

EMPRESA MODELO:
ARCTECH

Projeto Integrador I

*software para acompanhamento
de propostas de arquitetura*

Fatec Ipiranga
Big Data para Negócios
Semestre 01/06

Helena B P Flausino

Segmento da empresa

Prestadora de Serviços de
'Projetos de Infraestrutura'

Qual o objetivo principal do negócio?

O objetivo principal é a prestação de serviços de projetos de infraestrutura, abrangendo diversas áreas como transportes, real state, institucionais/públicos, aeroportos, estádios, entre outros.

Posicionamento

descrição do problema e
soluções propostas



o problema

Dificuldade de acompanhar o andamento das propostas comerciais, e de visualizar métricas relevantes relacionadas a elas

afeta

Principalmente o operacional comercial, gestores de prospecção e os diretores.

cujo impacto é

Melhorar o entendimento e acompanhamento dos status das propostas enviadas, em andamento, canceladas e aprovadas. Entender essas taxas de conversão.

uma boa solução seria

Um software com os dados de todas as propostas enviadas, para de acordo com o histórico, gerar um panorama que possibilitasse gerar insights para o negócio.

Funcionalidades

Relatórios e BD

Identificar os tipos de projetos orçados,
solicitados(...):

- por área de atuação

- por tipo de setor

- (...)

Identificar os tickets médios destas propostas:

- por período

- por status de propostas

- (...)

Quantidade de propostas:

- enviadas

- aprovadas

- canceladas

- interrompidas

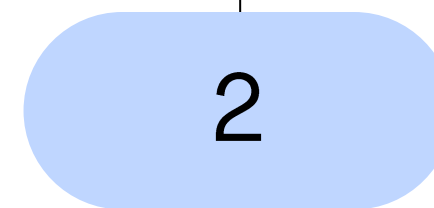
- (...)

Os usuários do sistema serão principalmente



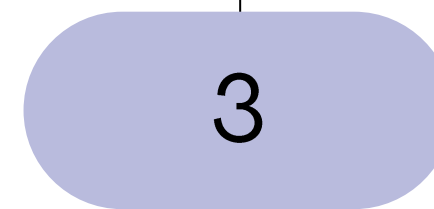
Administrador/Operacional

cadastra dados das propostas e status delas no sistema



Gerente de Propostas

atualiza status da proposta e valor



Diretor

extrai relatórios e informações chave da aplicação

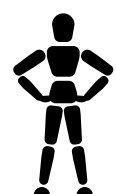
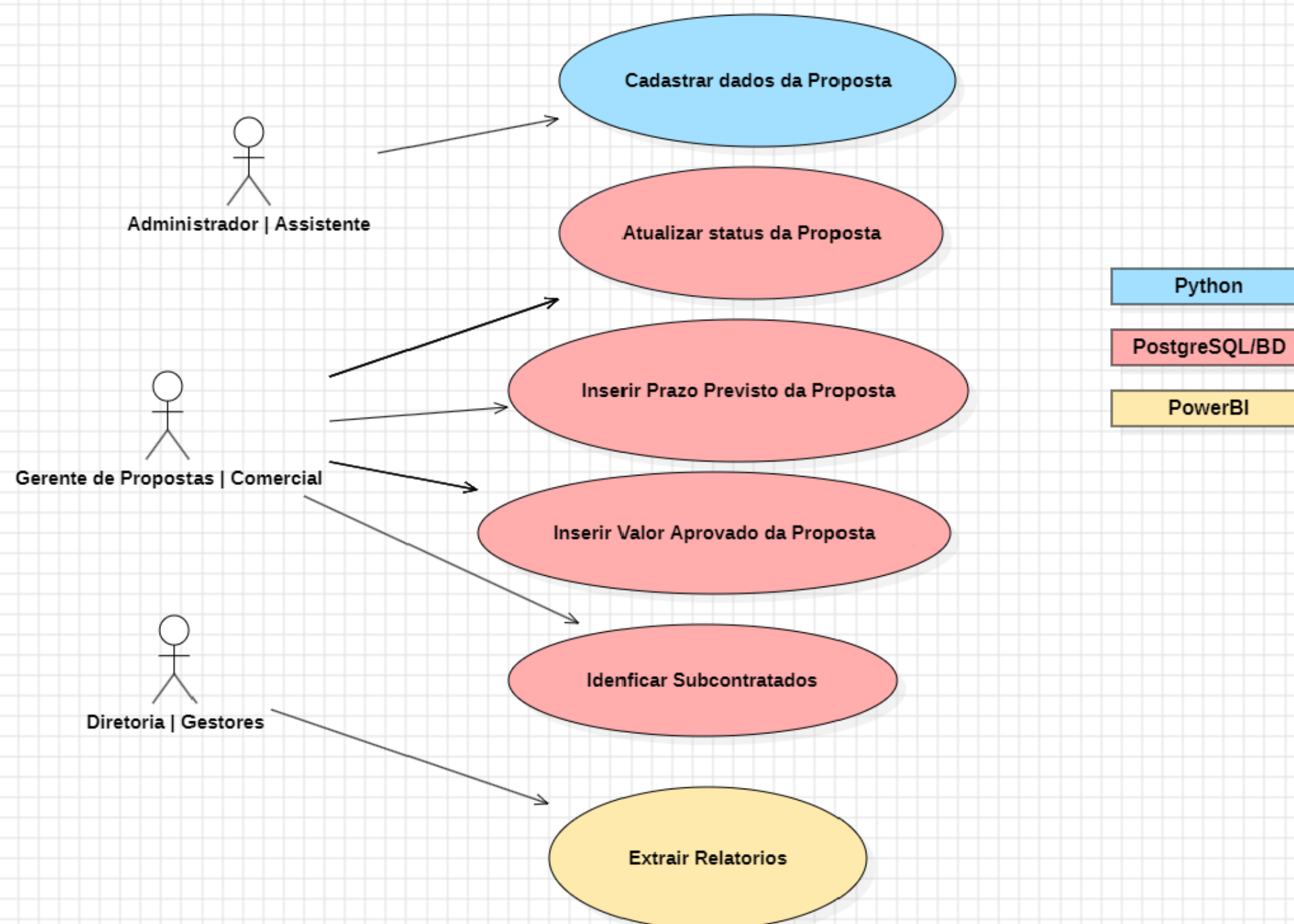
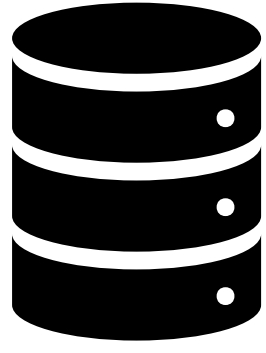


DIAGRAMA DE USO INICIAL



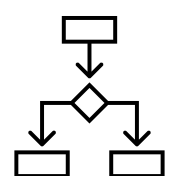
A equipe administrativa e os assistentes serão responsáveis por inserir os dados da proposta no sistema, incluindo os dados do cliente, tipo do projeto, localização, área de intervenção etc. Posteriormente, o gerente de propostas assumirá a responsabilidade de atualizar o status das propostas, identificando se foram enviadas, se estão em análise, aprovadas, não aprovadas, canceladas ou adiadas, além do prazo previsto para a execução de cada etapa de acordo com as negociações com o cliente final. Além disso, será incumbido de identificar os subcontratados e fornecedores necessários para o projeto, e registrar o valor final aprovado, juntamente com o prazo previsto, em semanas, para a adjudicação do contrato e a execução do projeto em si.

A diretoria e os gestores analisarão os dados históricos das propostas. Seu objetivo será identificar quais tipos de projetos foram mais solicitados, em quais regiões existem mais propostas, ticket médio, taxa de conversão entre outros. Eles também estarão atentos para verificar se as metas estão sendo cumpridas. Esses insights serão utilizados para tomar decisões estratégicas sobre o negócio como um todo.

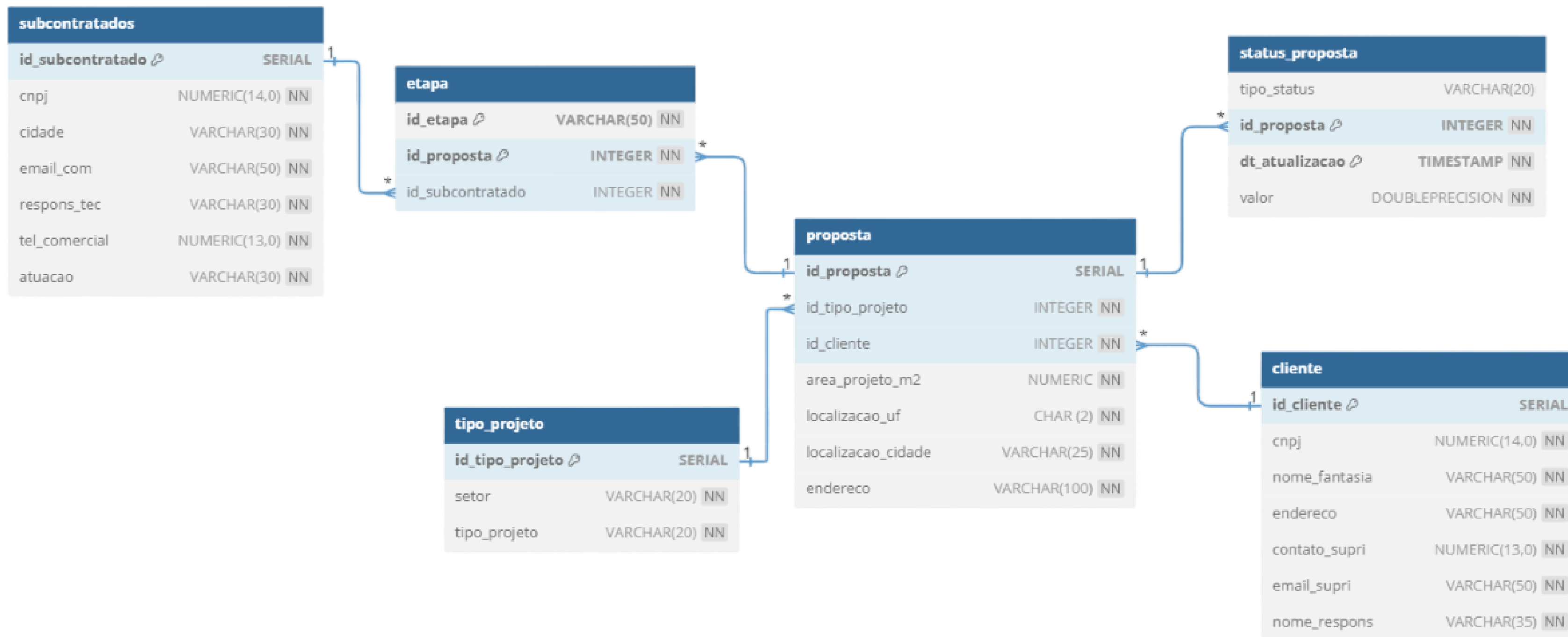


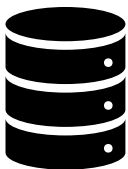
Banco de Dados





VERSÃO FINAL DO MODELO LÓGICO





pgAdmin 4
File Object Tools Help

Object Explorer

Servers (1)
PostgreSQL 16
Databases (6)
BD_PI_012024_G6
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas (1)
public
Aggregates
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (6)
cliente
etapa
proposta
status_proposta
subcontratados
tipo_projeto
Trigger Functions
Types
Views
Subscriptions
banco1
bancoguardado1405
hannuinho P2

Dashboard Properties SQL Statistics Dependencies Dependents Processes BD_PI_012024_G6/postgres@PostgreSQL 16*

BD_PI_012024_G6/postgres@PostgreSQL 16

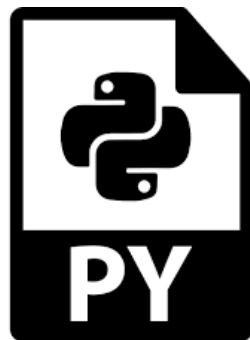
Query Query History

```
1 -- Active: 1716856021564@@127.0.0.1@5432@BD_PI_012024_G6
2 -- Listar todas as tabelas no banco de dados
3 SELECT table_schema, table_name
4 FROM information_schema.tables
5 WHERE table_schema NOT IN ('information_schema', 'pg_catalog')
6 ORDER BY table_schema, table_name ;
7
8 -- Listar todas as colunas de cada tabela
9 SELECT table_schema, table_name, column_name, data_type, character_maximum_length, is_nullable
10 FROM information_schema.columns
11 WHERE table_schema NOT IN ('information_schema', 'pg_catalog')
12 ORDER BY table_schema, table_name, ordinal_position ;
13
14 -- Listar todas as propostas populadas
15 SELECT *
16 FROM proposta ;
17
18 SELECT * FROM tipo_projeto ;
19
20 SELECT * FROM status_proposta ;
21
22 -- Listar todas as propostas e seus respectivos últimos status
23 SELECT sp.id_proposta AS "ID"
```

Data Output Messages Notifications

	tipo_status character varying (20)	id_proposta [PK] integer	dt_atualizacao [PK] timestamp without time zone	valor double precision
1	solicitada	1	2021-01-01 00:00:00	100000
2	em desenvolvimento	1	2021-01-05 00:00:00	500000
3	enviada	1	2021-01-10 00:00:00	300000
4	em analise	1	2021-01-15 00:00:00	700000
5	aprovada	1	2021-01-20 00:00:00	400000
6	solicitada	2	2021-02-01 00:00:00	800000
7	em desenvolvimento	2	2021-02-05 00:00:00	200000
8	enviada	2	2021-02-10 00:00:00	900000
9	cancelada	2	2021-02-15 00:00:00	600000
10	solicitada	3	2021-03-01 00:00:00	1100000
11	em desenvolvimento	3	2021-03-05 00:00:00	500000
Total rows: 231 of 231		Query complete 00:00:00.096		

Ln 26, Col 21



Interface Python


Inclusão de dados

Consulta rápida de dados operacionais





```
File Edit Selection View Go Run Terminal Help
tabelas_BD_G6.sql inserts_BD_G6.sql selects_BD_G6.sql PY_PI_G6.py x
PY_PI_G6.py > buscar_propostas
1 import PySimpleGUI as sg
2 import psycopg2
3
4 # inicialização Banco de Dados
5 conexaoBD = psycopg2.connect(host="localhost", database="BD_PI_012024_G6", user="postgres")
6
7 # carregar a imagem da logo
8 logo = sg.Image(filename='logo1.png', size=(936, 111))
9
10 # função para inserir dados na tabela proposta e atualizar um 1o status_proposta (tipo_
11 def inserir_proposta(conexaoBD, id_tipo_projeto, id_cliente, area_projeto_m2, localizac
12     cursor = conexaoBD.cursor()
13     cursor.execute(
14         """
15         INSERT INTO proposta (id_tipo_projeto, id_cliente, area_projeto_m2, localizacao
16         VALUES (%s, %s, %s, %s, %s, %s)
17         RETURNING id_proposta;
18         """,
19         (id_tipo_projeto, id_cliente, area_projeto_m2, localizacao_uf, localizacao_cida
20     )
21     id_proposta = cursor.fetchone()[0]
22     cursor.execute(
23         """
24         INSERT INTO status_proposta (id_proposta, tipo_status, dt_atualizacao, valor)
25         VALUES (%s, 'solicitada', CURRENT_TIMESTAMP, 0);
26         """,
27         (id_proposta,)
28     )
29     conexaoBD.commit()
30     cursor.close()
31     return id_proposta
32
33 # função para inserir dados na tabela status_proposta
34 def inserir_status_proposta(conexaoBD, id_proposta, tipo_status, valor):
35
36     cursor = conexaoBD.cursor()
37     cursor.execute(
38         """
39         INSERT INTO status_proposta (id_proposta, tipo_status, dt_atualizacao, valor)
40         VALUES (%s, %s, CURRENT_TIMESTAMP, %s);
41         """,
42         (id_proposta, tipo_status, valor)
43     )
44     conexaoBD.commit()
45     cursor.close()
46
47 # função para atualizar o status da proposta
48 def atualizar_status_proposta(conexaoBD, id_proposta, novo_status, valor):
49     cursor = conexaoBD.cursor()
50     cursor.execute(
51         """
52         UPDATE status_proposta
53         SET tipo_status = %s, dt_atualizacao = CURRENT_TIMESTAMP, valor = %s
54         WHERE id_proposta = %s;
55         """,
56         (novo_status, valor, id_proposta)
57     )
58     conexaoBD.commit()
59     cursor.close()
60
61 # função para excluir a proposta
62 def excluir_proposta(conexaoBD, id_proposta):
63     cursor = conexaoBD.cursor()
64     cursor.execute(
65         """
66         DELETE FROM proposta
67         WHERE id_proposta = %s;
68         """,
69         (id_proposta,)
70     )
71     conexaoBD.commit()
72     cursor.close()
73
74 # função para excluir o status da proposta
75 def excluir_status_proposta(conexaoBD, id_proposta):
76     cursor = conexaoBD.cursor()
77     cursor.execute(
78         """
79         DELETE FROM status_proposta
80         WHERE id_proposta = %s;
81         """,
82         (id_proposta,)
83     )
84     conexaoBD.commit()
85     cursor.close()
86
87 # função para listar as propostas
88 def listar_propostas(conexaoBD):
89     cursor = conexaoBD.cursor()
90     cursor.execute(
91         """
92         SELECT id_tipo_projeto, id_cliente, area_projeto_m2, localizacao_uf, localizacao_cida
93         FROM proposta;
94         """,
95     )
96     resultados = cursor.fetchall()
97     cursor.close()
98     return resultados
99
100 # função para listar os status das propostas
101 def listar_status_propostas(conexaoBD):
102     cursor = conexaoBD.cursor()
103     cursor.execute(
104         """
105         SELECT id_proposta, tipo_status, dt_atualizacao, valor
106         FROM status_proposta;
107         """,
108     )
109     resultados = cursor.fetchall()
110     cursor.close()
111     return resultados
112
113 # função para gerar o relatório
114 def gerar_relatorio(conexaoBD):
115     resultados_propostas = listar_propostas(conexaoBD)
116     resultados_status = listar_status_propostas(conexaoBD)
117     return resultados_propostas, resultados_status
118
119 # função para gerar o relatório em PDF
120 def gerar_relatorio_pdf(conexaoBD):
121     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
122     return gerar_relatorio_pdf(resultados_propostas, resultados_status)
123
124 # função para gerar o relatório em Excel
125 def gerar_relatorio_excel(conexaoBD):
126     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
127     return gerar_relatorio_excel(resultados_propostas, resultados_status)
128
129 # função para gerar o relatório em CSV
130 def gerar_relatorio_csv(conexaoBD):
131     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
132     return gerar_relatorio_csv(resultados_propostas, resultados_status)
133
134 # função para gerar o relatório em JSON
135 def gerar_relatorio_json(conexaoBD):
136     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
137     return gerar_relatorio_json(resultados_propostas, resultados_status)
138
139 # função para gerar o relatório em XML
140 def gerar_relatorio_xml(conexaoBD):
141     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
142     return gerar_relatorio_xml(resultados_propostas, resultados_status)
143
144 # função para gerar o relatório em YAML
145 def gerar_relatorio_yaml(conexaoBD):
146     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
147     return gerar_relatorio_yaml(resultados_propostas, resultados_status)
148
149 # função para gerar o relatório em Markdown
150 def gerar_relatorio_markdown(conexaoBD):
151     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
152     return gerar_relatorio_markdown(resultados_propostas, resultados_status)
153
154 # função para gerar o relatório em HTML
155 def gerar_relatorio_html(conexaoBD):
156     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
157     return gerar_relatorio_html(resultados_propostas, resultados_status)
158
159 # função para gerar o relatório em LaTeX
160 def gerar_relatorio_latex(conexaoBD):
161     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
162     return gerar_relatorio_latex(resultados_propostas, resultados_status)
163
164 # função para gerar o relatório em RTF
165 def gerar_relatorio_rtf(conexaoBD):
166     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
167     return gerar_relatorio_rtf(resultados_propostas, resultados_status)
168
169 # função para gerar o relatório em DOCX
170 def gerar_relatorio_docx(conexaoBD):
171     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
172     return gerar_relatorio_docx(resultados_propostas, resultados_status)
173
174 # função para gerar o relatório em PPTX
175 def gerar_relatorio_pptx(conexaoBD):
176     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
177     return gerar_relatorio_pptx(resultados_propostas, resultados_status)
178
179 # função para gerar o relatório em XLSX
180 def gerar_relatorio_xlsx(conexaoBD):
181     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
182     return gerar_relatorio_xlsx(resultados_propostas, resultados_status)
183
184 # função para gerar o relatório em CSV
185 def gerar_relatorio_csv(conexaoBD):
186     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
187     return gerar_relatorio_csv(resultados_propostas, resultados_status)
188
189 # função para gerar o relatório em JSON
190 def gerar_relatorio_json(conexaoBD):
191     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
192     return gerar_relatorio_json(resultados_propostas, resultados_status)
193
194 # função para gerar o relatório em XML
195 def gerar_relatorio_xml(conexaoBD):
196     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
197     return gerar_relatorio_xml(resultados_propostas, resultados_status)
198
199 # função para gerar o relatório em YAML
200 def gerar_relatorio_yaml(conexaoBD):
201     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
202     return gerar_relatorio_yaml(resultados_propostas, resultados_status)
203
204 # função para gerar o relatório em Markdown
205 def gerar_relatorio_markdown(conexaoBD):
206     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
207     return gerar_relatorio_markdown(resultados_propostas, resultados_status)
208
209 # função para gerar o relatório em HTML
210 def gerar_relatorio_html(conexaoBD):
211     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
212     return gerar_relatorio_html(resultados_propostas, resultados_status)
213
214 # função para gerar o relatório em LaTeX
215 def gerar_relatorio_latex(conexaoBD):
216     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
217     return gerar_relatorio_latex(resultados_propostas, resultados_status)
218
219 # função para gerar o relatório em RTF
220 def gerar_relatorio_rtf(conexaoBD):
221     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
222     return gerar_relatorio_rtf(resultados_propostas, resultados_status)
223
224 # função para gerar o relatório em DOCX
225 def gerar_relatorio_docx(conexaoBD):
226     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
227     return gerar_relatorio_docx(resultados_propostas, resultados_status)
228
229 # função para gerar o relatório em PPTX
230 def gerar_relatorio_pptx(conexaoBD):
231     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
232     return gerar_relatorio_pptx(resultados_propostas, resultados_status)
233
234 # função para gerar o relatório em XLSX
235 def gerar_relatorio_xlsx(conexaoBD):
236     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
237     return gerar_relatorio_xlsx(resultados_propostas, resultados_status)
238
239 # função para gerar o relatório em CSV
240 def gerar_relatorio_csv(conexaoBD):
241     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
242     return gerar_relatorio_csv(resultados_propostas, resultados_status)
243
244 # função para gerar o relatório em JSON
245 def gerar_relatorio_json(conexaoBD):
246     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
247     return gerar_relatorio_json(resultados_propostas, resultados_status)
248
249 # função para gerar o relatório em XML
250 def gerar_relatorio_xml(conexaoBD):
251     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
252     return gerar_relatorio_xml(resultados_propostas, resultados_status)
253
254 # função para gerar o relatório em YAML
255 def gerar_relatorio_yaml(conexaoBD):
256     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
257     return gerar_relatorio_yaml(resultados_propostas, resultados_status)
258
259 # função para gerar o relatório em Markdown
260 def gerar_relatorio_markdown(conexaoBD):
261     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
262     return gerar_relatorio_markdown(resultados_propostas, resultados_status)
263
264 # função para gerar o relatório em HTML
265 def gerar_relatorio_html(conexaoBD):
266     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
267     return gerar_relatorio_html(resultados_propostas, resultados_status)
268
269 # função para gerar o relatório em LaTeX
270 def gerar_relatorio_latex(conexaoBD):
271     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
272     return gerar_relatorio_latex(resultados_propostas, resultados_status)
273
274 # função para gerar o relatório em RTF
275 def gerar_relatorio_rtf(conexaoBD):
276     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
277     return gerar_relatorio_rtf(resultados_propostas, resultados_status)
278
279 # função para gerar o relatório em DOCX
280 def gerar_relatorio_docx(conexaoBD):
281     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
282     return gerar_relatorio_docx(resultados_propostas, resultados_status)
283
284 # função para gerar o relatório em PPTX
285 def gerar_relatorio_pptx(conexaoBD):
286     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
287     return gerar_relatorio_pptx(resultados_propostas, resultados_status)
288
289 # função para gerar o relatório em XLSX
290 def gerar_relatorio_xlsx(conexaoBD):
291     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
292     return gerar_relatorio_xlsx(resultados_propostas, resultados_status)
293
294 # função para gerar o relatório em CSV
295 def gerar_relatorio_csv(conexaoBD):
296     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
297     return gerar_relatorio_csv(resultados_propostas, resultados_status)
298
299 # função para gerar o relatório em JSON
300 def gerar_relatorio_json(conexaoBD):
301     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
302     return gerar_relatorio_json(resultados_propostas, resultados_status)
303
304 # função para gerar o relatório em XML
305 def gerar_relatorio_xml(conexaoBD):
306     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
307     return gerar_relatorio_xml(resultados_propostas, resultados_status)
308
309 # função para gerar o relatório em YAML
310 def gerar_relatorio_yaml(conexaoBD):
311     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
312     return gerar_relatorio_yaml(resultados_propostas, resultados_status)
313
314 # função para gerar o relatório em Markdown
315 def gerar_relatorio_markdown(conexaoBD):
316     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
317     return gerar_relatorio_markdown(resultados_propostas, resultados_status)
318
319 # função para gerar o relatório em HTML
320 def gerar_relatorio_html(conexaoBD):
321     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
322     return gerar_relatorio_html(resultados_propostas, resultados_status)
323
324 # função para gerar o relatório em LaTeX
325 def gerar_relatorio_latex(conexaoBD):
326     resultados_propostas, resultados_status = gerar_relatorio(conexaoBD)
327     return gerar_relatorio_latex(resultados_propostas, resultados_status)
328
329 #
```



ARCTECH

Proposta

Tipo Projeto

Cliente

Área Projeto m²

Localização UF

Localização Cidade

Endereço

Inserir dados proposta SOLICITADA

Status Proposta

ID Proposta

Tipo Status

Valor

Atualizar status proposta

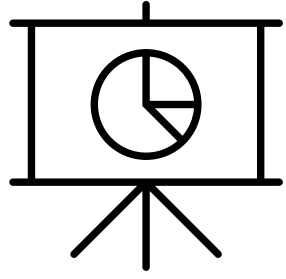
Buscar Propostas

ID Proposta

Buscar proposta

ID	Tipo Projeto	Setor	Área m²	UF	Cidade	Status	Dt Status	Valor Total

TRIAL PERIOD ⌕ ends in 29 days. Sign up.



Dados Consolidados

Dashboard de análise de dados



Power BI

