Documentação da Criptografia RSA no Código da API Flask para Previsão de Preço de Corridas

1. Visão Geral do Projeto Esta API, desenvolvida com Flask, tem como objetivo prever o preço de uma corrida entre dois endereços, utilizando um modelo de machine learning treinado previamente. Além disso, a API realiza criptografia dos dados pessoais (nome e e-mail) do usuário usando o algoritmo RSA antes de armazená-los em um banco de dados MySQL hospedado na nuvem.

2. O que é a Criptografia RSA RSA (Rivest-Shamir-Adleman) é um dos algoritmos de criptografia assimétrica mais utilizados no mundo. Em criptografia assimétrica, são usadas duas chaves:

- **Chave pública**: usada para criptografar os dados.
- **Chave privada**: usada para descriptografar os dados.

A segurança do RSA baseia-se na dificuldade de fatorar grandes números primos. Mesmo com a chave pública, é computacionalmente inviável recuperar os dados sem a chave privada.

3. Como a Criptografia RSA é Usada neste Código No código, a função criptografar_rsa(mensagem) é responsável por criptografar as informações pessoais do usuário antes de salvá-las no banco de dados:

```
from cryptography.hazmat.primitives.asymmetric import padding
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives import serialization
import base64

with open("public_key.pem", "rb") as f:
    public_key = serialization.load_pem_public_key(f.read())

def criptografar_rsa(mensagem):
    criptografado = public_key.encrypt(
        mensagem.encode("utf-8"),
        padding.OAEP(
            mgf=padding.MGF1(algorithm=hashes.SHA256()),
            algorithm=hashes.SHA256(),
            label=None
        )
    )
    return base64.b64encode(criptografado).decode("utf-8")
```

4. Detalhamento do Processo

1. A chave pública é carregada do arquivo public_key.pem.
2. A mensagem (nome ou e-mail) é convertida para bytes.
3. É aplicado o padding OAEP (Optimal Asymmetric Encryption Padding) com SHA-256, que é uma forma segura de preenchimento.
4. O resultado é convertido em Base64 para facilitar o armazenamento como string no banco de dados.

5. Ponto de Uso da Criptografia na API A criptografia acontece dentro da rota /api/prever antes do salvamento no banco:

nome = criptografar_rsa(nome)

email = criptografar_rsa(email)

6. Motivação para Uso de Criptografia A criptografia RSA é utilizada para garantir a segurança e privacidade dos dados pessoais dos usuários. Mesmo que o banco de dados seja comprometido, os dados armazenados estarão ilegíveis sem a chave privada correspondente.

7. Considerações de Segurança

- A chave pública pode ser distribuída livremente, mas a chave privada deve ser protegida rigorosamente.
- Em ambientes de produção, é recomendado o uso de módulos de hardware seguros (HSM) ou gestão segura de chaves com cofres como o Azure Key Vault, AWS KMS ou HashiCorp Vault.

8. Conclusão O uso de criptografia RSA no projeto demonstra a preocupação com a segurança da informação. Com esse mecanismo, a aplicação garante que dados sensíveis dos usuários sejam armazenados de forma protegida, alinhando-se com boas práticas de segurança da informação e conformidade com legislações como a LGPD.