.

Com o progresso na implementação, foram realizadas melhorias nas páginas web, redefinindo as cores padrão e aplicando um *branding* ao projeto. Mais detalhes, ver seção 5.3.2.

Logo, avançando com o desenvolvimento do projeto, foi realizada a implementação da página de usuário, permitindo com que houvesse a listagem de informações apenas associadas àquele utilizador específico, como artes, comentários e pedidos realizados. Com a posterior concepção da entidade `Setting`, que possuía, como dado, apenas a porcentagem de ganho do usuário por pedido realizado de alguma arte sua, foi possível associar os ganhos totais à página de usuário, sendo exibido somente ao mesmo quando logado.

Paralelamente, precisou ser realizada uma correção no renderizador para tornar-se compatível com a *framework* Turbo⁴⁷, que otimiza o carregamento de páginas por reaproveitar os recursos já existentes na página anterior, pois a implementação, até então, estava duplicando ou ocultando a exibição de artes quando a biblioteca estava ativada para as páginas, existindo tal regressão por conta da ausência de carregamento total da página, e sim apenas um reaproveitamento dos recursos já existentes ao acessar a próxima página. Esta correção foi de extrema importância por conta das mensagens de informação, aviso e erro serem apenas exibidas com a presença do Turbo. Esta melhoria no renderizador foi considerada uma das etapas mais difíceis do projeto.

Depois, realizou-se ajustes finos de interface e resolução de *bugs* de menor prioridade, resultando em um projeto com qualidade de Mínimo Produto Viável (MVP), sendo, então, enviado a um servidor na plataforma Heroku, que hospeda projetos de linguagens de programação diversas, incluindo em Ruby. Com a aplicação funcionando na web, foi realizada a compra de um domínio no Hostinger com o nome `facili.art` e, depois, alterou-se os endereços dos servidores de nome para a Cloudflare, que permitiu um melhor gerenciamento de acesso à aplicação, com prevenção à ataques DDOS (*Distributed Denial of Service*) e recursos aprimorados de redirecionamento, possibilitando que fosse viável o acesso utilizando apenas o endereço `facili.art` ao invés

⁴⁷<https://github.com/hotwired/turbo-rails>

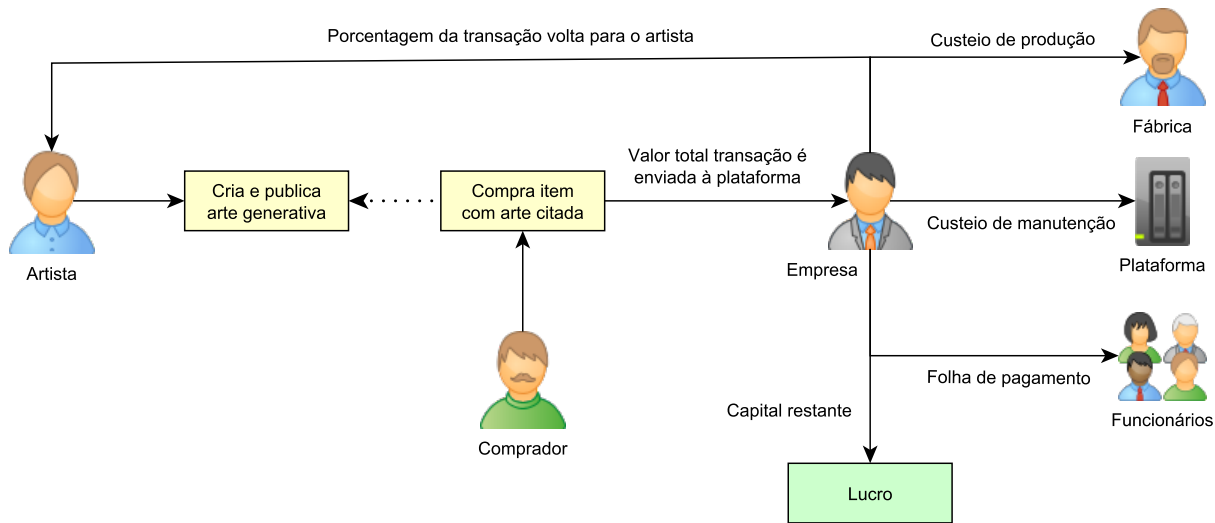
de www.facili.art, evitando que o usuário necessitasse inserir o subdomínio www.

5.5 Modelo de negócio

O modelo de negócio da Faciliart (figura 72) é similar a alguns *marketplaces* voltados para os artistas. Desta forma, é planejado que o artista receba ganhos considerando os pedidos efetuados por usuários que desejam um item que possua a obra do artista. Logo, ao efetuar um pedido, o artista recebe uma porcentagem, sendo tal percentil configurado pelos administradores, do valor do artigo a ser adquirido.

Em relação ao valor total da operação de venda, são consideradas tais variáveis em uma transação: ganho do artista baseado na porcentagem configurada, custo para a produção do artigo e a quantia destinada ao caixa da plataforma, que será destinada à manutenção do sistema, pagamento de funcionários e lucro.

Figura 72: Modelo de negócio da plataforma Faciliart



Fonte: Elaborado pelo autor

Logo, é possível representar o funcionamento do negócio matematicamente utilizando as expressões abaixo:

$$\text{Valor do produto} = \{x \in \mathbb{Q} : x > 0\}$$

$$\text{Custo do produto} = \{y \in \mathbb{Q} : y > 0\}$$

$$\text{Porcentagem de ganho} = \{z \in \mathbb{Q} : 0 < z < 100\}$$

$$\text{Ganho para usuario} = \frac{xz}{100}$$

$$\text{Ganhos totais para usuario} = \sum_{i=1}^n \frac{x_i z_i}{100}$$

$$\text{Ganhos para a plataforma} = x - y - \frac{xz}{100}$$

$$\text{Ganhos totais para a plataforma} = \sum_{i=1}^n x_i - y_i - \frac{x_i z_i}{100}$$

5.5.1 SWOT

Abaixo (figura 73), é mostrada uma matriz SWOT, possuindo o objetivo de identificar fatores internos ou externos que possam impactar positivamente ou negativamente na proposta de negócio.

Figura 73: Matriz SWOT da Faciliart

	Fatores úteis	Fatores prejudiciais
Fatores internos	Forças: • Desenvolvimento ágil • Produto com interface satisfatória • Estrutura organizacional simplificada • Terceirização da produção	Fraquezas: • Marketing deve melhorar • Possível débito técnico por conta da complexidade do componente de renderização
Fatores externos	Oportunidades: • Proposta inovadora em relação às alternativas • Com a evolução da IA, há possibilidade de integração de tal tecnologia ao produto	Ameaças: • Existência de muitas plataformas de arte • Complexidade ao empreender no Brasil

Fonte: Elaborado pelo autor

5.5.2 Canvas

Abaixo (figura 74), é mostrada um *Business Model Canvas*, possuindo o objetivo de destacar as inovações que o negócio propõe ao mercado, assim como mostrar a origem das receitas e despesas.

Figura 74: Business Model Canvas da Faciliart

Parceiros chave • Empresa de impressão de quadros • Empresa de fabricação de blusas estampadas • Transportadoras • Serviço de pagamentos online • Serviço de registro de domínio (registrar) • Plataforma Heroku, responsável pela hospedagem da aplicação na Web	Atividades chave • Fabricação terceirizada de quadros e blusas com estampa contendo arte generativa da plataforma • Pagamento automatizado • Hospedagem da aplicação na Web • Fretamento das encomendas dos clientes	Proposta de valor • Criação de arte generativa utilizando uma interface gráfica com padrões predefinidos disponíveis, dispensando que o usuário programe • Rede social de arte generativa por meio do armazenamento das mesmas em cada perfil do usuário, permitindo que outros usuários realizem comentários • Possibilidade de compra de peças com arte generativa utilizando uma interface intuitiva	Relação com o cliente • Vendas de quadros ou blusas pela plataforma • Divulgação de campanhas publicitárias em redes sociais • Suporte ao cliente • Contato profissional	Segmentos de mercado • Artistas, podendo receber uma porcentagem do valor das vendas realizadas em plataforma • Pessoas físicas em geral que estejam interessadas em comprar itens contendo arte generativa encontrada na plataforma. Não há planejamento para implementação de B2B até então
Recursos chave • Propriedade intelectual oriunda da plataforma como um todo, contendo a geração e armazenagem de artes no banco de dados, assim como o registro de pedidos			Canais • Plataforma web hospedada em facili.art • Redes sociais, como Instagram, Twitter e Facebook • E-mail	
Estrutura de custos • Renovação anual de domínio • Hospedagem (ex.: Heroku, AWS) • Manutenção de plataforma (implementação de <i>features</i> e correção de <i>bugs</i>)	• Folha de pagamento de funcionários • Despesas com água, luz e internet • Logística, com despesas de contratação de fretamento	Fontes de renda • Venda de artigos com arte generativa		

Fonte: Elaborado pelo autor

5.6 Pesquisa de usabilidade

5.6.1 Metodologia

A metodologia utilizada para a pesquisa de usabilidade é denominada UEQ (*User Experience Questionnaire*), cuja versão inicial foi desenvolvida em 2005 na Alemanha (SCHREPP, 2023, p. 1), a qual avalia fatores como atratividade, perspicuidade, eficiência, confiabilidade, estimulação e inovação, de maneira a mensurar as qualidades pragmáticas e hedônicas da experiência de usuário associado a um produto (SCHREPP, 2023, p. 2–3). O detalhamento dos fatores encontra-se na tabela abaixo:

Tabela 6: Fatores avaliados no UEQ

Fator	Descrição
Atratividade	Mensura se um usuário gostou ou não do produto.
Perspicidade	Avalia a facilidade do usuário ao utilizar o produto.
Eficiência	Determina se o usuário consegue realizar as operações disponibilizadas pelo produto de maneira rápida.
Confiabilidade	Verifica se o produto atende às expectativas do usuário.
Estimulação	Afere se o produto motiva um usuário a utilizá-lo.
Inovação	Avalia o grau de inovação de um produto no ponto de vista do usuário.

Fonte: SCHREPP, 2023, p. 2

Desta forma, são realizadas perguntas objetivas que avaliam, em uma escala de 1 a 7, sendo 1 associado à completa discordância e 7 à completa concordância (SCHREPP, 2023, p. 14), sendo posteriormente realizada, uma média aritmética, dividida em categorias em relação aos fatores associados, sobre todas as avaliações coletadas, o que permite uma análise geral do produto (SCHREPP, 2023, p. 5–6).

Logo, o formulário completo consiste nas perguntas abaixo (figura 75):

Figura 75: Questionário UEQ em português

	1	2	3	4	5	6	7	
Desagradável	☐	☐	☐	☐	☐	☐	☐	Agradável
Incompreensível	☐	☐	☐	☐	☐	☐	☐	Compreensível
Criativo	☐	☐	☐	☐	☐	☐	☐	Sem criatividade
De fácil aprendizagem	☐	☐	☐	☐	☐	☐	☐	De difícil aprendizagem
Valioso	☐	☐	☐	☐	☐	☐	☐	Sem valor
Aborrecido	☐	☐	☐	☐	☐	☐	☐	Excitante
Desinteressante	☐	☐	☐	☐	☐	☐	☐	Interessante
Imprevisível	☐	☐	☐	☐	☐	☐	☐	Previsível
Rápido	☐	☐	☐	☐	☐	☐	☐	Lento
Original	☐	☐	☐	☐	☐	☐	☐	Convencional
Obstrutivo	☐	☐	☐	☐	☐	☐	☐	Condutor
Bom	☐	☐	☐	☐	☐	☐	☐	Mau
Complicado	☐	☐	☐	☐	☐	☐	☐	Fácil
Desinteressante	☐	☐	☐	☐	☐	☐	☐	Atrativo
Comum	☐	☐	☐	☐	☐	☐	☐	Vanguardista
Incômodo	☐	☐	☐	☐	☐	☐	☐	Cômodo
Seguro	☐	☐	☐	☐	☐	☐	☐	Inseguro
Motivante	☐	☐	☐	☐	☐	☐	☐	Desmotivante
Atende às expectativas	☐	☐	☐	☐	☐	☐	☐	Não atende às expectativas
Ineficiente	☐	☐	☐	☐	☐	☐	☐	Eficiente
Evidente	☐	☐	☐	☐	☐	☐	☐	Confuso
Impraticável	☐	☐	☐	☐	☐	☐	☐	Prático
Organizado	☐	☐	☐	☐	☐	☐	☐	Desorganizado
Atraente	☐	☐	☐	☐	☐	☐	☐	Feio
Simpático	☐	☐	☐	☐	☐	☐	☐	Antipático
Conservador	☐	☐	☐	☐	☐	☐	☐	Inovador

Fonte: Formulário disponível em https://www.ueq-online.org/Material/UEQ_All_Languages.zip. Acesso em 19 de Outubro de 2023

Além disso, os autores criaram uma versão do questionário com menos perguntas denominado UEQ-S (figura 76). Embora isso possa prejudicar a precisão dos resultados, esta variação do questionário possui o objetivo de ampliar a coleta de respostas da por meio da redução no tempo de preenchimento (SCHREPP, 2023, p. 11–13).

Figura 76: Questionário UEQ-S em português

	1	2	3	4	5	6	7	
Obstrutivo	☐	☐	☐	☐	☐	☐	☐	Condutor
Complicado	☐	☐	☐	☐	☐	☐	☐	Fácil
Ineficiente	☐	☐	☐	☐	☐	☐	☐	Eficiente
Confuso	☐	☐	☐	☐	☐	☐	☐	Evidente
Aborrecido	☐	☐	☐	☐	☐	☐	☐	Excitante
Desinteressante	☐	☐	☐	☐	☐	☐	☐	Interessante
Convencional	☐	☐	☐	☐	☐	☐	☐	Original
Comum	☐	☐	☐	☐	☐	☐	☐	Vanguardista

Fonte: Formulário disponível em https://www.ueq-online.org/Material/UEQS_Items.pdf. Acesso em 19 de Outubro de 2023

No entanto, o presente trabalho não aplicou a pesquisa utilizando o modelo padrão do UEQ, mas sim uma adaptação do mesmo com a aplicação, por seção, de seis perguntas objetivas com a mesma escala de 1 a 7, na qual cada questão corresponde a um fator avaliado na UEQ, além de uma pergunta subjetiva na qual o usuário pode fornecer sugestões para tal seção. Esta medida foi realizada com o objetivo de haver a liberdade de redação de perguntas mais sucintas, além de simplificar ainda mais o questionário.

Tabela 7: Questionário de usabilidade elaborado com base nos fatores do UEQ

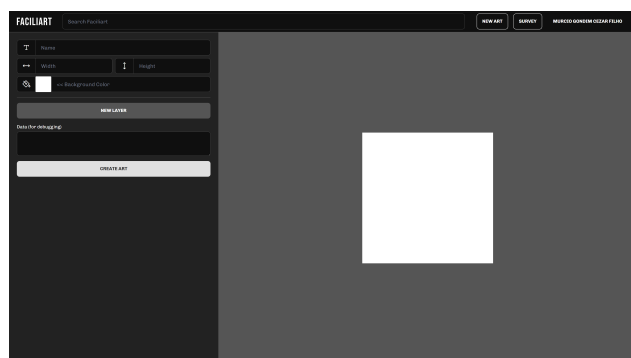
Pergunta	Tipo de pergunta	Fator analisado
Você gostou da experiência de utilização promovida pela página?	Objetiva	Atratividade
Esta página foi fácil de ser utilizada?	Objetiva	Perspicidade
Você utilizou esta página de maneira eficiente?	Objetiva	Eficiência
Esta página funcionou da maneira que você esperava?	Objetiva	Confiabilidade
Esta página lhe motivou a utilizar esta plataforma?	Objetiva	Estimulação
Esta página se apresentou inovadora em relação às alternativas?	Objetiva	Inovação
Você gostaria de sugerir melhorias para a página?	Subjetiva e opcional	—

Fonte: Elaborado pelo autor

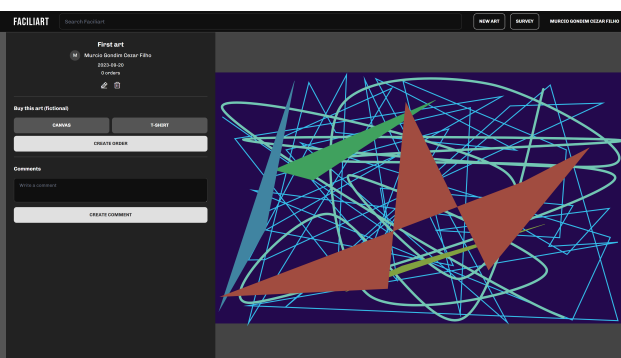
Logo, o questionário buscou estudar a reação do usuário em páginas diferentes da plataforma (figura 77), sendo aplicado o conjunto de perguntas já citado para cada tela, permitindo uma maior granularidade da pesquisa, sendo possível descobrir, com maior facilidade, os módulos da plataforma com algum tipo de problema de usabilidade.

Figura 77: Páginas analisadas na pesquisa

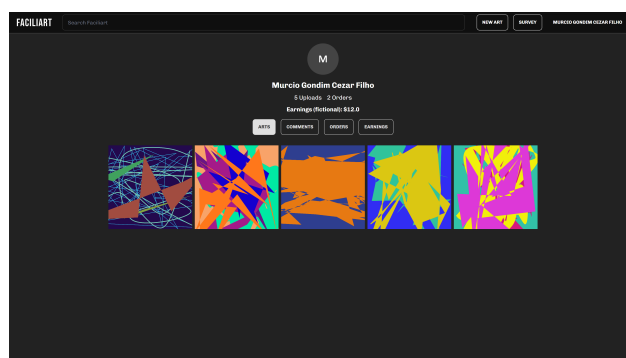
(a) Criação de arte



(b) Visualização de arte



(c) Perfil do usuário



Fonte: Elaborado pelo autor

5.6.2 Primeira pesquisa

A primeira pesquisa de usabilidade contou com a participação de **8 pessoas** entre os dias 25 de Setembro de 2023 e 19 de Outubro de 2023.

Tabela 8: Criação de arte — Resultados da pesquisa de usabilidade

Fator	Média	Mediana	Moda	Desvio padrão	Interpretação
Atratividade	6,25	7	7	1,75	Concordo
Perspicuidade	6	6,5	7	1,69	Concordo
Eficiência	5,5	6	6	1,93	Concordo parcialmente
Confiabilidade	6	7	7	1,77	Concordo
Estimulação	5,5	6	6	1,93	Concordo parcialmente
Inovação	5,25	6,5	7	2,31	Concordo parcialmente

Fonte: Elaborado pelo autor

- Sugestões:
 - Melhorar contraste de seção de camadas
 - Mudar título do botão de “Create Art” para “Save Art”
 - Adicionar controle de opacidade
 - Exibir elemento gerado antes de ser salvo para camada
 - Maior variedade de geradores

Tabela 9: Visualização de arte — Resultados da pesquisa de usabilidade

Fator	Média	Mediana	Moda	Desvio padrão	Interpretação
Atratividade	5,75	7	7	2,19	Concordo parcialmente
Perspicuidade	6	7	7	2,14	Concordo
Eficiência	5,5	7	7	2,33	Concordo parcialmente
Confiabilidade	5,88	7	7	2,23	Concordo parcialmente
Estimulação	5,5	6,5	7	2,14	Concordo parcialmente
Inovação	5,13	5,5	7	2,1	Concordo parcialmente

Fonte: Elaborado pelo autor

- Sugestões:
 - Botão de compartilhamento para redes sociais (ex.: Instagram, X, Facebook)

Tabela 10: Perfil de usuário — Resultados da pesquisa de usabilidade

Fator	Média	Mediana	Moda	Desvio padrão	Interpretação
Atratividade	6,38	7	7	1,19	Concordo
Perspicuidade	6,5	7	7	1,07	Concordo
Eficiência	6,38	7	7	1,06	Concordo
Confiabilidade	6,25	7	7	1,16	Concordo
Estimulação	6,25	7	7	1,16	Concordo
Inovação	5,75	6	7	1,28	Concordo parcialmente

Fonte: Elaborado pelo autor

- Sugestões:
 - Adicionar buscador de pedidos

5.6.3 Segunda pesquisa — Pós-ajustes

A segunda pesquisa de usabilidade contou com a participação de **10 pessoas** entre os dias 09 de Novembro de 2023 e 27 de Novembro de 2023.

- Ajustes realizados:
 - Melhoria de contraste de seção de camadas (Criação de arte)
 - Mudança de título do botão de “Create Art” para “Save Art” (Criação de arte)
 - Adição de controle de opacidade (Criação de arte)
 - Exibição elemento gerado antes de ser salvo para camada (Criação de arte)
 - Adição de mais geradores (Criação de arte)
 - Adição de descrição de gerador de arte no momento em que os parâmetros do mesmo são exibidos (Criação de arte)

Tabela 11: Criação de arte — Resultados da pesquisa de usabilidade — Pós-ajuste

Fator	Média	Mediana	Moda	Desvio padrão	Interpretação
Atratividade	6,4	6,5	7	0,7	Concordo
Perspicuidade	6,5	7	7	0,71	Concordo
Eficiência	6,6	7	7	0,7	Concordo
Confiabilidade	6,6	7	7	0,7	Concordo
Estimulação	5,9	6,5	7	1,37	Concordo
Inovação	6,1	7	7	1,73	Concordo parcialmente

Fonte: Elaborado pelo autor

Tabela 12: Visualização de arte — Resultados da pesquisa de usabilidade — Pós-ajuste

Fator	Média	Mediana	Moda	Desvio padrão	Interpretação
Atratividade	6,3	7	7	1,34	Concordo
Perspicuidade	6,6	7	7	0,7	Concordo
Eficiência	6,6	7	7	0,7	Concordo
Confiabilidade	6,3	6,5	7	0,82	Concordo
Estimulação	6	6,5	7	1,33	Concordo
Inovação	6,4	7	7	1,26	Concordo

Fonte: Elaborado pelo autor

Tabela 13: Perfil de usuário — Resultados da pesquisa de usabilidade — Pós-ajuste

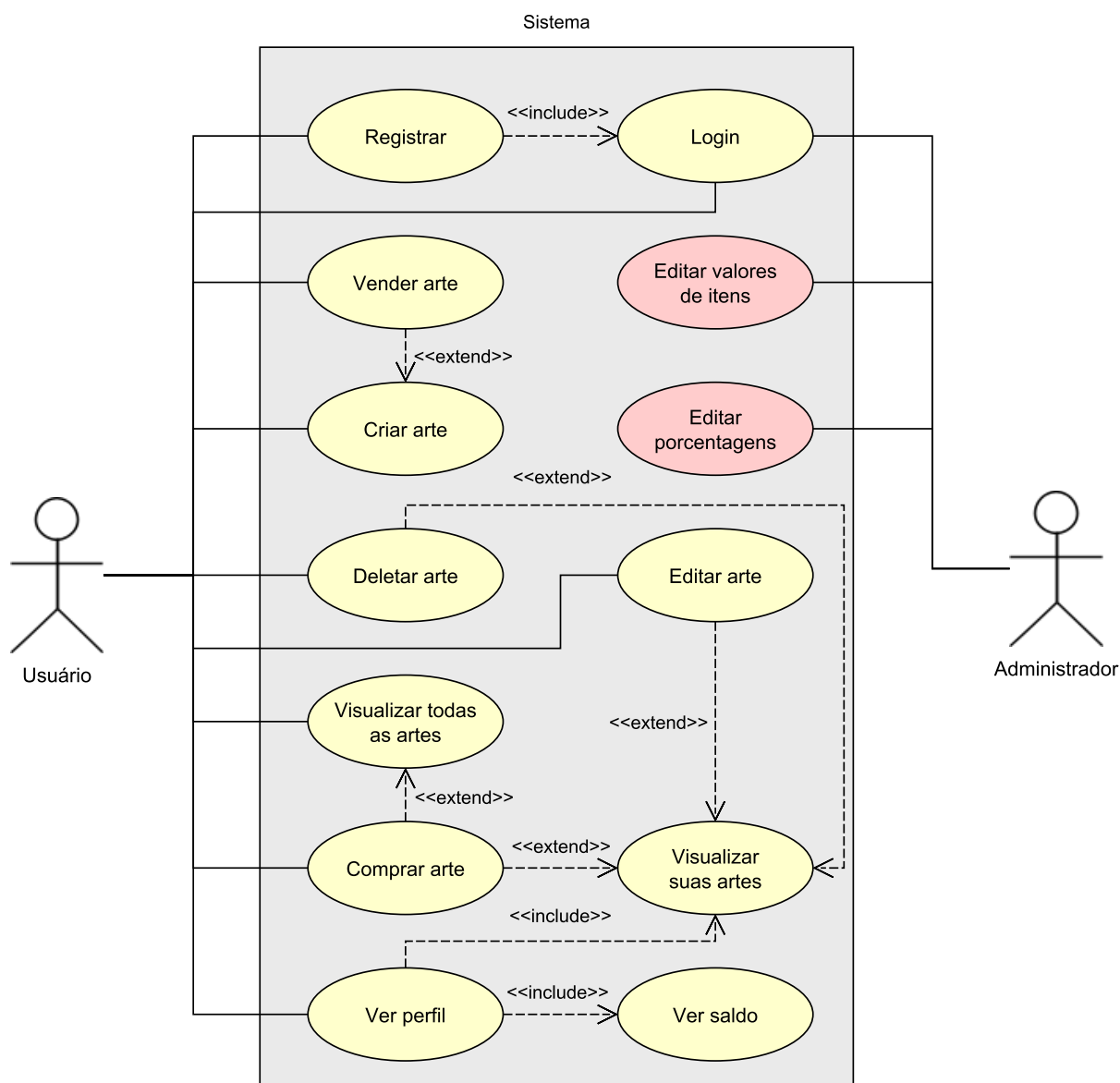
Fator	Média	Mediana	Moda	Desvio padrão	Interpretação
Atratividade	6,8	7	7	0,42	Concordo
Perspicuidade	6,5	7	7	0,71	Concordo
Eficiência	6,6	7	7	0,52	Concordo
Confiabilidade	6,5	6,5	7	0,53	Concordo
Estimulação	6,5	7	7	0,71	Concordo
Inovação	6,3	7	7	1,34	Concordo

Fonte: Elaborado pelo autor

5.7 Diagramas

5.7.1 UML — Casos de uso

Figura 78: Diagrama de casos de uso em UML

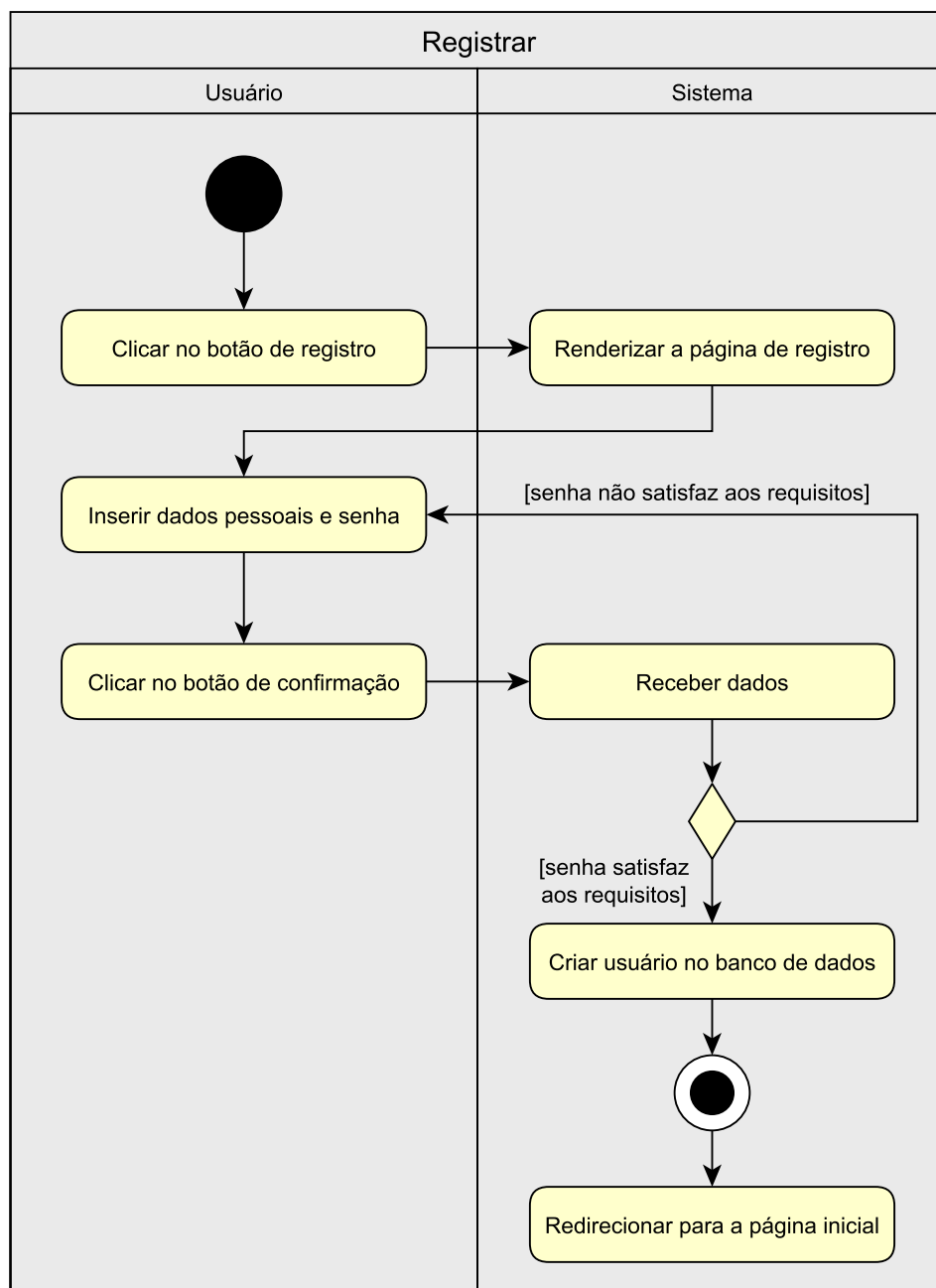


Fonte: Elaborado pelo autor

5.7.2 UML — Atividades

5.7.2.1 Registrar

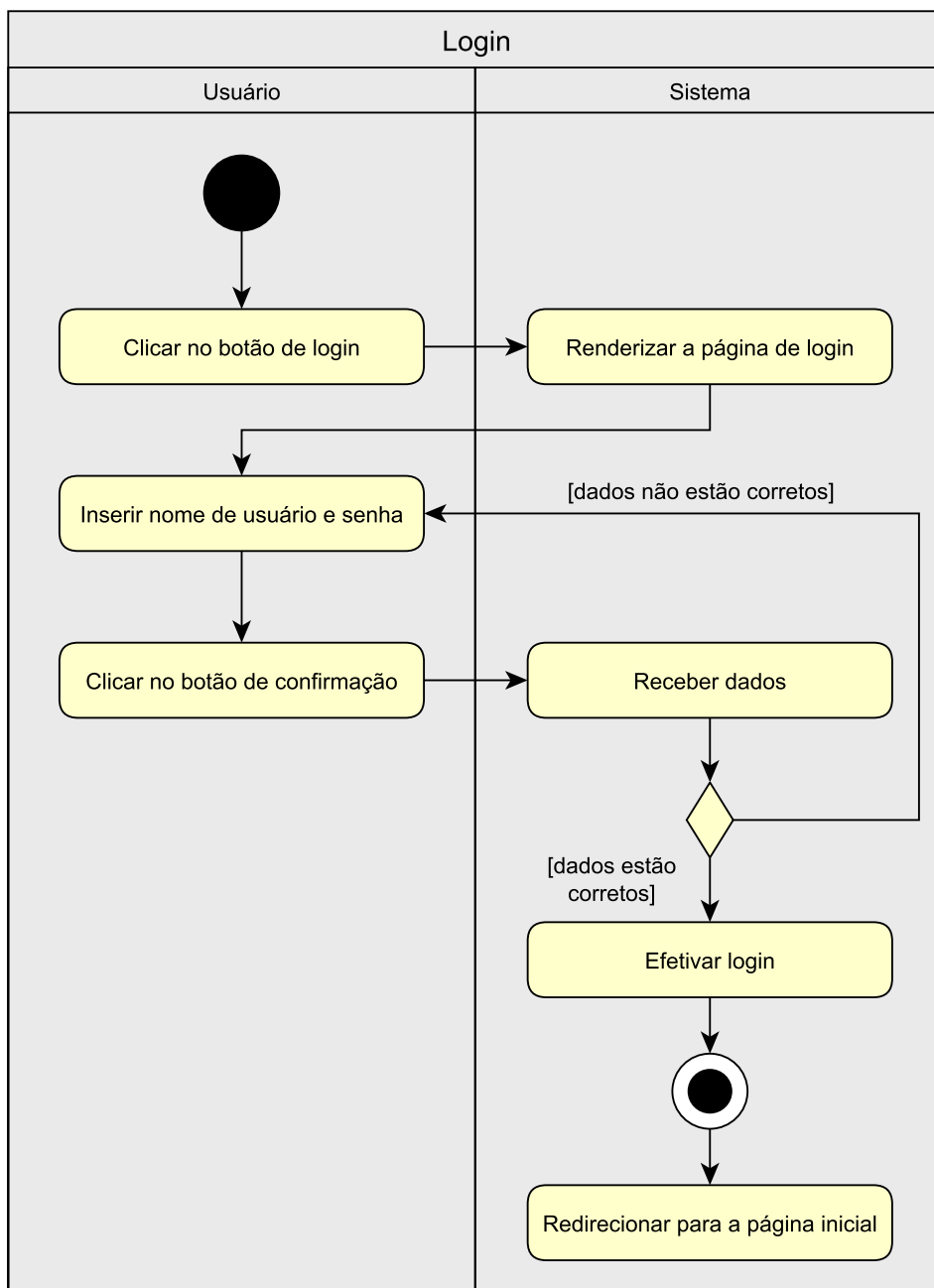
Figura 79: Diagrama de atividade de registro de usuário em UML



Fonte: Elaborado pelo autor

5.7.2.2 Login

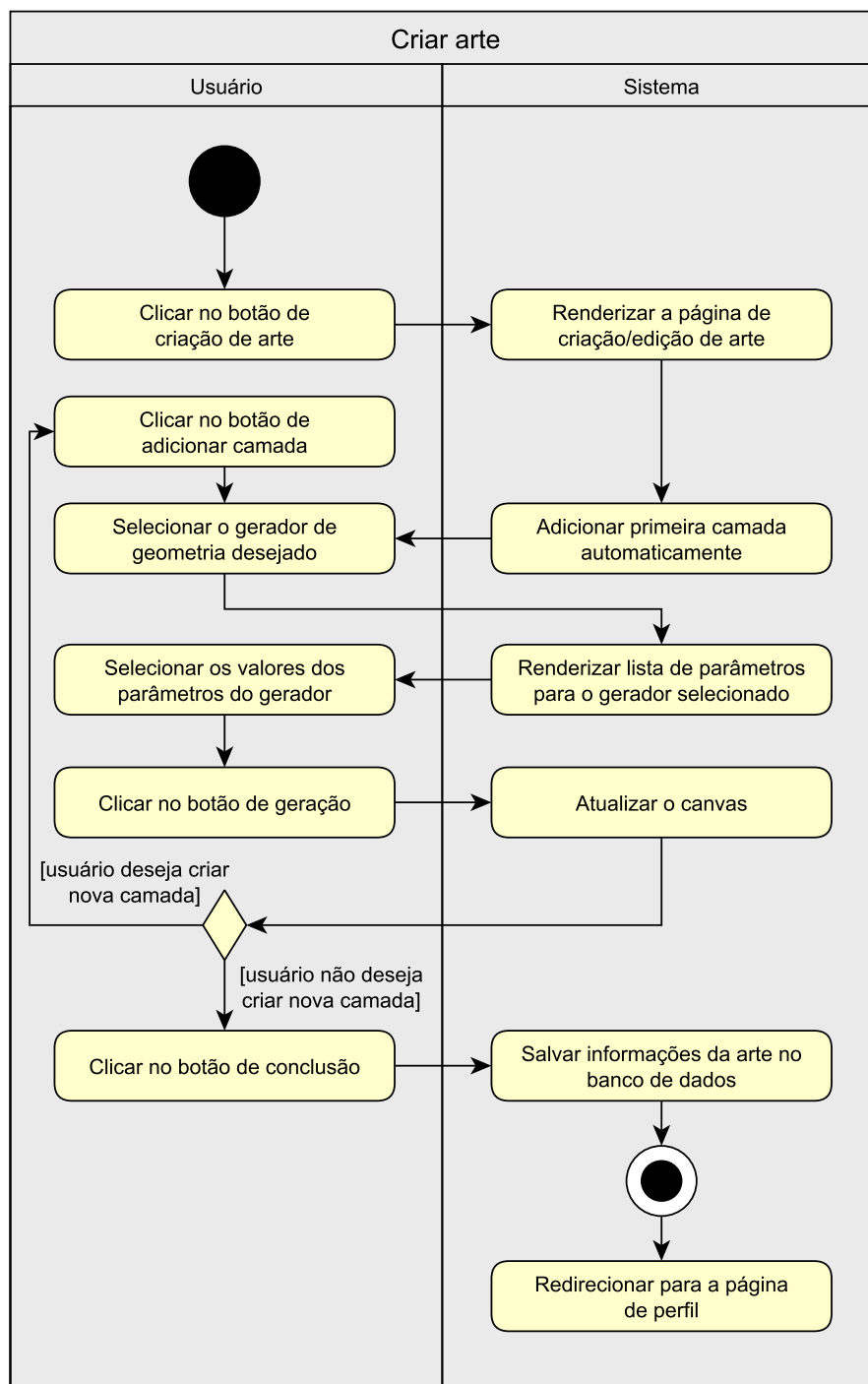
Figura 80: Diagrama de atividade de login de usuário em UML



Fonte: Elaborado pelo autor

5.7.2.3 Criar arte

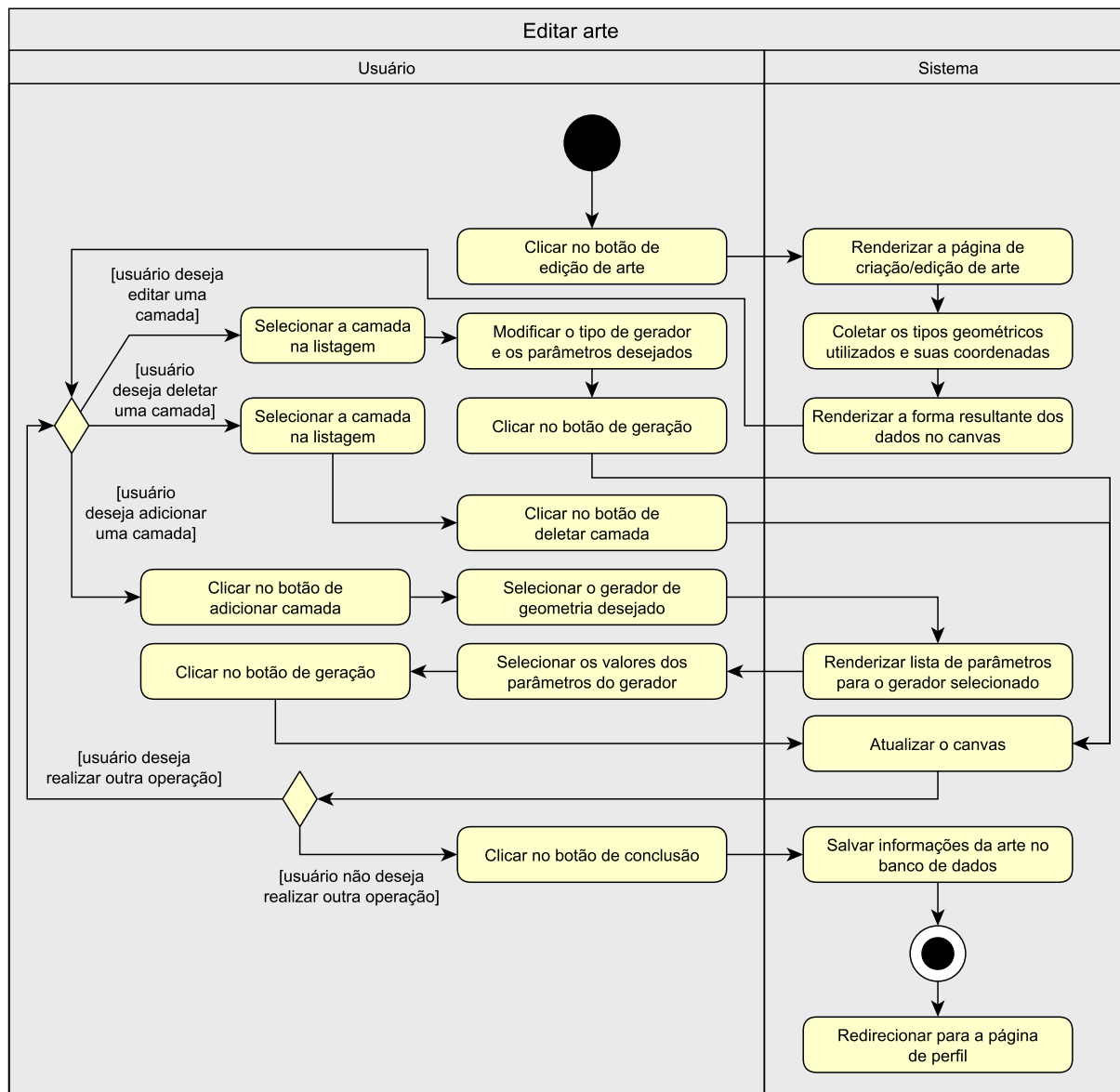
Figura 81: Diagrama de atividade de criação de arte em UML



Fonte: Elaborado pelo autor

5.7.2.4 Editar arte

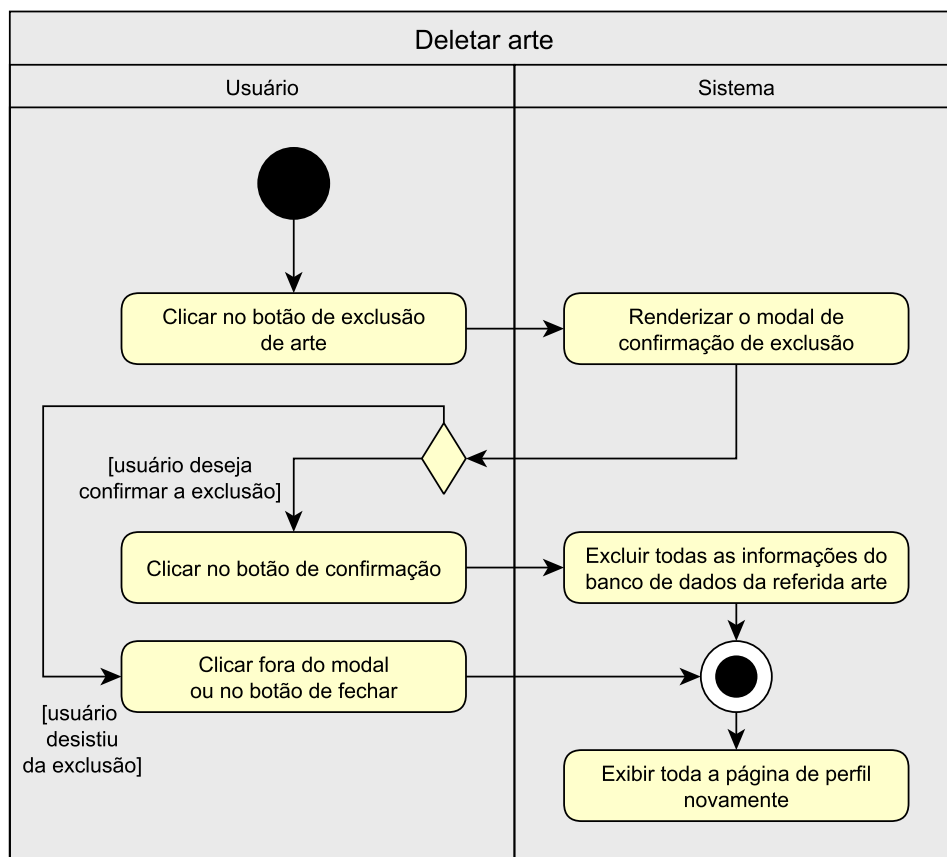
Figura 82: Diagrama de atividade de edição de arte em UML



Fonte: Elaborado pelo autor

5.7.2.5 Deletar arte

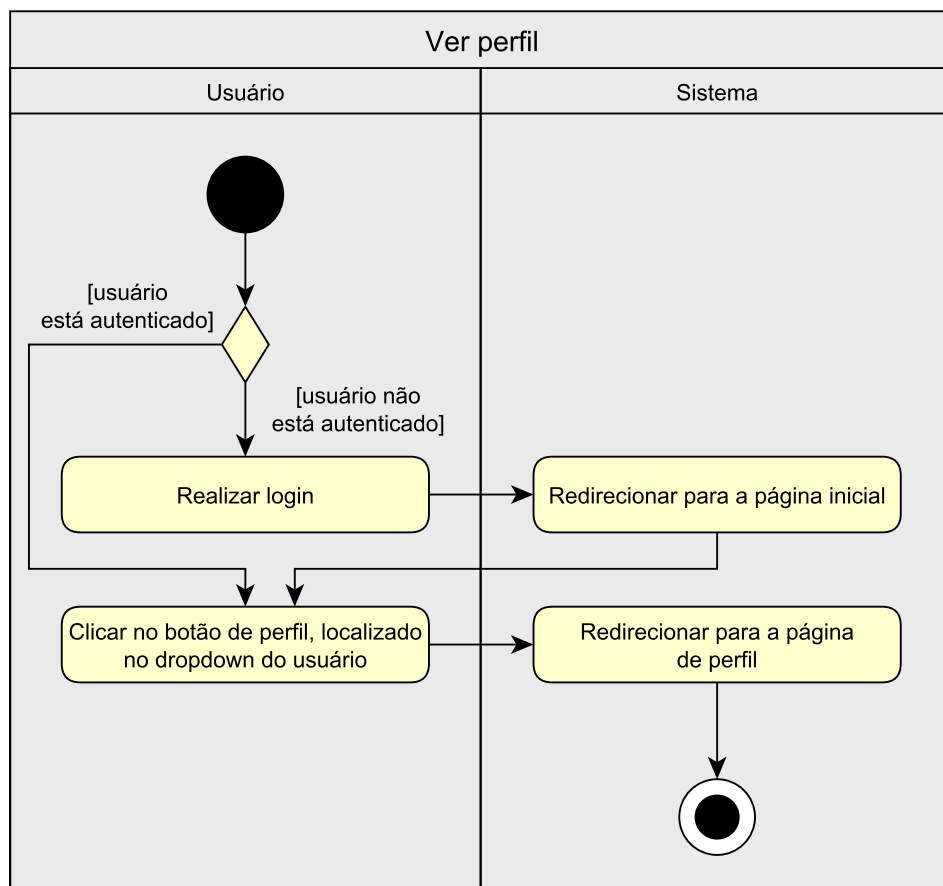
Figura 83: Diagrama de atividade de exclusão de arte em UML



Fonte: Elaborado pelo autor

5.7.2.6 Ver perfil

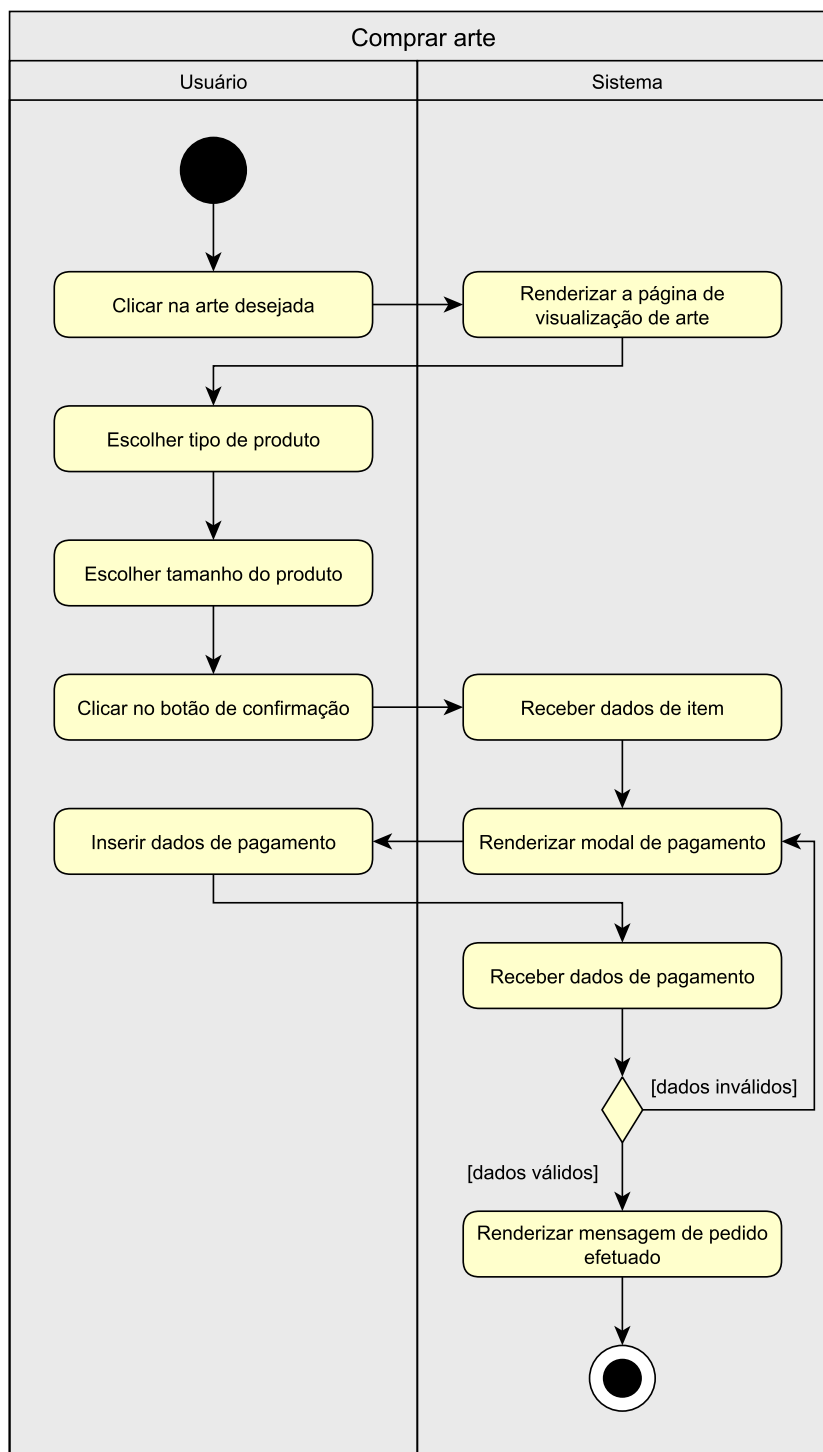
Figura 84: Diagrama de atividade de visualização de perfil em UML



Fonte: Elaborado pelo autor

5.7.2.7 Comprar arte

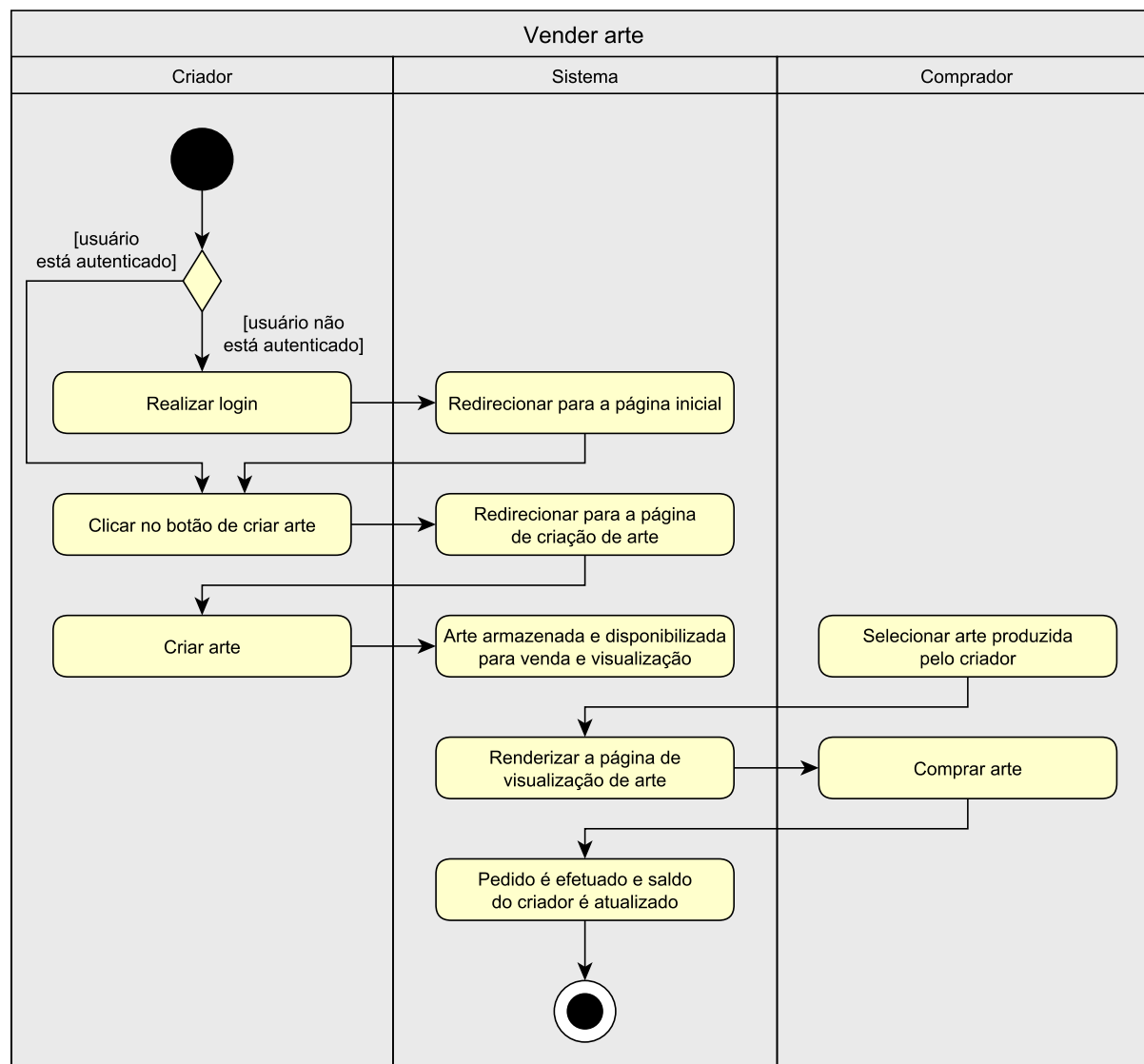
Figura 85: Diagrama de atividade de compra de arte em UML



Fonte: Elaborado pelo autor

5.7.2.8 Vender arte

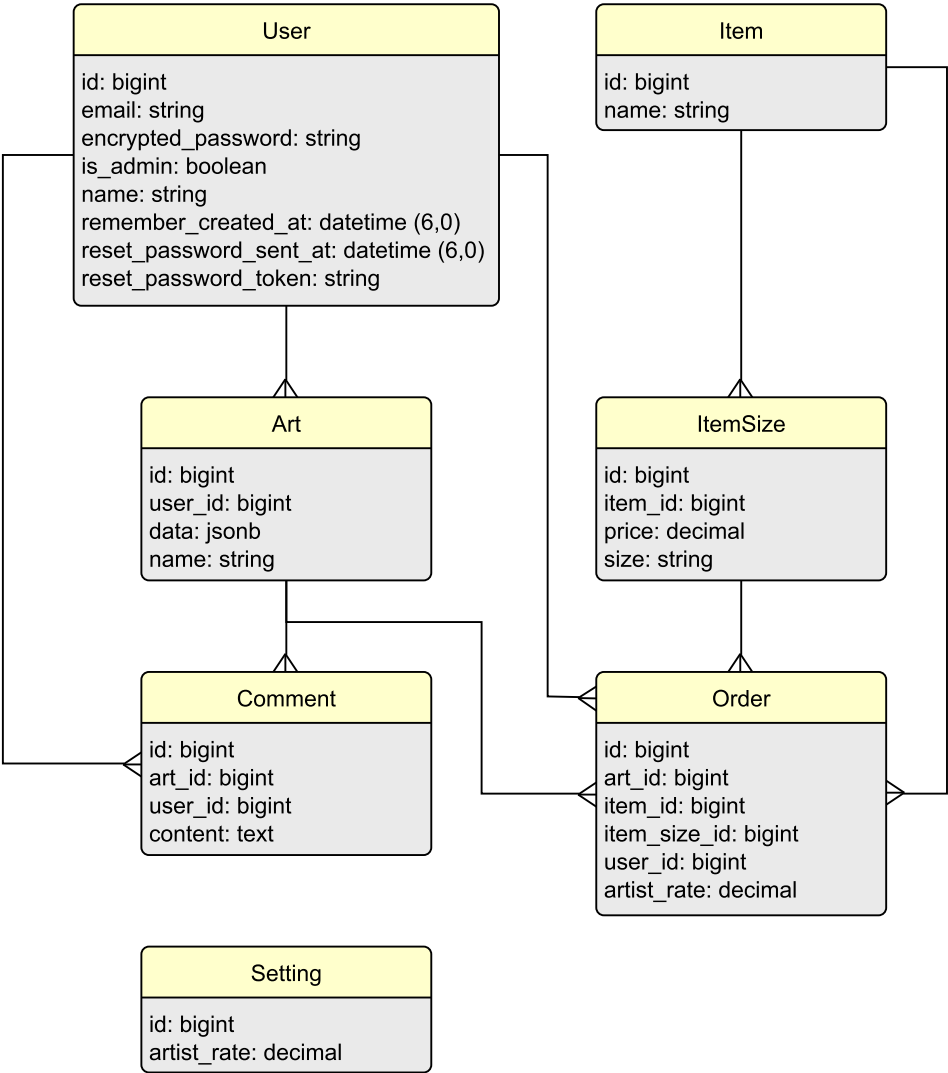
Figura 86: Diagrama de atividade de venda de arte em UML



Fonte: Elaborado pelo autor

5.7.3 Diagrama de Entidade e Relacionamento

Figura 87: Diagrama de Entidade e Relacionamento da Faciliart



Fonte: Elaborado pelo autor

Tabela 14: Entidade User

Atributo	Tipo	Função
id	bigint	Chave primária
email	string	Endereço de e-mail da conta do usuário
encrypted_password	string	Senha criptografada da conta do usuário
is_admin	boolean	Define se o usuário é administrador
name	string	Nome do usuário

Atributo	Tipo	Função
<code>remember_created_at</code>	<code>datetime (6,0)</code>	Data de criação da conta do usuário
<code>reset_password_sent_at</code>	<code>datetime (6,0)</code>	Data de solicitação de redefinição de senha (não implementado)
<code>reset_password_token</code>	<code>datetime (6,0)</code>	Token de autenticação de redefinição de senha (não implementado)

Fonte: Elaborado pelo autor

Tabela 15: Entidade Art

Atributo	Tipo	Função
<code>id</code>	<code>bigint</code>	Chave primária
<code>user_id</code>	<code>bigint</code>	Chave estrangeira para referência de usuário, servindo como associação de autoria
<code>data</code>	<code>jsonb</code>	Dados referentes à construção da arte generativa, possuindo as dimensões, tipos de geradores e coordenadas
<code>name</code>	<code>string</code>	Título da arte

Fonte: Elaborado pelo autor

Tabela 16: Entidade Comment

Atributo	Tipo	Função
<code>id</code>	<code>bigint</code>	Chave primária
<code>art_id</code>	<code>bigint</code>	Chave estrangeira para referência de arte
<code>user_id</code>	<code>bigint</code>	Chave estrangeira para referência de usuário, servindo como associação de autoria
<code>content</code>	<code>text</code>	Conteúdo do comentário

Fonte: Elaborado pelo autor

Tabela 17: Entidade Item

Atributo	Tipo	Função
<code>id</code>	<code>bigint</code>	Chave primária
<code>name</code>	<code>string</code>	Nome do item, sendo “Blusa” um exemplo

Fonte: Elaborado pelo autor

Tabela 18: Entidade ItemSize

Atributo	Tipo	Função
id	bigint	Chave primária
item_id	bigint	Chave estrangeira para referência de item
price	decimal	Preço do item de determinado tamanho
size	string	Tamanho do item, sendo “P”, “M” e “G” exemplos de tamanho para o item “Blusa”

Fonte: Elaborado pelo autor

Tabela 19: Entidade Order

Atributo	Tipo	Função
id	bigint	Chave primária
art_id	bigint	Chave estrangeira para referência de arte
item_id	bigint	Chave estrangeira para referência de item
item_size_id	bigint	Chave estrangeira para referência de tamanho de item, permitindo definir o valor da compra
user_id	bigint	Chave estrangeira para referência de usuário, servindo para identificar o comprador
artist_rate	decimal	Porcentagem de ganho de artista no pedido. Copiado do valor atual localizado em Setting

Fonte: Elaborado pelo autor

Tabela 20: Entidade Setting

Atributo	Tipo	Função
id	bigint	Chave primária
artist_rate	decimal	Porcentagem de ganho de artista em cada pedido

Fonte: Elaborado pelo autor

6 CONCLUSÃO

A referida pesquisa, em relação à análise de similares, concluiu-se que existe uma ausência de funcionalidades por parte das ferramentas de arte generativa por interface gráfica citadas na seção 4.1.1, com cada aplicação permitindo, na maioria das vezes, apenas um tipo de geração. Além disso, é possível evidenciar, nestes projetos atuais, o número reduzido de parâmetros disponíveis para controlar o que pode ser gerado. Logo, como foi explanado na contextualização (seção 1.1), a falta de recursos disponibilizados entre estes geradores pode constituir um problema de usabilidade por conta do usuário ter que recorrer à vários *softwares*, os quais podem possuir formas de trabalho divergentes, o que pode promover confusão ao utilizador.

Em relação à execução e validação do projeto proposto para sanar a problemática exposta acima, concluiu-se que a aplicação recebeu resultados favoráveis enquanto a mesma foi testada pelo público, nas quais as páginas analisadas receberam boas métricas na primeira pesquisa (seção 5.6.2), mas conseguindo uma melhoria considerável na segunda pesquisa (seção 5.6.3), realizada após realização de ajustes listados na mesma seção, possibilitando uma melhor experiência do usuário. Logo, com as sugestões recebidas, existe espaço para melhorias do projeto proposto, deixando-o viável à comercialização por meio de seu modelo de negócio (seção 5.5).

REFERÊNCIAS

ARTSY. **A Better Way to Sell on Artsy: Buy Now & Make Offer** — Artsy. [S. l.: s. n.], a2023. Disponível em: <https://partners.artsy.net/new-selling-options/>. Acesso em: 17 jun. 2023.

ARTSY. **How does pricing work?** [S. l.: s. n.], b2023. Disponível em: <https://support.artsy.net/s/article/How-does-pricing-work>. Acesso em: 17 jun. 2023.

ARTSY. **I'm an artist. Can I submit my own work to sell?** — Artsy. [S. l.: s. n.], c2023. Disponível em: <https://support.artsy.net/s/article/Im-an-artist-Can-I-submit-my-own-work-to-sell>. Acesso em: 17 jun. 2023.

BAGUI, S. S.; EARP, R. W. **Database Design Using Entity-Relationship Diagrams**. Boca Raton: CRC Press, 2023. <https://doi.org/10.1201/9781003314455>.

BARNES, S. **How much will I make per sale?** [S. l.: s. n.], 2023. Disponível em: <https://help.printify.com/hc/en-us/articles/4483609656721-How-much-will-I-make-per-sale->. Acesso em: 17 jun. 2023.

BENNEY, M.; KISTLER, P. **Generative Art Guide: Examples, Software and Tools to Make Algorithm Art** — AIArtists. [S. l.: s. n.], 2023. Disponível em: <https://aiartists.org/generative-art-design>. Acesso em: 22 abr. 2023.

BODEN, M. A. Précis of The creative mind: Myths and mechanisms. **Behavioral and Brain Sciences**, v. 17, n. 3, 1994.

BODEN, M. A. **The Creative Mind: Myths and mechanisms**. [S. l.]: Routledge, 2004.

BODEN, M. A.; EDMONDS, E. What is generative art? **Digital Creativity**, 2009.

BONSIEPE, G.; KELLNER, P.; POESSNECKER, H. **Metodologia Experimental: Desenho Industrial**. Brasília: CNPq, 1984.

BROCK, D. C. **50 Years Later, We're Still Living in the Xerox Alto's World** — IEEE Spectrum. [S. l.: s. n.], 2023. Disponível em: <https://spectrum.ieee.org/xerox-alto>. Acesso em: 21 jun. 2023.

BROWN, D. M. **Communicating Design: Developing Web Site Documentation for Design and Planning**. Berkeley: New Riders, 2011.

BUSH, V. **As We May Think** — The Atlantic. [S. l.: s. n.], 1945. Disponível em: <https://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881/>. Acesso em: 21 jun. 2023.

C4IOS. **C4iOS** — GitHub. [S. l.: s. n.], 2018. Disponível em: <https://github.com/C4Labs/C4iOS>. Acesso em: 16 jun. 2023.

CHM. **Apple Macintosh** — Computer History Museum. [S. l.: s. n.], a2023. Disponível em: <https://www.computerhistory.org/>

revolution/personal-computers/17/303/1203. Acesso em: 21 jun. 2023.

CHM. **The Lisa: Apple's Most Influential Failure**. [S. l.: s. n.], a2022. Disponível em: <https://computerhistory.org/blog/the-lisa-apples-most-influential-failure/>. Acesso em: 21 jun. 2023.

CHM. **Xerox Alto — Computer History Museum**. [S. l.: s. n.], b2023. Disponível em: <https://www.computerhistory.org/revolution/input-output/14/347>. Acesso em: 21 jun. 2023.

CHM. **Yesterday's Computer of Tomorrow: The Xerox Alto — Computer History Museum**. [S. l.: s. n.], b2022. Disponível em: <https://computerhistory.org/events/yesterdays-computer-tomorrow-xerox-alto/>. Acesso em: 21 jun. 2023.

COURSERA. **Front End vs. Back End: Learning Skills and Tools — Coursera**. [S. l.: s. n.], a2023. Disponível em: <https://www.coursera.org/articles/front-end-vs-back-end>. Acesso em: 15 jun. 2023.

COURSERA. **What Does a Back-End Developer Do?** [S. l.: s. n.], b2023. Disponível em: <https://www.coursera.org/articles/back-end-developer>. Acesso em: 15 jun. 2023.

DAS, T. **10 Best Generative Art Tools to Use for Digital or NFT Artworks — The NFT Brief**. [S. l.: s. n.], 2022. Disponível em: <https://thenftbrief.com/generative-art-tools/>. Acesso em: 21 abr. 2023.

DESLAURIERS, M. **Generative Machines with Matt DesLauriers — YouTube**. [S. l.: s. n.], 2019. Disponível em: <https://www.youtube.com/watch?v=8Uo6zFwSO78>. Acesso em: 22 abr. 2023.

DEVMEDIA. **UML e os diagramas de casos de uso, sequência e atividade — DevMedia**. [S. l.: s. n.],. Disponível em: <https://www.devmedia.com.br/uml-e-os-diagramas-de-casos-de-uso-sequencia-e-atividade/21996>. Acesso em: 18 abr. 2023.

DIMITRIS. **Business Requirements vs Functional Requirements — Coara**. [S. l.: s. n.], 2020. Disponível em: <https://coara.co/blog/business-requirements-vs-functional-requirements>. Acesso em: 16 abr. 2023.

ETSY. **Fees & Payments Policy — Etsy**. [S. l.: s. n.], 2023. Disponível em: <https://www.etsy.com/legal/fees/>. Acesso em: 17 jun. 2023.

FISHWICK, P. **Aesthetic Computing**. [S. l.]: The MIT Press, 2006.

FOWLER, M. **UML Distilled: A Brief Guide to the Standard Object Modeling Language**. [S. l.]: Addison-Wesley, 2004.

GALANTER, P. **A Companion to Digital Art**. (C. Paul, org.). [S. l.]: John Wiley & Sons, Inc., 2016.

GRAM, L. **Artist to Artist: Every Question I Get About Selling My Artwork On Society6 with Luke Gram — Society6**. [S. l.:

s. n.], 2017. Disponível em: <https://blog.society6.com/every-question-get-selling-artwork-society6/>. Acesso em: 17 jun. 2023.

GROUP, O. M. **Unified Modeling Language™ (UML®) — Object Management Group**. [S. l.: s. n.], 2017. Disponível em: <http://www.omg.org/spec/UML/>. Acesso em: 10 jun. 2023.

GUBAYDULLIN, A.; DEREVYANNIKH, V.; GEYCHENKO, M.; RATMAN, K.; KLIMOV, A.; ROZHIN, A.; OVECHKINA, D. **Corpus — Behance**. [S. l.: s. n.], 2018. Disponível em: <https://www.behance.net/gallery/64306153/Corpus>. Acesso em: 21 abr. 2023.

HENCZ, A. **What is Generative Art? From Seminal Experiences to New Frontiers — Artland**. [S. l.: s. n.], 2023. Disponível em: <https://magazine.artland.com/generative-art/>. Acesso em: 22 abr. 2023.

HENKE, R. **Building Accessible Front-Ends — Medium**. [S. l.: s. n.], 2019. Disponível em: <https://medium.com/capital-one-tech/building-accessible-front-ends-298c03e5bd86>. Acesso em: 15 jun. 2023.

HINDERKS, A.; SCHREPP, M.; THOMASCHEWSKI, J. **User Experience Questionnaire**. [S. l.: s. n.], 2018. Disponível em: <https://www.ueq-online.org/>. Acesso em: 19 out. 2023.

HIRAMA, K. **Engenharia de software: qualidade e produtividade com tecnologia**. [S. l.]: Elsevier, 2011.

HOPE, C.; RYAN, J. **Digital Arts: An Introduction to New Media**. [S. l.]: Bloomsbury, 2014.

JANSEN, B. J. The graphical user interface. **ACM SIGCHI Bulletin**, v. 30, n. 2, p. 22–26, 1998. <https://doi.org/https://doi.org/10.1145/279044.279051>.

KORPUS. [S. l.: s. n.], 2021. Disponível em: <https://korpus.io/>. Acesso em: 21 abr. 2023.

KOTHARI, C. R. **Research Methodology: Methods & Techniques**. New Delhi: New Age International (P) Limited, 1990.

KOTLER, P.; KELLER, K. L. **Marketing Management**. Harlow: Pearson Education Limited, 2016.

KUMAR, H. **An Introduction to Generative Art: Examples, Artists & Software**. [S. l.: s. n.], 2022. Disponível em: <https://medium.com/@imhimanshu/an-introduction-to-generative-art-examples-artists-software-2a4627a36ff6>. Acesso em: 22 abr. 2023.

KUMAR, S. **Looper — GitHub**. [S. l.: s. n.], 2019. Disponível em: <https://github.com/sureskumar/Looper>. Acesso em: 16 abr. 2023.

LWJGL. **LWJGL**. [S. l.: s. n.], 2023. Disponível em: <https://www.lwjgll.org/>. Acesso em: 16 jun. 2023.

MARTINEZ, W. L. Graphical User Interfaces. **WIRES Computation Statistics**, v. 3, p. 119–133, 2011. <https://doi.org/10.1002/wics.150>.

MDN. **CRUD — Mozilla Developer Network**. [S. l.: s. n.], a2023. Disponível em: <https://developer.mozilla.org/en-US/docs/Glossary/CRUD>. Acesso em: 15 jun. 2023.

MDN. **Front-end web developer — Mozilla Developer Network**. [S. l.: s. n.], b2023. Disponível em: https://developer.mozilla.org/en-US/docs/Learn/Front-end_web_developer. Acesso em: 15 jun. 2023.

MDN. **Introduction to the server side — Mozilla Developer Network**. [S. l.: s. n.], c2023. Disponível em: https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Introduction. Acesso em: 15 jun. 2023.

NANNOU. **Why Nannou? — Nannou**. [S. l.: s. n.], 2022. Disponível em: https://guide.nannou.cc/why_nannou.html. Acesso em: 16 jun. 2023.

NIELSEN, J. **10 Usability Heuristics for User Interface Design**. [S. l.: s. n.], 2010. Disponível em: <https://www.nngroup.com/articles/ten-usability-heuristics>. Acesso em: 21 maio 2023.

NIELSEN, J. **Usability Engineering**. San Francisco: Morgan Kaufmann, 1993.

OSTERWALDER, A.; PIGNEUR, Y. **Business Model Generation**. Hoboken: John Wiley & Sons, 2010.

PANG, A. S.-K. **The Xerox PARC Visit — Making the Macintosh: Technology and Culture in Silicon Valley**. [S. l.: s. n.], 2000. Disponível em: <https://web.stanford.edu/dept/SUL/sites/mac/parc.html>. Acesso em: 21 jun. 2023.

PEARSON, M. **generative art: a practical guide using processing**. (M. P. Co., org.). [S. l.]: Manning Publications Co., 2011.

PERNICE, K. **UX Prototypes: Low Fidelity vs. High Fidelity — Nielsen Norman Group**. [S. l.: s. n.], 2016. Disponível em: <https://www.nngroup.com/articles/ux-prototype-hi-lo-fidelity/>. Acesso em: 13 jun. 2023.

PILONE, D.; PITMAN, N. **UML 2.0 in a Nutshell**. [S. l.]: O'Reilly, 2005.

PRINTIFY. **Pricing — Printify**. [S. l.: s. n.], 2023. Disponível em: <https://printify.com/pricing/>. Acesso em: 17 jun. 2023.

PROCESSING. **Reference — Processing**. [S. l.: s. n.], 2023. Disponível em: <https://processing.org/reference>. Acesso em: 4 maio 2023.

RAILS, R. on. **The Rails Command Line — Ruby on Rails**. [S. l.: s. n.], 2023. Disponível em: https://guides.rubyonrails.org/command_line.html. Acesso em: 15 jun. 2023.

REAS, C.; FRY, B. **A Modern Prometheus: The History of Processing by Casey Reas and Ben Fry** — **Medium**. [S. l.: s. n.], 2018. Disponível em: <https://medium.com/processing-foundation/a-modern-prometheus-59aed94abe85>. Acesso em: 4 maio 2023.

REAS, C.; MCWILLIAMS, C.; LUST. **Form + Code in Design, Art, and Architecture**. New York: Princeton Architectural Press, 2010.

REDBUBBLE. **Understanding how the new account fee may apply to you** — **Redbubble**. [S. l.: s. n.], 2023. Disponível em: <https://blog.redbubble.com/2023/04/how-account-fees-may-apply-to-you/>. Acesso em: 17 jun. 2023.

REDBUBBLE. **Where does my order ship from?** — **Redbubble**. [S. l.: s. n.], 2022. Disponível em: <https://help.redbubble.com/hc/en-us/articles/217196086-Where-does-my-order-from-Redbubble-print-and-ship-from->. Acesso em: 17 jun. 2023.

REIMER, J. **A History of the GUI**. [S. l.: s. n.], 2005. Disponível em: <https://arstechnica.com/features/2005/05/gui/>. Acesso em: 22 jun. 2023.

RUBY-DOC. **class String** — **Ruby-Doc**. [S. l.: s. n.], 2023. Disponível em: <https://ruby-doc.org/3.2.2/String.html>. Acesso em: 15 jun. 2023.

RUDD, J.; STERN, K.; ISENSEE, S. Low vs. high-fidelity prototyping debate. **Interactions**, v. 3, n. 1, p. 76–85, 1996.

SACRAMENTO, G. **Qual a diferença entre web designer e web developer?** — **Rock Content**. [S. l.: s. n.], 2021. Disponível em: <https://rockcontent.com/br/talent-blog/web-designer/>. Acesso em: 15 jun. 2023.

SCHREPP, M. **User Experience Questionnaire Handbook: All you need to know to apply the UEQ successfully in your projects**. [S. l.: s. n.], 2023.

SEBRAE. **Business Model Canvas: como construir seu modelo de negócio?** — **SEBRAE Digital**. [S. l.: s. n.], 2021. Disponível em: <https://digital.sebraers.com.br/blog/estrategia/business-model-canvas-como-construir-seu-modelo-de-negocio/>. Acesso em: 16 jun. 2023.

SRI. **1968 “Mother of All Demos” Forecasted Much of the Technology We Use Every Day** — **SRI International**. [S. l.: s. n.], 2012. Disponível em: <https://www.sri.com/blog-archive/1968-mother-of-all-demos-forecasted-much-of-the-technology-we-use-every-day/>. Acesso em: 21 jun. 2023.

STUMPF, A. **Create generative art without coding** — **Generative Hut**. [S. l.: s. n.], 2021. Disponível em: <https://www.generativehut.com/post/create-generative-art-without-coding>. Acesso em: 16 abr. 2023.

ŠVEHLA, M. **Online tools for generative graphics** — **Medium**. [S. l.: s. n.], 2018. Disponível em: <https://medium.com/@mila-nvehla/online-tools-for-generative-graphics-8722fe86d579>. Acesso em: 22 abr. 2023.

TAN, D. S. **human-machine interface** — **Britannica**. [S. l.: s. n.], 2022. Disponível em: <https://www.britannica.com/technology/human-machine-interface>. Acesso em: 21 jun. 2023.

TANEMBAUM, A. S.; WOODHULL, A. S. **Operating Systems: Design and Implementation**. New Jersey: Pearson Prentice Hall, 2006.

WOOD, Z. **Etsy sellers offering fake “handmade” products at ripoff prices, says Which? — The Guardian**. [S. l.: s. n.], 2023. Disponível em: <https://www.theguardian.com/money/2023/apr/27/etsy-sellers-handmade-products-prices-which>. Acesso em: 17 jun. 2023.