Структура Базы данных  
  
  
class User(AbstractUser):  
 groups = models.ManyToManyField(  
 'users.Group',  
 related\_name='custom\_user\_set',  
 blank=True,  
 help\_text='The groups this user belongs to.',  
 verbose\_name='groups',  
 )  
  
 user\_permissions = models.ManyToManyField(  
 'users.Permission',  
 related\_name='custom\_user\_permissions',  
 blank=True,  
 help\_text='Specific permissions for this user.',  
 verbose\_name='user permissions',  
 )  
  
 first\_name = models.CharField(max\_length=50, verbose\_name="Имя")  
 phone = models.CharField(max\_length=13, unique=True, null=False, verbose\_name='Номер телефона')  
 user\_phone = models.CharField(max\_length=13, unique=True, blank=True, null=True, verbose\_name='Номер телефон пользователя')  
 email = models.EmailField(unique=True, null=True, blank=True)  
 code = models.CharField(max\_length=50, null=True, verbose\_name='Код', blank=True)  
 last\_sms\_date = models.DateTimeField(null=True, verbose\_name='Дата отправки кода', blank=True)  
 is\_confirm = models.BooleanField(default=False, blank=True, verbose\_name='Подтверждение почты')  
 created\_at = models.DateField(auto\_now\_add=True, verbose\_name='Дата создание')  
 objects = UserManager()  
 last\_name = None  
 username = None  
  
 USERNAME\_FIELD = "phone"  
 REQUIRED\_FIELDS = []  
  
 def \_\_str\_\_(self):  
 return str(self.phone)  
  
 class Meta:  
 verbose\_name = 'Пользователь'  
 verbose\_name\_plural = 'Пользователи'  
  
  
class Contract(models.Model):  
 client\_name = models.CharField(max\_length=255)  
 service\_description = models.TextField()  
 date\_created = models.DateField(auto\_now\_add=True)  
 total\_price = models.DecimalField(max\_digits=10, decimal\_places=2)  
  
 def \_\_str\_\_(self):  
 return f"Договор для {self.client\_name}"

Интеграция с Лактекс  
  
import subprocess  
from jinja2 import Environment, FileSystemLoader  
from pathlib import Path  
import tempfile  
  
def render\_contract\_to\_pdf(context, template\_name):  
 env = Environment(loader=FileSystemLoader('document/templates/latex'))  
 template = env.get\_template(template\_name)  
 rendered\_tex = template.render(context)  
  
 with tempfile.TemporaryDirectory() as tmpdir:  
 tex\_path = Path(tmpdir) / "document.tex"  
 pdf\_path = Path(tmpdir) / "document.pdf"  
  
 tex\_path.write\_text(rendered\_tex, encoding="utf-8")  
  
 process = subprocess.run(  
 ["pdflatex", "-interaction=nonstopmode", str(tex\_path)],  
 cwd=tmpdir,  
 stdout=subprocess.PIPE,  
 stderr=subprocess.PIPE,  
 text=True,  
 encoding="utf-8",  
 errors="replace"  
 )  
  
 print("XELATEX STDOUT:\n", process.stdout)  
 print("XELATEX STDERR:\n", process.stderr)  
  
 if process.returncode != 0:  
 raise RuntimeError(f"xelatex failed with code {process.returncode}")  
  
 if pdf\_path.exists():  
 return pdf\_path.read\_bytes()  
  
 return None

Бизнес логика  
  
from django.shortcuts import render  
from django.http import HttpResponse  
from .models import Contract  
from .latex\_renderer import render\_contract\_to\_pdf  
  
def generate\_contract(request, contract\_id):  
 contract = Contract.objects.get(id=contract\_id)  
 context = {  
 "client\_name": contract.client\_name,  
 "service\_description": contract.service\_description,  
 "date\_created": contract.date\_created,  
 "total\_price": contract.total\_price,  
 }  
 pdf\_bytes = render\_contract\_to\_pdf(context, "contract\_template.tex.j2")  
  
 if pdf\_bytes:  
 response = HttpResponse(pdf\_bytes, content\_type='application/pdf')  
 response['Content-Disposition'] = f'attachment; filename=contract\_{contract.id}.pdf'  
 return response  
  
 return HttpResponse("Ошибка при генерации PDF", status=500)

Конфигурация программы Латекс  
  
\documentclass[12pt]{article}  
\usepackage[utf8]{inputenc}  
\usepackage[T1]{fontenc}  
  
\begin{document}  
  
\section\*{Service Contract}  
  
Client: \textbf{ {{ client\_name }} }  
  
Service description:  
  
{{ service\_description }}  
  
Date created: {{ date\_created }}  
  
Total to pay: \textbf{ {{ total\_price }} } som  
  
\end{document}