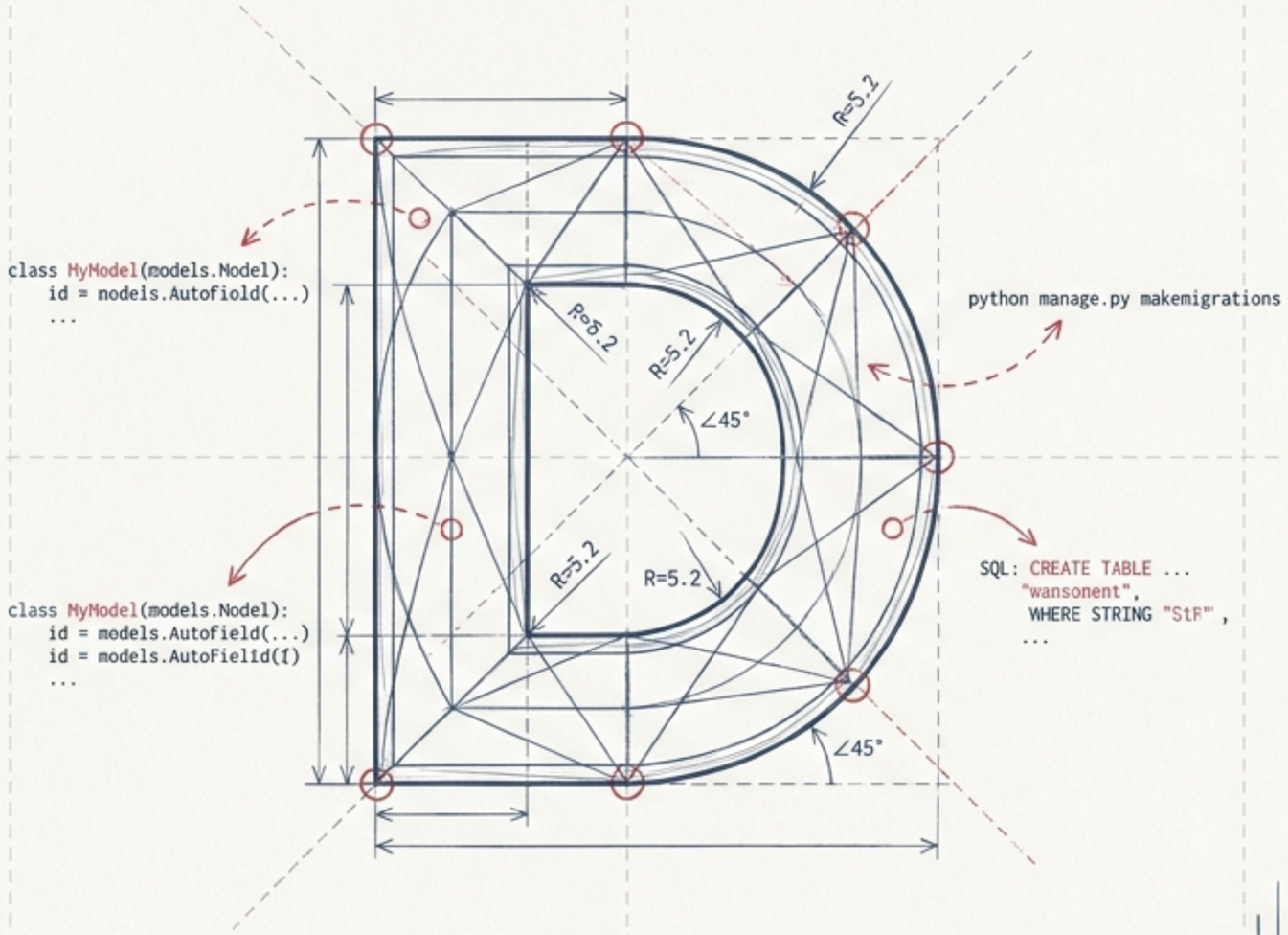
O Raio-X do seu Primeiro Projeto Django

Como uma classe vira tabela e o que o framework faz por você (sem mágica, pura arquitetura).



Foco em estruturar modelos mentais sólidos para iniciantes.

Gemini Notebook

A Metáfora Base: O Restaurante e seus Departamentos

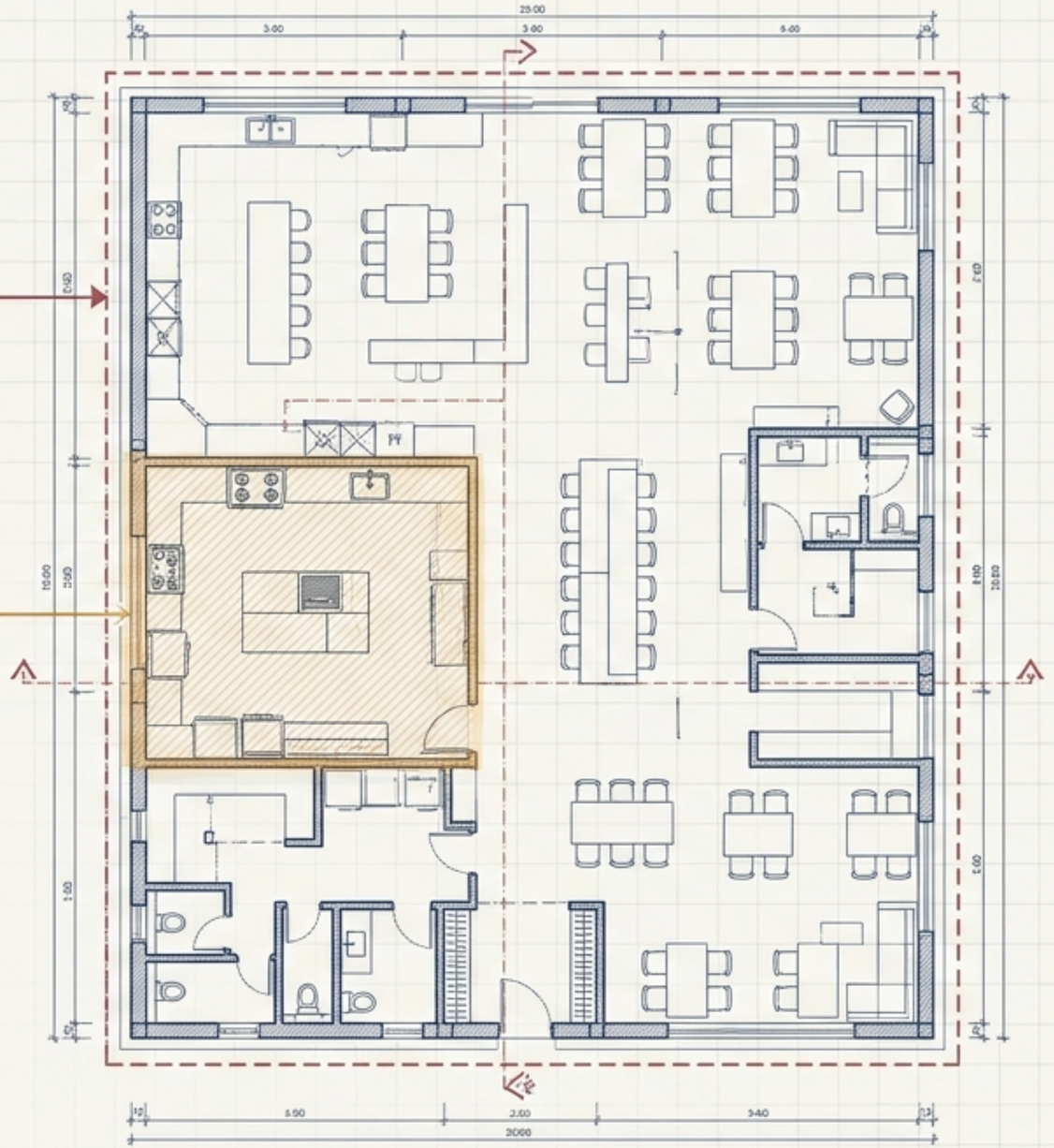
Uma antarzona conexinam a metáfora descirante e restaurantes rzm no projeto cranam o ponnoizis.

O Projeto (minimarket)

É o restaurante inteiro. Contém o alvará, o endereço e as regras gerais da casa.

O App (catalogo)

É um departamento específico (ex: cozinha quente, confeitaria). Um projeto real tem vários apps, cada um cuidando de um assunto só.



Departamentos bem separados podem ser reaproveitados e testados cada um por si.

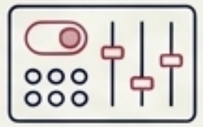
Gemini Notebook

Matriz Estrutural: Projeto × App

Dimensão	Projeto (minimarket)	App (catalogo)
Metáfora	O Restaurante inteiro	Um departamento isolado
Comando de Criação	`python -m django startproject minimarket .`	`python manage.py startapp catalogo`
Arquivo Principal	settings.py (Regras da casa)	models.py (Fichas das prateleiras)
Quantidade no Sistema	Apenas 1 (ponto de entrada)	Vários (um para cada domínio)

Gemini Notebook

O Ecossistema de Arquivos (A Planta da Cozinha)

`manage.py` → 

Painel de comando (recebe ordens: runserver, migrate).

`settings.py` → 

Quadro de regras (banco de dados, segurança).

`urls.py` → 

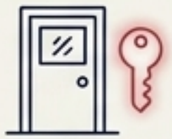
Garçom-chefe (tabela de rotas: endereço → quem atende).

`models.py` → 

Fichas das prateleiras (estrutura das tabelas).

`views.py` → 

O cozinheiro (código que atende os pedidos).

`admin.py` → 

Sala do gerente (o que aparece no painel de gestão).

`migrations/` → 

Plantas de reforma (histórico de mudanças na despensa).

Gemini Notebook

`settings.py`: O Quadro de Regras da Casa

settings.py

```
DEBUG = True

ALLOWED_HOSTS = []

INSTALLED_APPS = [ ... ]

DATABASES = {
    'default': { ...
    }
}
```

Perigo na Internet!

Mostra código e configurações no navegador se algo der errado. Ótimo para você, desastroso em produção. **Sempre False em produção.**



O Carimbo de Segurança

Usado para assinar sessões e senhas. Nunca envie para o Git; em produção, deve vir de uma variável de ambiente.



A Despensa

O padrão é SQLite (um único arquivo, excelente para estudo).

Gemini Notebook

A Conexão Vital: Registrando o Departamento

```
INSTALLED_APPS = [
    'django.contrib.admin',
    ...,
    'catalogo', # ← A sua linha vital
]
```



Armadilha Clássica do Iniciante

O Django não adivinha que o seu app existe. Sem essa linha, comandos como makemigrations respondem ‘No changes detected’, mesmo que seu código esteja perfeito.

Gemini Notebook

O Primeiro ‘Ao Vivo’: Servidor de Desenvolvimento

```
python manage.py runserver
```

http://127.0.0.1:8000/



runserver

O **‘fogão de bancada’.** Recarrega sozinho ao salvar arquivos. Nunca usado em produção.

127.0.0.1

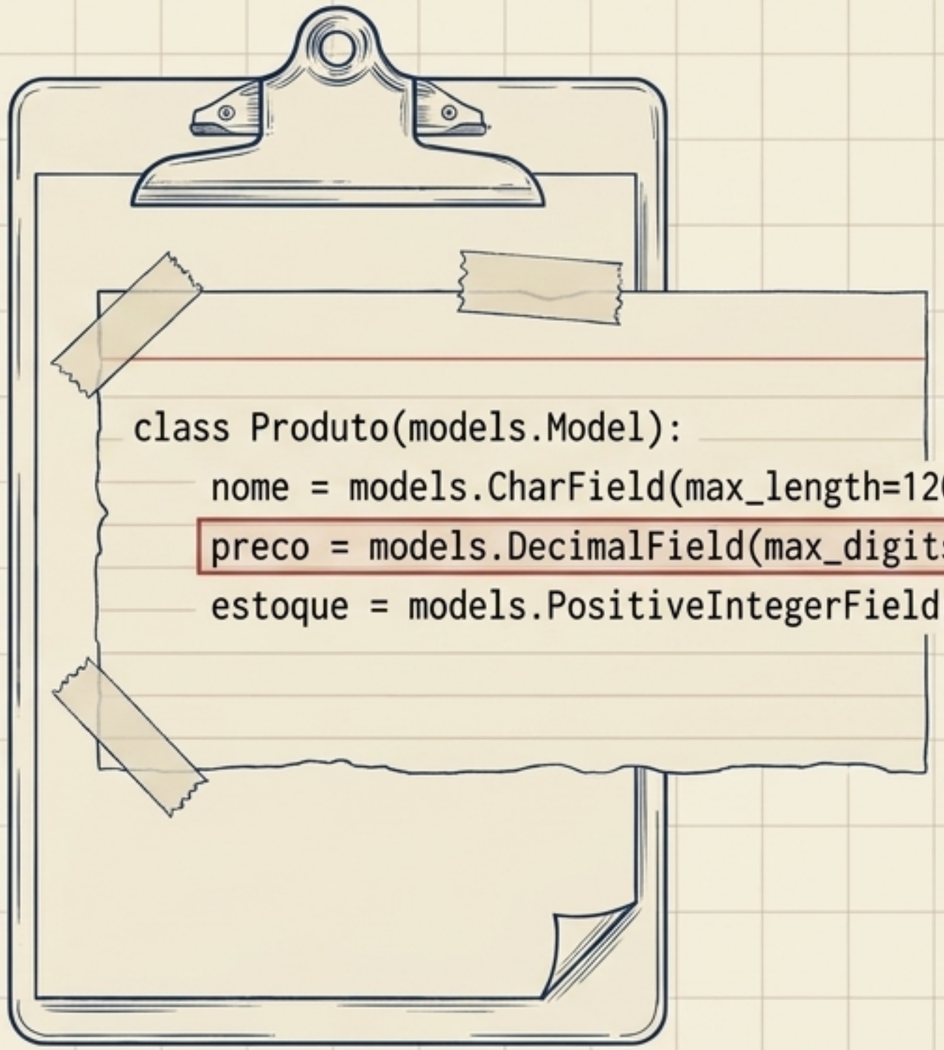
‘Este computador’. Apenas a sua máquina acessa.

8000

A ‘porta’. A janela de atendimento virtual por onde o tráfego flui.

Gemini Notebook

Models: A Ficha da Prateleira (Regras de Negócio)



```
class Produto(models.Model):
    nome = models.CharField(max_length=120)
    preco = models.DecimalField(max_digits=10, decimal_places=2)
    estoque = models.PositiveIntegerField(default=0)
```

`CharField(120)`

Limite explícito. Evita o armazenamento de textos gigantes indesejados.

`DecimalField`

Nunca use Float para dinheiro!
O Float erra centavos. O Decimal garante precisão exata.

`PositiveIntegerField`

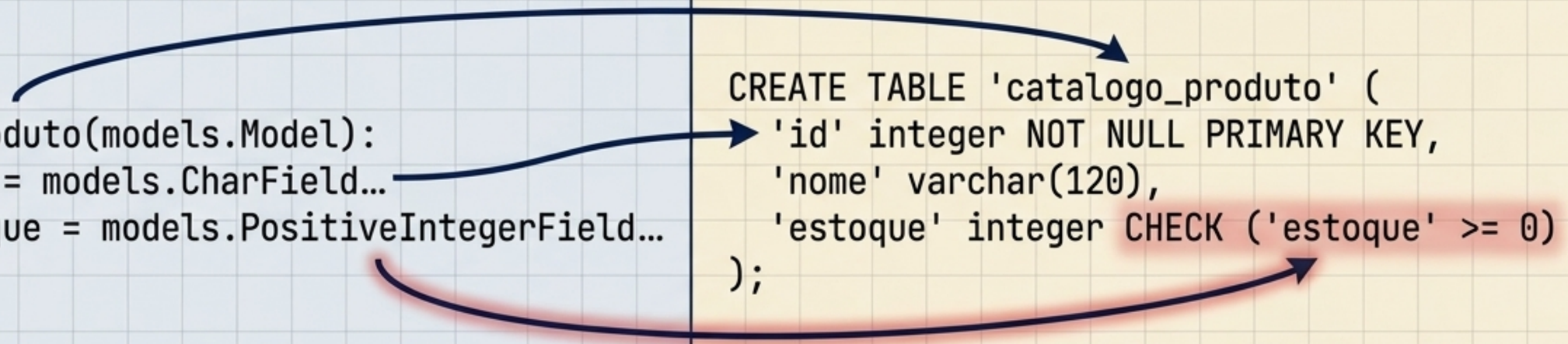
Garante que estoque negativo não exista (a regra é imposta no banco).

Gemini Notebook

A Mágica Traduzida (Python → SQL)

Python	SQL

`class Produto(models.Model):`
`  nome = models.CharField…`
`  estoque = models.PositiveIntegerField…`
	`CREATE TABLE 'catalogo_produto' (`
	`  'id' integer NOT NULL PRIMARY KEY,`
	`  'nome' varchar(120),`
	`  'estoque' integer CHECK ('estoque' >= 0)`
	`);`



Hidden Feature

Você não escreveu `id`, mas o Django gerou `id integer NOT NULL PRIMARY KEY AUTOINCREMENT`.

Table Naming

O nome da tabela gerada é `catalogo_produto` (Padrão: `app_modelo`).

The Database Lock

O `PositiveIntegerField` no Python virou `CHECK ('estoque' >= 0)` no SQL. O próprio banco agora recusa números negativos. Dupla trava de segurança!

Gemini Notebook

O Ciclo das Migrações: Por que duas etapas?

1. O Design



`models.py`

Você altera o código Python. A ideia existe, mas a obra não começou.

2. A Planta da Obra



`makemigrations`

O Django compara seu código com o histórico e escreve um arquivo Python de migração. Vai para o Git.

3. A Obra Executada



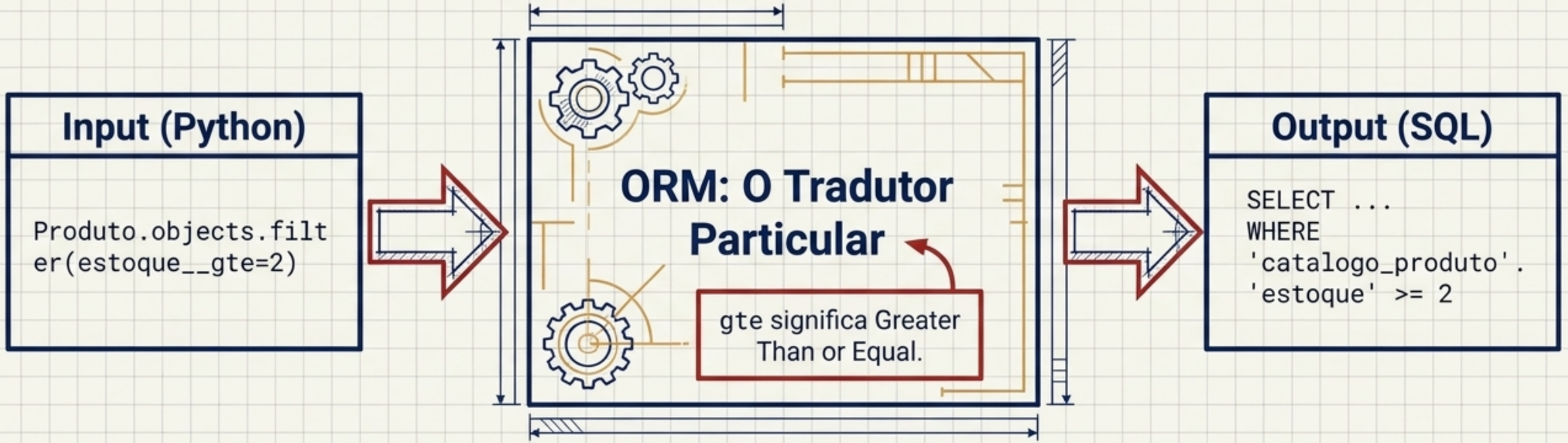
`migrate`

O Django pega as plantas pendentes e aplica de fato no banco de dados. A tabela é criada.

Gemini Notebook

Falando com a Despensa: O Tradutor ORM

`python manage.py shell`



Pro-Tip

Adicione `.query` ao final do seu filtro no shell para ver exatamente o SQL que o ORM vai rodar. **Se o sistema ficar lento um dia**, é aqui que você investiga.

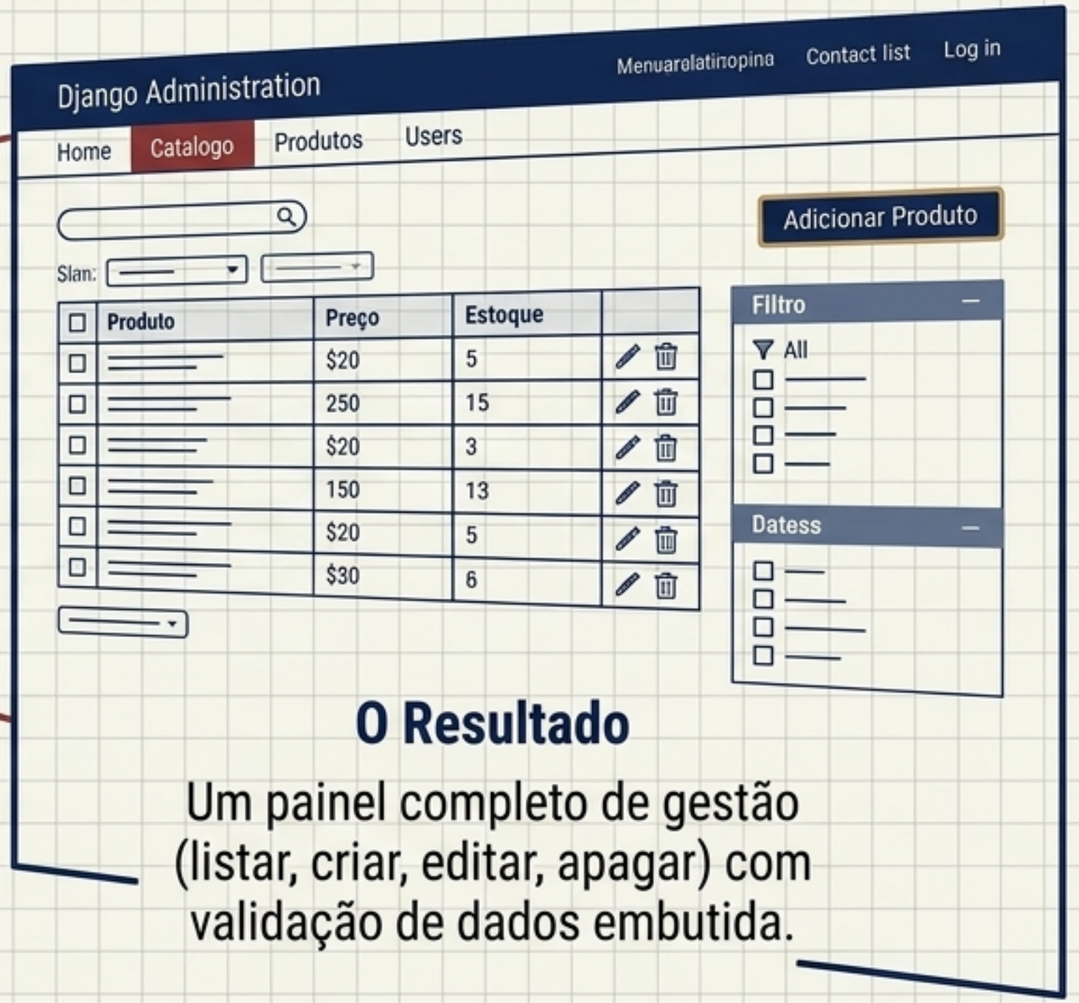
Gemini Notebook

O Painel Admin: “Baterias Incluídas”

O Código (catalogo/admin.py)

```
from django.contrib import admin
from .models import Produto

admin.site.register(Produto)
```



O Resultado

Um painel completo de gestão (listar, criar, editar, apagar) com validação de dados embutida.

Comando Necessário



`python manage.py createsuperuser` (cria as chaves de acesso do gerente).

Gemini Notebook

Matriz de Diagnóstico: Os 3 Erros Clássicos

Laboratório de Erros		
O Sintoma	**A Causa**	**A Solução**
`makemigrations` diz `"No changes detected"`.	O app esquecido.	Registrar o app em `INSTALLED_APPS`.
`OperationalError: no such column.`	A migração que falta. Mudou o código, mas a despensa não foi avisada.	Rodar `makemigrations` e depois `migrate`.
`Model isn't in an application in INSTALLED_APPS.`	Importar sem registrar.	Ler a primeira linha do erro! O Django te dá a resposta exata.

Gemini Notebook

Síntese: O Caminho da Informação

Step 1: O Alvará



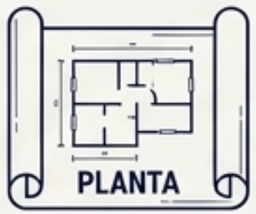
Registrar o app no `INSTALLED_APPS` (`settings.py`).

Step 2: A Ficha



Escrever a classe no `models.py`.

Step 3: A Planta



Rodar `makemigrations` (O framework gera a estrutura).

Step 4: A Obra



Rodar `migrate` (O banco `SQLite` cria a tabela).

Step 5: A Vitrine



Cadastrar no `admin.py` e ver a mágica operando no navegador.

Gemini Notebook

Fichas de Revisão Rápida

Projeto × App?

Restaurante inteiro × Um departamento isolado.

`DEBUG = True`?

Apenas desenvolvimento. Revela segredos em produção.

`makemigrations` × `migrate`?

Escrever a planta × Executar a obra no banco.

O que você dominou

A arquitetura web de um sistema monolítico.

Próximo Passo

Transformar esse catálogo visual em uma API de dados bruta usando Django Rest Framework (DRF).

Gemini Notebook