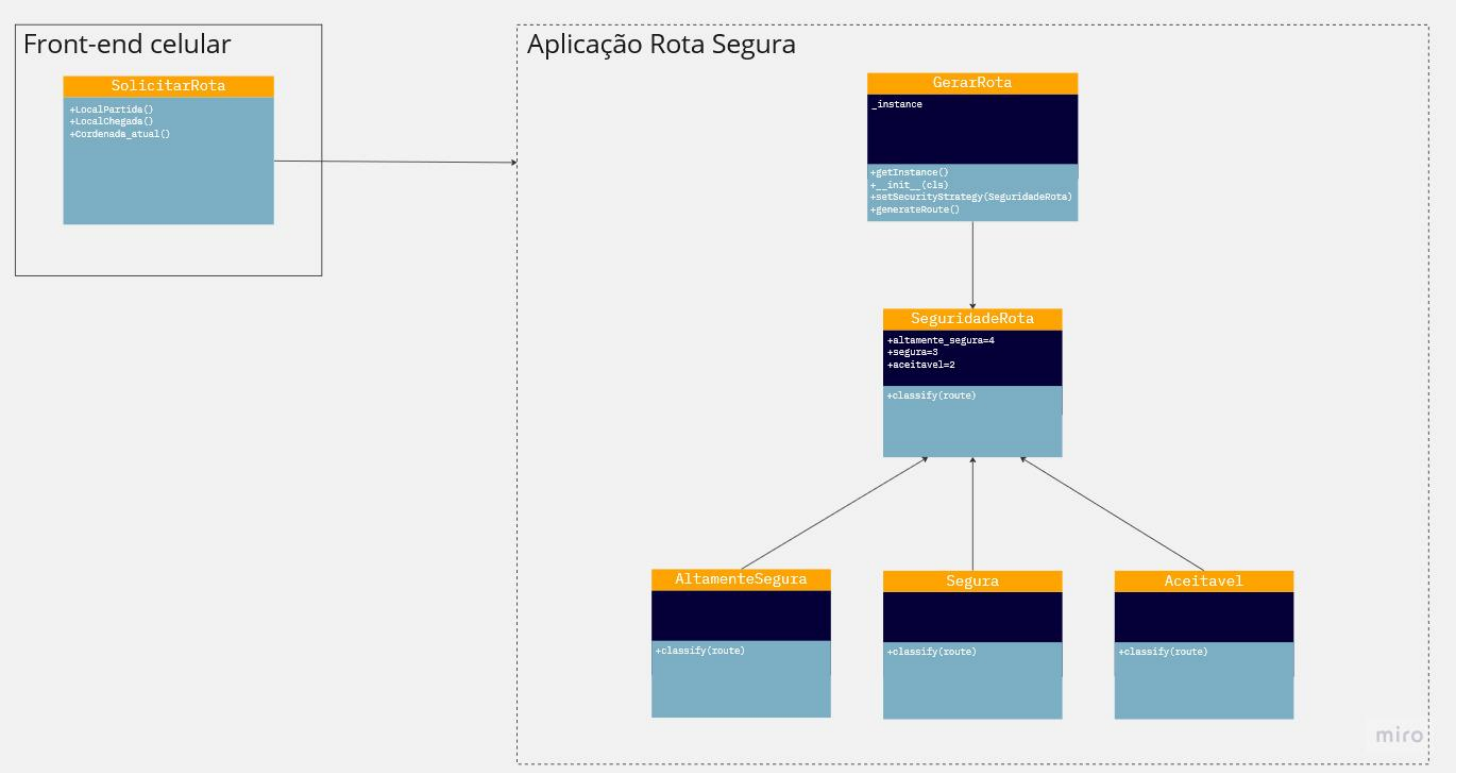
Diagrama UML



Pseudo codigo python

Considerando que a função RoutGeneration devolve uma nota de 0 a 3, sendo 0 = insegura.

```
from typing import List

class SegurancaRota:
      ALTISSIMA_SEGURANCA = 3
      SEGURA = 2
      ACEITAVEL = 1

class EstrategiaRota:
      def calculo_nivel_seguranca(self, route: List[str]) -> int:
      pass

class RotaAltamenteSegura(EstrategiaRota):
      def calculo_nivel_seguranca(self, route: List[str]) -> int:
      # Lógica para calcular a segurança de uma rota altamente segura
      pass
class RotaSegura(EstrategiaRota):
      def calculo_nivel_seguranca(self, route: List[str]) -> int:
      # Lógica para calcular a segurança de uma rota segura
```

```
      pass

class RotaAceitavel(EstrategiaRota):
      def calculo_nivel_seguranca(self, route: List[str]) -> int:
      # Lógica para calcular a segurança de uma rota aceitável
      pass

class GeradorRota:
      _instance = None

      def __init__(self):
      if self._instance is not None:
            raise ValueError()
      self._nivel_seguranca_rota = SegurancaRota.ACEITAVEL
      self._route_strategy = RotaAceitavel()

      @classmethod
      def getInstance(cls):
      if cls._instance is None:
            cls._instance = GeradorRota()
      return cls._instance

      def set_nivel_seguranca(self, level: int):
      self._nivel_seguranca_rota = level


      if level == SegurancaRota.ALTISSIMA_SEGURANCA:
            self._route_strategy = RotaAltamenteSegura()
      elif level == SegurancaRota.SEGURA:
            self._route_strategy = RotaSegura()
      elif level == SegurancaRota.ACEITAVEL:
            self._route_strategy = RotaAceitavel()

      def gerar_rota(self, origem: str, destino: str) -> List[str]:
      # Lógica para gerar a rota
      pass
```