UNIFEOB
ESCOLA DE NEGÓCIOS

2023

PROJETO INTEGRADO

UNIFEOB

CENTRO UNIVERSITÁRIO DA FUNDAÇÃO DE ENSINO OCTÁVIO BASTOS

ESCOLA DE NEGÓCIOS

A.D.S. E CIÊNCIA DA COMPUTAÇÃO

PROJETO INTEGRADO

IOT DATA STREAMER

ILight

SÃO JOÃO DA BOA VISTA, SP

NOVEMBRO 2023

UNIFEOB

CENTRO UNIVERSITÁRIO DA FUNDAÇÃO DE ENSINO OCTÁVIO BASTOS

ESCOLA DE NEGÓCIOS

A.D.S. E CIÊNCIA DA COMPUTAÇÃO

PROJETO INTEGRADO

IOT DATA STREAMER

ILight

MÓDULO MODELAGEM E DESENVOLVIMENTO DE SISTEMAS

Gestão Financeira – Profa. Renata Elizabeth de Alencar Marcondes

Programação Orientada a Objeto – Prof. Nivaldo Andrade

Lógica de Programação – Prof. Marcelo Ciacco de Almeida

Modelagem de Dados – Prof. Max Streicher Vallim

Projeto de Modelagem e Desenvolvimento de Sistemas – Profª. Mariângela Martimbianco Santos

Estudantes:

Guilherme Carvalho André Rodrigues, RA 23000766
Gustavo Aurelio Barboza da Cruz, RA 23001003
João Pedro Rezende Ferreira, RA 23001017
Luis Gabriel Brito Felicio, RA 23000698
Matheus Gimenes Taconi, RA 23001163
Vitor Guilherme Dantas Zuin, RA 23001160

SÃO JOÃO DA BOA VISTA, SP
NOVEMBRO 2023

SUMÁRIO

1. INTRODUÇÃO 4
2. DESCRIÇÃO DA EMPRESA 7
3. PROJETO INTEGRADO 8
   3.1 PROGRAMAÇÃO ORIENTADA A OBJETO 8
      3.1.1 CLASSES E OBJETOS 8
      3.1.2 ATRIBUTOS, MÉTODOS, ENCAPSULAMENTO, HERANÇA E POLIMORFISMO. 8
      3.1.3 MÉTODOS ESTÁTICOS, PÚBLICOS E PRIVADOS 8
   3.2 LÓGICA DE PROGRAMAÇÃO 8
      3.2.1 CONCEITOS FUNDAMENTAIS DO DESENVOLVIMENTO DE SOFTWARE 9
      3.2.2 DESENVOLVIMENTO DE APLICAÇÕES DESKTOP 9
   3.3 MODELAGEM DE DADOS 9
      3.3.1 MODELO CONCEITUAL 9
      3.3.2 MODELO LÓGICO E FÍSICO 9
      3.3.3 SQL 9
   3.4 GESTÃO FINANCEIRA 10
      3.4.1 CLASSIFICAÇÃO DOS CUSTOS 10
      3.4.2 CUSTOS DO PRODUTO / SERVIÇO 10
      3.4.3 PRECIFICAÇÃO 10
   3.5 CONTEÚDO DA FORMAÇÃO PARA A VIDA: GERENCIANDO FINANÇAS 10
      3.5.1 GERENCIANDO FINANÇAS 11
      3.5.2 ESTUDANTES NA PRÁTICA 11
4. CONCLUSÃO 13

REFERÊNCIAS 14

ANEXOS 15

ISSN 1983-6767

1. INTRODUÇÃO

Nos últimos anos, a automação residencial tem experimentado um crescimento exponencial, transformando a maneira como interagimos com nossos espaços domésticos. Uma das inovações mais notáveis nesse campo é a introdução das persianas automáticas, que oferecem não apenas conveniência, mas também eficiência energética e melhor controle da iluminação e privacidade em nossas residências. Esta revolução tecnológica nas persianas, combinando hardware e software inteligente, não apenas facilita a vida cotidiana, mas também representa um passo importante na busca por ambientes mais sustentáveis e confortáveis, esperamos fornecer insights valiosos que ajudarão a entender o impacto dessa inovação no cotidiano das pessoas e no futuro da automação residencial..

A automação residencial tem se tornado uma tendência crescente no setor de design de interiores e construção civil. Essa revolução tecnológica tem proporcionado maior conforto, eficiência energética e segurança nas residências, tornando-as mais inteligentes e adaptadas ao estilo de vida moderno. Um elemento fundamental nesse cenário é a automação das persianas, que desempenham um papel crucial na regulação da luz natural e privacidade.

Com isso, nosso objetivo consiste em ajudar as pessoas com alguma dificuldade no dia a dia, em lojas, empresas, casas e etc. Sabendo desse problema, elaboramos algumas pesquisas e chegamos a conclusão de que é um problema existente e comum, não só em estabelecimentos, estamos no século de desenvolvimento e muitas pessoas querem modernizar até as tarefas simples como abrir e fechar janelas e cortinas. Pensando nisso, criamos uma Persiana automática com site e app, podendo controlar os horários de abertura e fechamento.

Em nossas plataformas utilizamos um design de fácil entendimento e aprendizado, pensando em atender todas as faixas etárias, armazenamos as atividades em um histórico podendo ser consultado pelos usuários a qualquer hora em nosso aplicativo ou site para ver consumo diário e mensal de utilização.

Tendo um sistema de seguro podendo entrar com google e com fácil criação de contas, métodos de segurança como biometria e até mesmo o reconhecimento facial para facilitar o acesso dos usuários.

Ao longo deste estudo, será evidenciado que as persianas automáticas não são apenas um elemento estético, mas também um componente inteligente que pode aprimorar significativamente o conforto e a funcionalidade das residências modernas.

MURATORI, José Roberto; DAL BÓ, Paulo Henrique. Capítulo I Automação residencial: histórico, definições e conceitos. **O Setor elétrico**, p. 70-77, 2011.

FERNANDES, Flávia Gonçalves et al. Sistema de automação residencial controlado por dispositivos móveis e vestíveis. **FTT Journal of Engineering and Business**, n. 1, 2016.

2. DESCRIÇÃO DA EMPRESA

A empresa em foco foi nomeada como Ilight em nove de abril de dois mil e vinte três, localizada na avenida doutor Otávio da Silva Bastos, 2439 - Jardim Nova São João, São João da Boa Vista - São Paulo, seu cadastro nacional de pessoa jurídica é 59.764.555/0001-52.

A ILight é uma empresa que compete com suas atividades e profissionalismo no setor de fabricação, instalação e manutenção em automação de móveis e decorativos caseiros é conhecida nacionalmente por seus facilitadores que tornam a vida de um dono(a) de casa menos monótona e totalmente automatizada. Iniciaram suas fabricações em persianas automáticas no qual permitia que o usuário tivesse controle sobre horário ao abrir ou fechar suas persianas, assim mantendo o maior fluxo de luz do sol em suas casas e evitando que a noite fique aberta para terceiros. Cresceu no mercado empresarial pelo seu suporte em curto período de tempo e bom relacionamento com seus clientes, contribuindo para uma maior confiança em seus produtos.

3. PROJETO INTEGRADO

Como nosso projeto visa automatizar e ser eficiente e facilitar a utilização e a vida do usuário, cada conteúdo da unidade de estudo é muito importante, e iremos apresentar cada ideia utilizada aos mínimos detalhes.

3.1 PROGRAMAÇÃO ORIENTADA A OBJETO

Utilizamos programação orientada a objetos em nosso site para autenticação dos usuários e para a segurança de senha utilizando Hash bcrypt.

Código anexo 1:

```
77          import mysql.connector
78          import bcrypt
79
80  class SistemaAutenticacaoComBanco:
81      def __init__(self, host, user, password,
          database):
82          self.conn = mysql.connector.connect(
83              host=host,
84              user=user,
85              password=password,
86              database=database
87          )
88      def autenticar(self, nome_usuario, email
          , senha):
89          cursor = self.conn.cursor()
90          cursor.execute("SELECT senha FROM
              usuarios WHERE nome_usuario = %s
              AND email = %s", (nome_usuario,
              email))
91          resultado = cursor.fetchone()
92
93          if resultado and bcrypt.checkpw
              (senha.encode('utf-8'),
              resultado[0].encode('utf-8')):
94              print("Autenticação bem
                  -sucedida!")
95          else:
96              print("Falha na autenticação.
                  Verifique o nome de usuário, e
                  -mail e senha.")
97          cursor.close()
98      def adicionar_usuario(self, nome_usuario
          , email, senha):
99          cursor = self.conn.cursor()
100         hashed_senha = bcrypt.hashpw(senha
              .encode('utf-8'), bcrypt.gensalt
              ())
101         cursor.execute("INSERT INTO usuarios
              (nome_usuario, email, senha)
              VALUES (%s, %s, %s)",
              (nome_usuario, email, hashed_senha
              .decode('utf-8')))
102         self.conn.commit()
103
104         cursor.close()
```

Utilizamos o módulo bcrypt para gerar um hash seguro da senha ao adicionar um novo usuário no banco de dados. A função bcrypt.gensalt() é usada para gerar um salt aleatório.

Nós decidimos utilizar o hash após pesquisar muito sobre a segurança das senhas dos usuários em nosso site.

Ao autenticar um usuário, o método bcrypt.checkpw é usado para comparar a senha fornecida com o hash de senha armazenado no banco de dados.

3.1.1 CLASSES E OBJETOS

Uma classe é um modelo ou uma descrição abstrata de um objeto. Ela define os atributos (variáveis) e métodos (funções) que os objetos desse tipo terão. Em essência, a classe é um "plano" para criar objetos.

As classes são como moldes que especificam a estrutura e o comportamento dos objetos que serão criados a partir delas.

Por exemplo, você pode ter uma classe "Pessoa" que define os atributos (nome, idade) e métodos para exibir as informações (mostrar informações) que as pessoas compartilham.

Exemplo:

Anexo 2

```python
class Pessoa:

    def __init__(self, nome, idade):
        self.nome = nome
        self.idade = idade

    def mostrar_informacoes(self):
        print(f"Nome: {self.nome}, Idade:
          {self.idade}")

pessoa1 = Pessoa("João", 15)
pessoa2 = Pessoa("marco", 19)

pessoa1.mostrar_informacoes()
pessoa2.mostrar_informacoes()
```

A saída do código será "nome: João, idade 15", e "nome: Marco, idade 19"

3.1.2 ATRIBUTOS, MÉTODOS, ENCAPSULAMENTO, HERANÇA E POLIMORFISMO.

Atributos: São as características ou dados associados a um objeto em uma classe.

Exemplo: Em uma classe "Pessoa", os atributos podem ser "idade", "nome".

```python
class Animal:
    def __init__(self, especie, nome):
        self.especie = especie
        self.nome = nome

meu_animal = Animal("Cachorro", "Rex")
print(f"Meu animal de estimação é um
  {meu_animal.especie} chamado {meu_animal
  .nome}")
```

Métodos: São funções definidas dentro de uma classe que descrevem o comportamento dos objetos dessa classe.

Exemplo: Em uma classe "Animal", os métodos podem ser "latir" e "miar".

Anexo 3

```python
class Animal:
    def __init__(self, especie, nome):
        self.especie = especie
        self.nome = nome

    def fazer_som(self):
        pass

class Cachorro(Animal):
    def fazer_som(self):
        return "Au au!"

class Gato(Animal):
    def fazer_som(self):
        return "Miau!"

meu_cachorro = Cachorro("Cachorro", "Rex")
meu_gato = Gato("Gato", "Whiskers")

print(meu_cachorro.fazer_som())
print(meu_gato.fazer_som())
```

a saída seria "Au au!" E "Miau!".

Encapsulamento: É o conceito de esconder os detalhes internos de uma classe e fornecer uma interface pública para interagir com ela.

Em Python, isso é alcançado por convenção, usando underscore (como _atributo_privado) para indicar que um atributo ou método deve ser tratado como privado.

Exemplo:

Anexo 4

```python
class Pessoa:
    def __init__(self, nome, idade):
        self.__nome = nome
        self.__idade = idade

    def get_nome(self):
        return self.__nome

    def set_idade(self, nova_idade):
        if nova_idade > 0:
            self.__idade = nova_idade

pessoa = Pessoa("Luis", 19)
print(f"Nome: {pessoa.get_nome()}")
pessoa.set_idade(67)
```

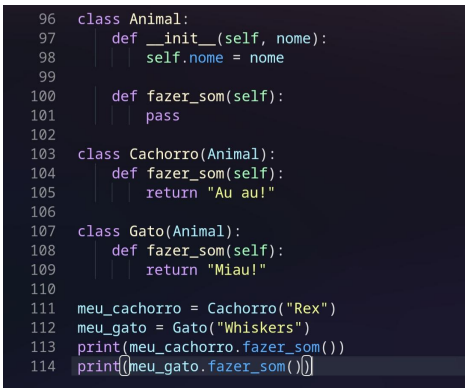
Esse exemplo mostra como o encapsulamento é usado para proteger os atributos da classe, garantindo que eles sejam acessados e modificados de maneira controlada por meio de métodos públicos.

Herança: É um mecanismo que permite criar uma nova classe (subclasse) com base em uma classe existente (superclasse).

A subclasse herda os atributos e métodos da superclasse e pode adicionar ou modificar seu comportamento.Isso promove a reutilização de código e a criação de hierarquias de classes.

Exemplo:

Anexo 5



```
class Animal:
    def __init__(self, nome):
        self.nome = nome

    def fazer_som(self):
        pass

class Cachorro(Animal):
    def fazer_som(self):
        return "Au au!"

class Gato(Animal):
    def fazer_som(self):
        return "Miau!"

meu_cachorro = Cachorro("Rex")
meu_gato = Gato("Whiskers")
print(meu_cachorro.fazer_som())
print(meu_gato.fazer_som())
```

O resultado do código é "au au" para o cachorro e "Miau" para o Gato, esse é um exemplo básico de herança onde o cachorro e Gato herdam características e comportamentos da classe base Animal.

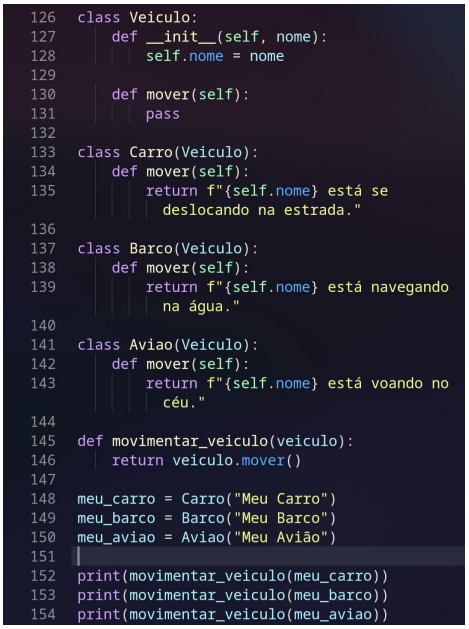
Polimorfismo: É a capacidade de objetos de diferentes classes responderem de maneira única a um método comum.

Isso permite que você trate objetos de diferentes classes de maneira uniforme, desde que implementem o mesmo método.

Pode ser alcançado através da herança e da substituição de métodos em subclasse.

Exemplo:

Anexo 6



```
class Veiculo:
    def __init__(self, nome):
        self.nome = nome

    def mover(self):
        pass

class Carro(Veiculo):
    def mover(self):
        return f"{self.nome} está se
          deslocando na estrada."

class Barco(Veiculo):
    def mover(self):
        return f"{self.nome} está navegando
          na água."

class Aviao(Veiculo):
    def mover(self):
        return f"{self.nome} está voando no
          céu."

def movimentar_veiculo(veiculo):
    return veiculo.mover()

meu_carro = Carro("Meu Carro")
meu_barco = Barco("Meu Barco")
meu_aviao = Aviao("Meu Avião")

print(movimentar_veiculo(meu_carro))
print(movimentar_veiculo(meu_barco))
print(movimentar_veiculo(meu_aviao))
```

Esse é um exemplo simples de polimorfismo onde objetos de classes diferentes respondem de maneira única ao método em comum mover.

3.1.3 MÉTODOS ESTÁTICOS, PÚBLICOS E PRIVADOS

Métodos Públicos: São métodos que podem ser acessados e chamados de qualquer lugar no código, dentro ou fora da classe.

Geralmente, esses métodos são usados para fornecer uma interface pública para interagir com os objetos da classe.São definidos sem o uso de underscores no início ou no final do nome do método.

Métodos Privados: São métodos que devem ser acessados somente de dentro da classe onde foram definidos.

Geralmente, são usados para implementações internas e detalhes de funcionamento da classe.

São definidos com um underscore no início do nome do método, como _metodo_privado().

Métodos Estáticos: São métodos que pertencem à classe em vez de instâncias da classe.

Não têm acesso aos atributos ou métodos da instância e não podem ser chamados usando self.

São úteis quando você deseja criar métodos que não dependem de um estado de objeto específico.

São definidos como métodos de classe com o decorador @staticmethod ou chamados diretamente pela classe, sem necessidade de uma instância.

Exemplo:

Anexo 7

```
157 class MinhaClasse:
158     def __init__(self, valor):
159         self.valor = valor
160 
161     def metodo_publico(self):
162         return "Método público"
163 
164     def _metodo_privado(self):
165         return "Método privado"
166 
167     @staticmethod
168     def metodo_estatico():
169         return "Método estático"
170 
171 
172 objeto = MinhaClasse(42)
173 
174 print(objeto.metodo_publico())
175 print(objeto._metodo_privado())
176 print(MinhaClasse.metodo_estatico())
```

No exemplo mostrado acima "metodo_publico" é um método público, "_metodo_privado" é um método privado e "metodo_estatico" é um método estático.
Cada um deles possui um propósito e um acesso diferente.

ISSN 1983-6767

03.2 LÓGICA DE PROGRAMAÇÃO

Lógica de Programação: A lógica de programação se refere à capacidade de pensar de forma lógica e estruturada para resolver problemas por meio de um programa de computador.

Envolve a compreensão de fluxos de controle, estruturas de repetição e tomada de decisões.

Algoritmos: Um algoritmo é um conjunto de passos sequenciais que descreve como realizar uma tarefa ou resolver um problema.

Os algoritmos são a base de qualquer programa de computador.

Variáveis: Variáveis são espaços de armazenamento que mantêm valores temporários na memória do computador.

Em JavaScript, você pode declarar variáveis usando palavras-chave como var, let e const.

Tipos de Dados: Os tipos de dados em JavaScript incluem números, strings, booleanos, arrays, objetos e muito mais.Cada tipo de dado tem um propósito específico e é representado de forma diferente na linguagem.

Funções: Funções são blocos de código reutilizáveis que realizam tarefas específicas.

Em JavaScript, ela pode definir funções usando a palavra-chave function.

Estruturas Condicionais: Estruturas condicionais, como if, else if e else, permitem que você tome decisões com base em condições.

Operadores Lógicos e Operadores de Comparação: Operadores lógicos, como && (AND) e || (OR), são usados para combinar condições.Operadores de comparação, como == (igual) e != (diferente), são usados para comparar valores.

Prototipação: A prototipação é um conceito de desenvolvimento de software que envolve a criação de protótipos ou versões iniciais de um sistema para validar ideias e funcionalidades.

Aplicações Desktop com Electron JS: O Electron é um framework que permite criar aplicativos desktop multiplataforma usando tecnologias da web, como HTML, CSS e JavaScript.

Banco de Dados Relacional (MySQL): Bancos de dados relacionais, como o MySQL, organizam dados em tabelas com relações entre elas.

Eles são usados para armazenar e recuperar informações de forma estruturada.

Esses conceitos são fundamentais para o desenvolvimento de software em JavaScript e para a criação de aplicações desktop com Electron JS, especialmente quando se lida com banco de dados como o MySQL.

3.2.1 CONCEITOS FUNDAMENTAIS DO DESENVOLVIMENTO DE SOFTWARE

Algoritmos: São conjuntos de instruções bem definidas e sequenciais para resolver um problema. São a base de qualquer programa de computador.

Variáveis: São espaços de armazenamento para guardar dados temporariamente.

Elas podem conter números, textos, booleanos e outros tipos de dados.Tipos de Dados: Representam os diferentes tipos de valores que uma variável pode conter. Exemplos incluem números inteiros, números de ponto flutuante, strings (textos) e booleanos (verdadeiro/falso).

Funções: São blocos de código que executam tarefas específicas e podem ser reutilizados. As funções aceitam parâmetros e retornam valores.

Estruturas Condicionais: São usadas para tomar decisões no código. Por exemplo, o if permite executar diferentes blocos de código com base em condições.

Operadores: São símbolos ou palavras-chave usadas para realizar operações em variáveis e valores, como adição, subtração, igualdade, comparação, etc.

Dominar esses conceitos é essencial para escrever programas eficazes, resolver problemas de forma eficiente e criar software de alta qualidade.

3.2.2 DESENVOLVIMENTO DE APLICAÇÕES DESKTOP

Em nosso site, empregamos o framework Electron para prototipar a aplicação. Para alcançar esse objetivo, começamos por instalar o Node.js. Em seguida, incorporamos o Electron ao projeto. Dado que a maior parte do site já estava desenvolvida, a etapa de front-end envolveu principalmente a transferência do código JavaScript, a integração com o HTML e a configuração da conexão entre o arquivo package.json e o código JavaScript.

Fazer essa integração não foi uma tarefa simples, uma vez que nunca tínhamos trabalhado com o Node.js ou o Electron anteriormente. No entanto, com dedicação e tempo dedicados ao aprendizado, conseguimos superar os desafios e tornar isso possível.

JS:

Anexo 8

pg1.js

```
const { app, BrowserWindow } = require
  ('electron');

function createWindow() {
  const win = new BrowserWindow({ width: 800,
    height: 600 });
  win.loadFile('nsnsn.html');
}

app.whenReady().then(createWindow);
```

Package.json:

Anexo 9

package.json

```
{
  "name": "ilight.com",
  "version": "2.0.0",
  "description": "Ilight protótipo",
  "main": "pg1.js",
  "scripts": {
    "dev": "electron ."
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "devDependencies": {
    "electron": "^25.1.0"
  }
}
```

ISSN 1983-6767

3.3 MODELAGEM DE DADOS

Considerando que nosso site para Desktop é simples utilizamos algumas funções básicas no sql, nossas tables são Usuário, Gráfico de Uso (histórico), Persiana que mostra em qual persiana cada usuário está logado no momento. E utilizamos insert, Delete, Update e Selects para usar como exemplo.

Imagens:

Table usuário e persiana:

Criamos uma tabela usuário que será responsável por armazenar dados do usuário como Nome, E-mail e Senha, que será salvo a cada usuário que utilize nossas persianas, lembrando que cada usuário pode ter mais de uma persiana logada em uma única conta.

Table histórico:

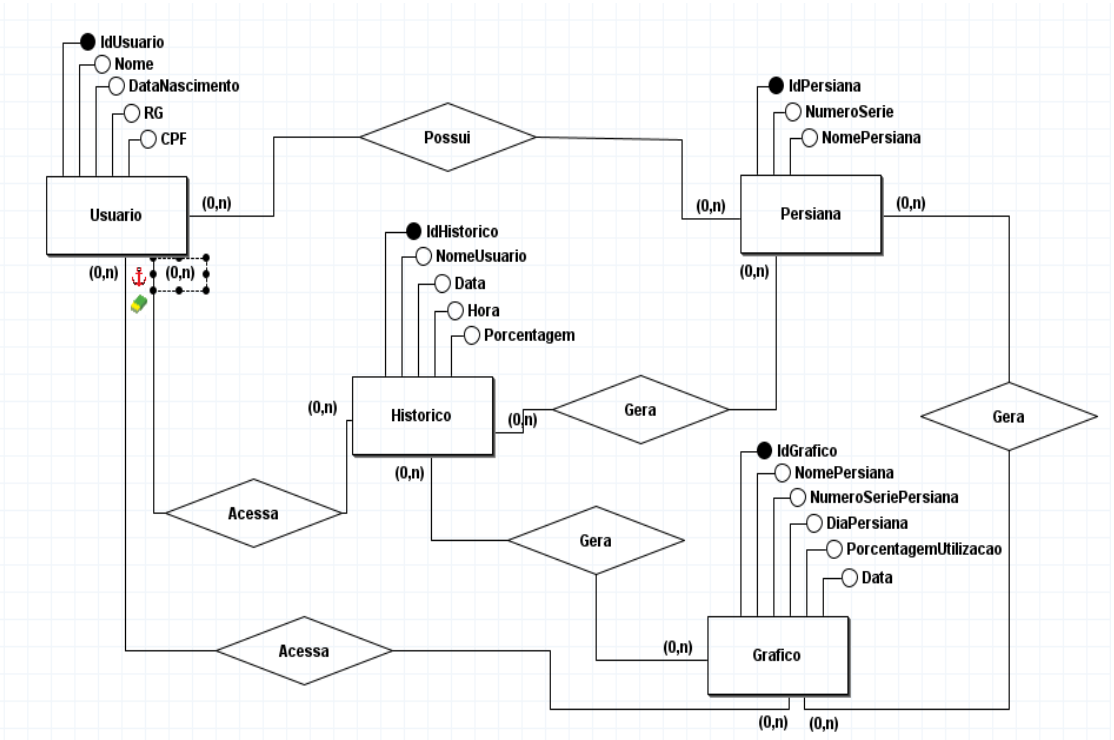
A tabela histórico Armazena atividades dos usuários como que horas a persiana foi utilizada e em que dia ou horário ela foi utilizada.

-Ex; Persiana 1 :utilizada a 3 dias

3.3.1 MODELO CONCEITUAL

A equipe deve entender o problema e propor uma solução para o banco de dados do sistema, demonstrando o modelo por meio de um DER (Diagrama de Entidade Relacionamento), devidamente documentado.

Para iniciarmos o nosso banco de dados precisamos fazer um esboço de como ele irá funcionar, para isso utilizamos o aplicativo BRModelo, que nos possibilita fazer um modelo conceitual para servir de base para o nosso banco.



No modelo conceitual nota-se todas as ligações e o que vai acessar o que, por exemplo, conforme o uso da persiana ela gera um histórico, a partir dele é gerado um gráfico com o usuário tendo acesso a eles e podendo se conectar usando o seu login.

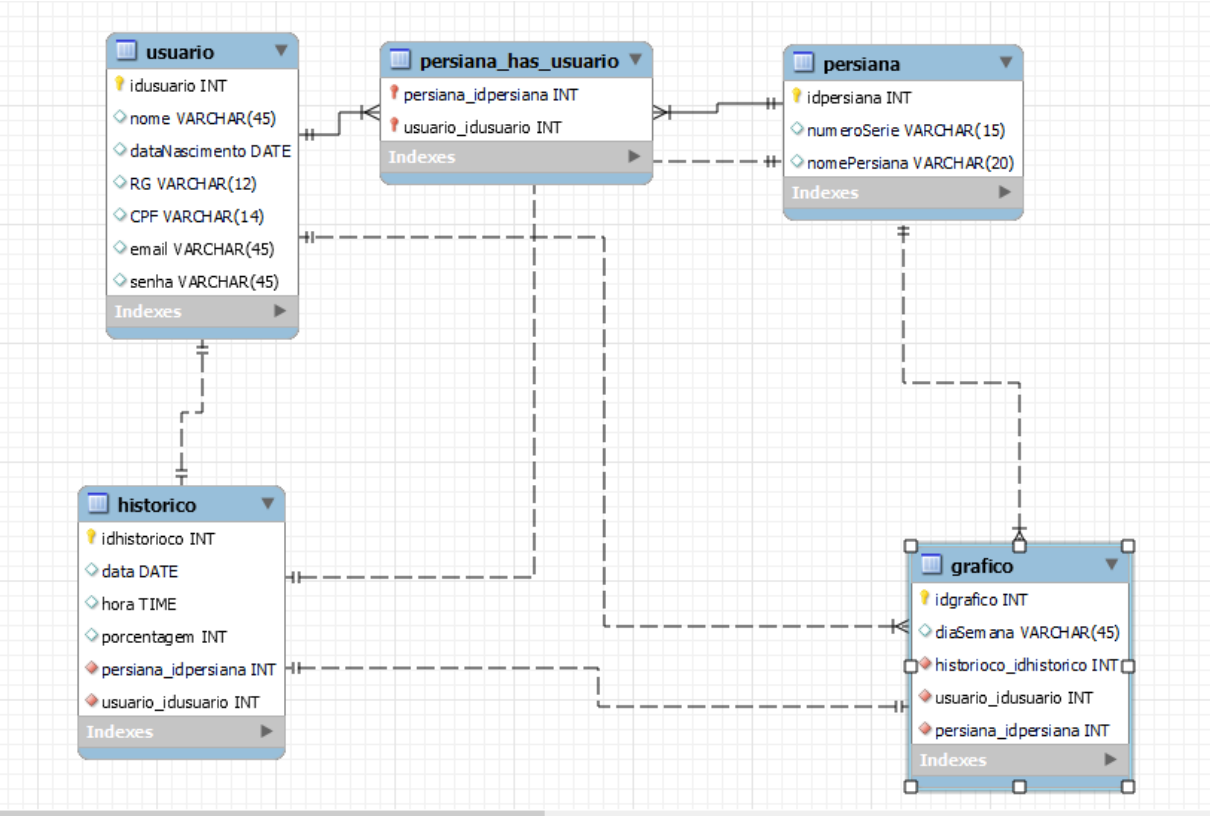
Depois de usarmos o BRModelo para montar o esboço, passamos para o MySQL e criamos um DER para criarmos as tabelas e suas conexões, para darmos início ao nosso banco de dados.

3.3.2 MODELO LÓGICO E FÍSICO

O SGBD utilizado por nós da ILight é o MySQL Workbench, por ser de mais fácil entendimento e por ser o aprendido nas aulas.

Imagens;

Modelo físico:



3.3.3 SQL

Inserts,Updates, Delete e selects utilizados por nós da ILight para testar o banco de dados. os inserts foram utilizados para inserir nome de teste a persiana e um nome de usuario no banco de dados

Insert:

```
INSERT INTO usuario (nome, email) VALUES ('João
Fernandez', 'JoãoFernandez01@gmail.com');


INSERT INTO persiana (nome) VALUES ('Persiana
Escritorio 1');


INSERT INTO historico (descricao) VALUES
('Persiana Utilizada Há 9min');

INSERT INTO grafico (historico_idhistorico,
usuario_idusuario, persiana_idpersiana) VALUES
(2301, 444, 999);
```

Update/Select e Delete:

Nós utilizamos o update de teste para alterar o nome do usuário na table do banco de dados, como podemos ver na imagem acima o nome era João e foi alterado para Luís e o nome da persiana foi alterada persiana escritorio e os códigos dos inserts foi alterado.

```
DELETE FROM usuario WHERE idusuario = 444;

DELETE FROM persiana WHERE idpersiana = 999;

DELETE FROM historico WHERE idhistorico = 2301;

DELETE FROM grafico WHERE historico_idhistorico
= 999;

UPDATE usuario SET nome = 'Luis Gabriel' WHERE
idusuario = 444;

UPDATE persiana SET nome = 'Persiana Escritório
3' WHERE idpersiana = 999;

UPDATE historico SET descricao = 'Persiana
Utilizada Há 9Dias' WHERE idhistorico = 2301;

UPDATE grafico SET usuario_idusuario = 444
WHERE historico_idhistorico = 777;

SELECT * from usuario;
SELECT * from perciana;
SELECT * from histotico;
SELECT * from grafico;
```

ISSN 1983-6767

3.4 GESTÃO FINANCEIRA

Com a crescente profissionalização na automatização de nossos produtos e móveis, adotamos uma abordagem cada vez mais estruturada no gerenciamento de valores e custos dentro de nossa empresa. A classificação e controle desses elementos desempenham um papel fundamental em nossa estratégia de negócios. Para atingir esse objetivo, baseamo-nos em uma análise minuciosa da nossa estrutura de custos e na comparação com o mercado atual, a fim de calcular preços de forma consistente e entrar no mercado de maneira competitiva. Neste contexto, destacamos os seguintes tópicos como base de nossa abordagem:

Benefícios das Persianas Automáticas:

- Economia de energia: Persianas automáticas podem ser programadas para otimizar o uso da luz natural e reduzir o consumo de energia.
- Conforto e privacidade: Permitem aos usuários ajustar facilmente o nível de luz e privacidade.
- Estilo e estética: São disponíveis em uma variedade de estilos, materiais e cores para se adequar à decoração.

Componentes das Persianas Automáticas:

- Motorização: O motor elétrico é responsável pelo movimento das persianas.
- Controle: Pode incluir controles remotos, interruptores de parede, aplicativos móveis e sistemas de automação residencial.
- Sensores: Alguns modelos podem ser equipados com sensores de luz, temperatura e vento para ajustar automaticamente as persianas.

Custos de Aquisição:

- Preço das persianas automáticas em si.
- Custos de instalação, incluindo mão de obra e materiais.

Custos Operacionais:

- Consumo de energia elétrica pelo motor das persianas.
- Custos de manutenção e reparo, incluindo limpeza e substituição de componentes.

Custos de Controle e Automação:

- Custos associados a sistemas de controle, como controles remotos ou sistemas de automação residencial.

Custos de Substituição:

- Custos futuros de substituição das persianas automáticas, à medida que envelhecem e se desgastam.

Custos de Integração:

- Custos de integração das persianas automáticas em sistemas de automação residencial.

Custos de Manutenção e Reparo:

- Custos recorrentes relacionados à manutenção e reparo do sistema motorizado.

https://conteudos.xpi.com.br/aprenda-a-investir/relatorios/markup-o-que-e-como-calcular/

3.4.1 CLASSIFICAÇÃO DOS CUSTOS

/-Os custos são todos os gastos que temos durante o processo de produção do produto ou serviços dentro de uma empresa, podendo ser classificados como custos direto, indiretos, fixos e variáveis.

- **Custo Direto:** São todos os custos diretamente relacionados ao produto, como matéria-prima do produto confeccionado;
- **Custo Indireto:** São aqueles que não tem uma ligação específica ao produtos, mas são muitos importantes para o funcionamento, como o salário do funcionário, aluguel, água e força;
- **Custos Fixos:** Eles não mudam independentemente se houve uma alta ou baixa produção no mês, ou seja, não importa a quantidade produzida se foi de 100 ou 1000 produtos, eles sempre vão ser os mesmos. Um bom exemplo são os salários, aluguel, água e limpeza;
- **Custos Variáveis:** Ao contrário dos custos fixos, os custos variáveis podem mudar de acordo com a produção da empresa, ou seja, caso a produção aumentar os valores aumentam, ou o inverso, caso a produção decaia os valores diminuem. Alguns exemplos são as matérias-primas e transporte.

Na tabela abaixo contém todos os custos da nossa empresa, classificados em custos diretos, indiretos, fixos e variáveis.

Gastos	Custos	Rateio de cada produto	
Arduino	209,90 R$	Arduino	514,25 R$

Modo bluetooth	53,53 R$	Modo bluetooth	131,14 R$
Sensor de luminosidade	49,90 R$	Sensor de luminosidade	122,25 R$
Domínio GoDaddy	19,99 R$	Domínio GoDaddy	48,75 R$
6 chave fim de curso	23,93 R$	6 chave fim de curso	58,62 R$
Motor vidro elétrico	30 R$	Motor vidro elétrico	73,50 R$
Persiana	190 R$	Persiana	465,5 R$
Força	700 R$	Força	
Internet	120 R$	Internet	
Água	500 R$	Água	
Servidor	100 R$	Servidor	

Total	Total Custos Indiretos	Total de Custos Diretos
1.997,87 R$	1.420 R$	577,87 R$

3.4.2 CUSTOS DO PRODUTO / SERVIÇO

Após uma análise minuciosa de custos relacionados à nossa empresa, chegamos à conclusão de que a implementação de apenas um projeto seria inviável, uma vez que seu custo total alcançaria R$1.997,87. Diante dessa realidade, decidimos adotar uma abordagem mais estratégica, alinhada com as dinâmicas do mercado. Optamos por desenvolver dois projetos distintos para atender às variadas necessidades de nossos clientes.

O primeiro projeto é a "Persiana Completa", uma solução totalmente automatizada e pronta para uso. Esta persiana vem equipada com nossa tecnologia integrada, eliminando a necessidade de configurações complexas por parte do usuário. Basta instalá-la no local desejado, baixar o aplicativo e começar a utilizá-la. Para aqueles que desejam acompanhar o histórico de uso em um desktop, oferecemos a opção de criar uma conta em nosso site. O preço competitivo para esta solução é de R$1.700,00, com a instalação incluída.

O segundo projeto é o "Motor Uni il 8", um motor de persianas vendido separadamente. Ele é projetado para qualquer tipo de persiana que suporte modificações. Com o Motor Uni il 8, os usuários podem desfrutar das mesmas funcionalidades de automação, embora algumas funções avançadas do aplicativo estejam ausentes. A funcionalidade básica

do aplicativo permite o controle das funções essenciais do motor "persiana". Este produto é precificado em R$1.200,00, incluindo o custo de instalação.

Para permanecer competitivos no mercado, definimos nossos preços de forma a estar em sintonia com as empresas especializadas do setor. Nossos valores variam entre R$1.200,00 e R$1.700,00, uma faixa que se alinha com os preços predominantes das empresas concorrentes, que normalmente oscilam entre R$1.500,00 e R$2.000,00.

Além disso, observamos que, no estágio atual como uma startup, conseguimos gerenciar a montagem dos produtos e o início do projeto com apenas uma pessoa. Entretanto, à medida que a demanda cresça, poderemos considerar a contratação de profissionais adicionais, se necessário, para atender às necessidades de expansão.

3.4.3 PRECIFICAÇÃO

Depois de definirmos o preço de nossos produtos precisamos calcular o markup e a margem de contribuição entre outros, e para isso precisamos entender o que é o markup e tudo isso.

Mas antes de começarmos a calcular precisamos saber alguns conceitos fundamentais para a formação dos custos:

- **Markup:** Percentual que uma empresa adiciona ao custo de um produto ou serviço para determinar o preço de venda. Ele é usado para cobrir os custos operacionais e gerar lucro.O resultado do cálculo do Markup é um número que indica quantas vezes o custo deve ser aumentado para se chegar ao preço de venda desejado ;
- **Margem de Contribuição:** Diferença entre o preço de venda de um produto ou serviço e os custos variáveis associados a ele. Os custos variáveis são aqueles que mudam de acordo com o volume de produção ou vendas, como matéria-prima, mão de obra variável e despesas diretas de produção.
- A Margem de Contribuição é uma medida importante para determinar a capacidade de um produto em contribuir para cobrir os custos fixos e gerar lucro;
- **Margem:** A Margem é uma medida que representa a diferença entre o preço de venda e o custo total de um produto ou serviço, levando em consideração tanto os custos variáveis como os custos fixos. Em outras palavras, a Margem é o lucro bruto obtido após a dedução de todos os custos associados ao produto;

- **Índice de Markup:** O Índice de Markup é a relação entre o preço de venda e o custo de um produto ou serviço. O Índice de Markup ajuda a determinar o quanto o preço de venda é maior do que o custo, mas não considera a estrutura de custos em detalhes.

Para calcularmos o Markup primeiro somamos a margem de contribuição que é o custo total menos o preço do produto:

- Custo Total - Preço = Margem de contribuição

Depois pegamos esse valor e dividimos pelo preço para acharmos a margem:

- Margem de contribuição / Preço

Em seguida calculamos o markup que a divisão da margem de contribuição pelo custo total multiplicado por 100, para acharmos a porcentagem:

- Margem de contribuição / Custo Total * 100 = Markup

E por último o Índice de Markup que é a divisão do Markup por 100 somando mais 1:

- Markup / 100 + 1 = Índice de Markup

Serviços	Custo Total	Preço	Margem de Contribuição	Margem	Markup	Índice de Markup
Persiana Completa	998,93 R$	1.700 R$	701,07 R$	41,24%	70,18%	1,70
Motor Uni il 8	887,93 R$	1.200 R$	321,07 R$	26,76%	36,53%	1,37

ISSN 1983-6767

3.5 CONTEÚDO DA FORMAÇÃO PARA A VIDA: GERENCIANDO FINANÇAS

Para essa unidade escolhemos fazer um vídeo intitulado como lidar com suas finanças e hábitos financeiros, abaixo deixamos o roteiro e algumas falas ditas no vídeo.

Introdução

- "Boa noite a todos! Hoje, estamos aqui para discutir algo essencial para nossa qualidade de vida: o cuidado com as finanças pessoais. Vamos entender como esse tema pode impactar positivamente nossas vidas."

Finanças Pessoais:

- "Vamos começar entendendo o básico. Finanças pessoais se resumem a um equilíbrio delicado entre o que ganhamos e o que gastamos. É crucial acompanhar nossos gastos e receitas."

- "Além disso, é importante diferenciar entre gastos essenciais e supérfluos. Estabelecer metas financeiras claras e construir uma reserva de emergência são passos fundamentais nessa jornada."

---

* Hábitos Financeiros Sustentáveis:*

- "Agora, vamos falar sobre o futuro. Planejamento a longo prazo, evitar dívidas e diversificar investimentos são estratégias essenciais para garantir estabilidade financeira."

- "Não podemos esquecer da educação financeira contínua e da criação de um fundo de aposentadoria. É fundamental adaptar nossos planos conforme nossa situação financeira evolui."

Crescendo Financeiramente e Contribuindo para a Comunidade:

- "Falando em evolução, o crescimento financeiro a longo prazo é alcançável. A busca por novas oportunidades e investimentos é uma estratégia inteligente."

- "Além disso, é importante considerar a responsabilidade social. Contribuir para a comunidade e praticar a generosidade são aspectos valiosos do sucesso financeiro."

Conclusão:

- "Em resumo, cuidar de nossas finanças é um processo contínuo. Esperamos que essas dicas ajudem a construir um caminho mais seguro e próspero. Estamos abertos para perguntas e discussões. Muito obrigado!"

3.5.1 GERENCIANDO FINANÇAS

- **Tópico 1:** Introdução aos conceitos econômicos e financeiros básicos

  Em suma a ILight utilizou de conceitos económicos e financeiros para realizar a precificação de seu produto, utilizamos de custos diretos que está diretamente relacionado com a produção de bens ou serviços, como materiais, mão de obra direta e despesas específicas na produção, e juntamente com os custos indiretos que não está diretamente vinculado com a produção, mas auxilia para a operação geral da empresa, com essas informações utilizamos da ferramenta de controle através de planilha nomeado como excel para realizar nosso controle.

- **Tópico 2:** Entendendo o ambiente: independência financeira, o valor da minha riqueza e o registro do dia a dia

  Para independência financeira a ILight utiliza de renda passiva que ocorre através de investimentos, como tesouro direto que tem uma taxa de rendimento de 10,96% anualmente e parcerias estratégicas e juntamente com o software de controle com planilhas excel, controlamos os valores que entram em nossa empresa e geram lucros e que geram custos diário assim evitando problemas que podem ocorrer posteriormente.

- **Tópico 3:** Dívidas e juros compostos, opções de empréstimo e alternativas ao endividado

  A ILight tem como custos inicial 1.997,87 reais e com um investimento inicial de 10.000 reais podemos manter a empresa estabilizada por um período de tempo mesmo que não ocorra nenhuma venda, para caso de dividas que possam ocorrer posteriormente, entraremos com um empréstimo que será realizado no banco do brasil o valor não é definido pois depende do valor da dívida, entraremos juntamente com uma negociação com o credor.

- **Tópico 4:** Estabelecer metas para a realização de seus sonhos e como envolver o grupo a que você pertence para atingir seus objetivos

  A ILight tem como meta inicial entrar no mercado de automação residencial e a partir de uma iniciação estável expandir com produtos para atender fábricas e comércios locais, com a evolução do mercado pretendemos expandir nossa empresa tornando-a uma empresa de grande porte e posteriormente uma empresa global que possa atender demandas mundialmente expandido assim nosso site, comércio, gerenciamento de custos e despesas e valores de lucros, para alcançar nossos objetivos realizaremos parcerias com fabricantes e anunciantes e mantendo nossos valores para enfim chegarmos em nosso sonho de ser uma multinacional conhecida por sua transparência com seus clientes e boa conduta de nossos trabalhadores por parte de implantação de nossos produtos.

A síntese precisa apresentar exemplos práticos dos seus conteúdos, ou seja, de modo que possam ser utilizados ou verificados no dia-a-dia.

ISSN 1983-6767

3.5.2 ESTUDANTES NA PRÁTICA

Para explicar o conteúdo em questão realizamos um vídeo pontuando alguns tópicos importantes para os usuários lidarem melhor com as suas finanças e com sua vida em geral. o vídeo intitulado como “Finanças Pessoais”, video foi gravado por três integrantes da ILight.LTDA e foi postado no youtube na intenção de atingir o máximo de pessoas possível com o intuito de apresentar e ajudar pessoas mesmo se elas ainda não estiverem na fase de lidar com suas finanças.

4. CONCLUSÃO

A criação e execução do nosso projeto chamado iLight ocorreu graças ao esforço, comprometimento e cooperação de todos os membros do grupo em suas determinadas funções, tanto na parte de programação quanto no design e na gestão de gastos, e com isso conseguimos desenvolver um sistema que não só atende às necessidades do usuário, mas também simplifica a vida cotidiana com um produto inovador, prático e com objetivo claro.

O projeto, focado em persianas automáticas, representa uma inovação significativa ao proporcionar conveniência e praticidade no controle de iluminação e privacidade nas residências e estamos confiantes de que o ILight se destacará no mercado e proporcionará uma experiência de usuário excepcional, combinando eficiência e simplicidade.

ISSN 1983-6767

REFERÊNCIAS

Gestão:

https://conteudos.xpi.com.br/aprenda-a-investir/relatorios/markup-o-que-e-como-calcular/

Introdução:

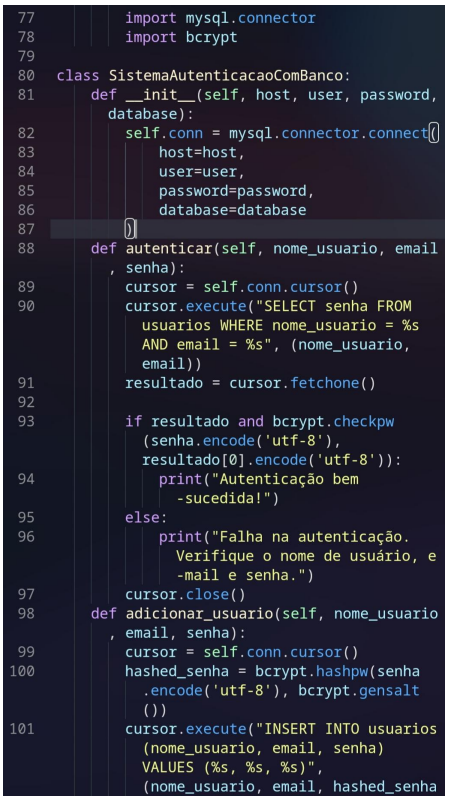
MURATORI, José Roberto; DAL BÓ, Paulo Henrique. Capítulo I Automação residencial: histórico, definições e conceitos. **O Setor elétrico**, p. 70-77, 2011.

FERNANDES, Flávia Gonçalves et al. Sistema de automação residencial controlado por dispositivos móveis e vestíveis. **FTT Journal of Engineering and Business**, n. 1, 2016.

ISSN 1983-6767

ANEXOS

anexo código 1:



```
        import mysql.connector
        import bcrypt

class SistemaAutenticacaoComBanco:
    def __init__(self, host, user, password,
      database):
        self.conn = mysql.connector.connect(
            host=host,
            user=user,
            password=password,
            database=database
        )
    def autenticar(self, nome_usuario, email
      , senha):
        cursor = self.conn.cursor()
        cursor.execute("SELECT senha FROM
          usuarios WHERE nome_usuario = %s
          AND email = %s", (nome_usuario,
          email))
        resultado = cursor.fetchone()

        if resultado and bcrypt.checkpw
          (senha.encode('utf-8'),
          resultado[0].encode('utf-8')):
            print("Autenticação bem
              -sucedida!")
        else:
            print("Falha na autenticação.
              Verifique o nome de usuário, e
              -mail e senha.")
        cursor.close()
    def adicionar_usuario(self, nome_usuario
      , email, senha):
        cursor = self.conn.cursor()
        hashed_senha = bcrypt.hashpw(senha
          .encode('utf-8'), bcrypt.gensalt
          ())
        cursor.execute("INSERT INTO usuarios
          (nome_usuario, email, senha)
          VALUES (%s, %s, %s)",
          (nome_usuario, email, hashed_senha
```