Розділ 3. Розробка веб-застосунку «Інструментарій підприємця - початківця» з використанням технологій Django

3.1. Про актуальність проблеми та етапи розробки ППЗ

Актуальність проблеми: У сучасному бізнесі багато підприємців-новачків стикаються з низкою складнощів при управлінні своїм підприємством. Вони потребують інструменти, які допоможуть їм ефективно управляти клієнтами, співробітниками, товарами та замовленнями, а також вирішувати завдання управління запасами. Однак, багато з доступних на ринку програмних продуктів не задовольняють їхні потреби або мають високу вартість, що робить їх недоступними для малого бізнесу. Тож створення прикладного програмного забезпечення "Інструментарій підприємця-початківця" буде дуже актуальним, він зможе надавати підприємцям-новачкам необхідні інструменти для ефективного управління своїм підприємством.

Етапи розробки ППЗ такі:

1. Спочатку потрібно створити всі необхідні бази даних, де буде вказана вся необхідна інформація стосовно певного об’єкта.
2. Прив’язка цих БД до django та створення можливості їх заповнення через адміністративний інтерфейс.
3. Створення доступу до адміністративного інтерфейсу та заповнення БД необхідними даними.
4. Підключення html файлу для виведення необхідної інформації у веб-сторінку.
5. Передача необхідних даних до html файлу та створення програмного коду мовою html, щоб вивести потрібні дані.
6. Вирішення прикладних задач управління запасом товарів та виведення в окрему веб-сторінку.

3.2. Проєктування структури інтерфейсу користувча

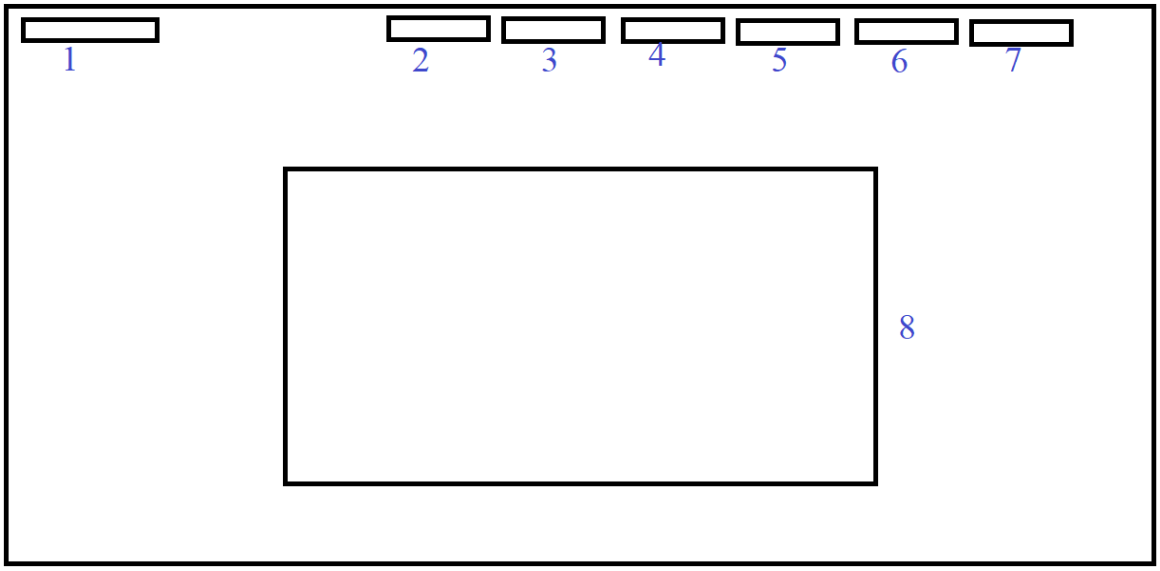

Рис.3.2.1. Макет ППЗ

На макеті ППЗ можемо побачити числа від 1 до 8. Біля числа 1 буде назва завдання. Там де стоїть 2 створю сторінку де буде посилання на сайт податкової служби. Натиснувши на це посилання вас перекине на сайт де знаходиться податковий кодекс.

Біля числа 3 зроблю посилання на сторінку, де я виведу список всіх клієнтів, що знаходяться у БД клієнтів.

Біля 4 буде знаходитися посилання на сторінку де буде інформація про усіх працівників. Перейшовши на цю сторінку, можна буде переглянути список усіх працівників та їх заробітню плату.

Над числом 5 буде знаходитися посилання на створену сторінку, яка буде називатися «Товари». Зайшовши на цю сторінку, підприємець побачить список усіх товарів, що знаходяться у відповідній БД. Зможе побачити в якій компанії закуповується певний товар та зможе подивитися на ціну за одиницю товару.

Біля 6 стоятиме сторінка для демонстрування усіх замовлень. На тій сторінці будуть виводитися замовлення, що сформовані у відповідній базі даних. Підприємець зможе побачити наявні замовлення та деталі про них: який товар замовили, яку кількість та який працівник працює над виконанням цього замовлення.

Біля числа 7 стоятиме посилання на сторінку, де будуть вирішуватися декілька прикладних задач. Перейшовши за цим посиланням, можна буде побачити вирішені прикладні задачі за допомогою даних, що знаходяться у базі даних товарів.

Ділянка, біля числа 8 буде заповнена інформацією про сторінки, які згадувалися вище. Користувач зможе прочитати інформацію про кожну сторінку, за допомогою чого зрозуміє, яка сторінка йому необхідна. Усі бази даних заповнюються в адміністративному вікні. Там є доступ до усіх баз даних, та роботи з ними.

3.3. Засоби Django для реалізації ідеї

Django — високорівневий відкритий Python-фреймворк для розробки вебсистем. Названо його було на честь джазмена Джанго Рейнхардта [9]. Перевага Джанго перед іншими фреймворками полягає у величезному наборі готового функціоналу, за рахунок якого можна швидко та просто створити систему реєстрації на сайті, додати форум на сайт, реалізувати систему пошуку по сайту або виконати якусь іншу дію на сайті.

У Джанго використовується схема MVC. Ця схема дозволяє зручно розбити файли на одну з трьох категорій: HTML шаблони, файли моделі для

роботи з базою даних та файли контролери для зв'язку моделей та HTML шаблонів між собою [10].

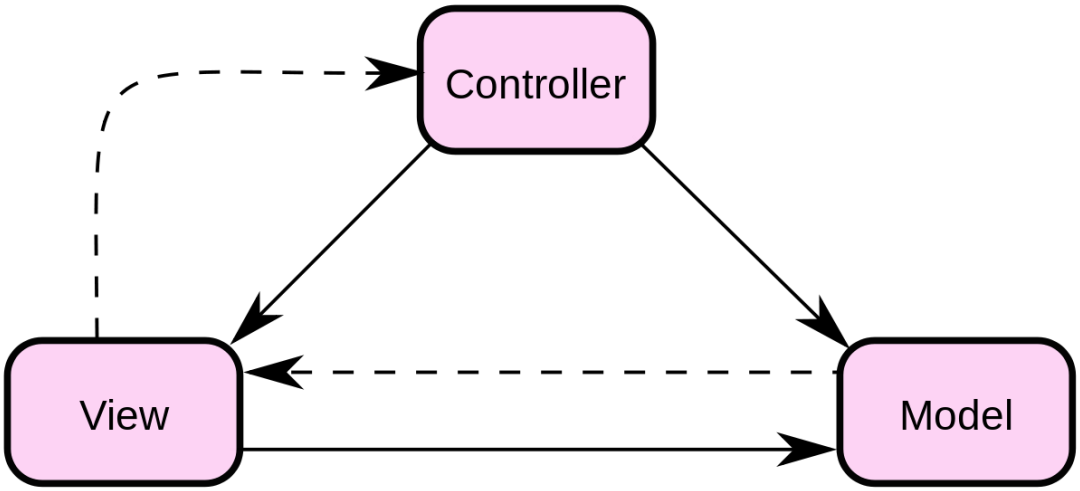

Рис 3.3.1. схема MVC

Щоб реалізувати ідею необхідно спочатку створити проект django. За допомогою команди django-admin startproject назва папки да створюєте проект, створюються всі необхідні програмні файли для роботи з django. Потім для початку, необхідно створити потрібні БД для подальшої роботи. БД створюються у файлі models.py засобами мови Python. Створивши клас, який буде містити необхідну інформацію, потрібно зареєструвати у адміністративному інтерфейсі, для подальшої зручності їх створення, заповнення, корегування та видалення у файлі admin.py, за допомогою функції admin.site.register(назва класу).

Тепер створимо будь яку веб-сторінку, щоб побачити як запустити веб-сторінку. Спочатку створюємо папку з назвою templates, в якій будемо створювати всі необхідні файли з розширенням html. Створимо будь який файл де напишемо будь який код, наприклад щоб він виводив просто якийсь текст і назвемо цей файл index.html. У файлі views.py, який знаходиться в директорії вашої програми необхідно імпортувати необхідний модуль: from django.shortcuts import render. Необхідно визначити функцію подання, яка оброблятиме запити та повертатиме HTML файл. Всередині функції використовуємо метод render:

```
def index_page(request):
    return render(request, 'index.html').
```

Потім у файлі urls.py, який також знаходиться в директорії вашої програми, додамо маршрут для даного представлення:

```
urlpatterns = [
    path(' ', index_page),
    path('admin/', admin.site.urls),
```

]

Тепер якщо запустити локальний сервер Django за допомогою python manage.py runserver, ми отримаємо адрес, перейшовши по якому, попадемо на сторінку де відображається файл index.html.

Тепер створимо якісь записи у базах даних. Для цього нам потрібно перейти до адміністративного інтерфейсу. Для цього, в кінці посилання на сторінці, на якій ми спочатку знаходимося, потрібно додати /admin. Нас перекине до вікна де необхідно буде ввести ім’я зареєстрованого користувача та пароль. Щоб зареєструватися, необхідно у консолі прописати команду python manage.py createsuperuser. Далі нас попросять ввести ім’я користувача та пароль. Після цього ми повертаємося до вікна де вводимо ім’я та пароль, і нас запускають до адміністративного сайту. Ми на ньому можемо побачити наші бази даних. Ми можемо в них заходити, створювати нові об’єкти, виправляти дані та видаляти.

Заповнюємо бази даних необхідними даними. В моєму випадку є БД клієнтів, куди заповнюється ім'я, прізвище та вік людини. Є БД співробітників, куди я записую ім'я, прізвище та зарплату. Є БД товарів, куди я заповнюю назву товару, ціну продажу за 1 шт., компанія де купую, річний попит, ціну закупівлі за 1 шт., ціна зберігання 1 товару протягом року, час поповнення запасів, попит за цей місяць та попит за минулий місяць. І є БД замовлень, де я вибираю клієнта, які вже занесені до БД клієнтів, вибираю товар, який занесений до БД товарів, їжу кількість товару та обираю працівника із БД працівників.

Після цього нам необхідно створити сайт, такий як вказаний на макеті. Для цього створимо спочатку головну сторінку, яка буде об’єднувати усі інші сторінки index2.html. Так само як ми розглядали приклад з файлом index.html, зробимо все те саме, тільки для файлу index2.html. Після цього будемо створювати нові сторінки та наповнювати файл index2.html. Засобами CSS та тегом style створимо веб-сторінку таку саме, як на макеті. Потім створимо посилання на перший сайт. Додамо до файлу index2.html посилання на сайт податкової служби в виведемо його. Тепер запустивши цей файл, ми побачимо, що у нас з’явилось посилання на даний сайт.

Далі, для створення вже інших посилань, необхідно буде створювати власті html файли. Для того щоб створити посилання на сторінку «Клієнти», необхідно у файлі views.py створити функцію в якій у створену зміну будуть передаватися усі дані з певної БД та передавати цю зміну до потрібного файлу html.

```
def client_page(request):
    all_clients = Clients.objects.all()
    context = {
```

```
    'clients': all_clients,
}
return render(request, 'clients.html', context).
```

Приклад створення такої функції для клієнтів. У зміну all_clients передаються усі дані з БД Clients за допомогою objects.all(). Потім all_clients передається у новий файл 'clients.html' для виведення інформації про клієнтів. У цьому файлі clients.html зміна 'clients' буде такою ж самою як і all_clients. Тож для виведення можна скористатися звичайним циклом for на мові Python. Коли всю необхідну інформацію прописали для виведення на сторінку, передаємо файл clients.html до файлу index2.html. Після цього, зробимо аналогічно для сторінок «Працівники», «Товари» та «Замовлення» тільки використовуючи їх бази даних.

На сторінці «Задачі управління запасами» реалізуємо три прикладні задачі: задача оптимального рівня запасу, задача оптимального замовлення, прогнозування попиту на наступний місяць. Для реалізації цих задач я додаю у базу даних товарів потрібні значення: значення річного попиту, ціну закупівлі за 1 шт., ціна зберігання 1 товару протягом року, час поповнення запасів, попит цього місяця і попит минулого місяця. Для реалізації задачі оптимального рівня запасу потрібно знати середньомісячний попит та час поповнення запасів. Перемноживши ці значення, можна отримати рівень оптимального запасу, нижче якого не варто опускатися. Отримане число буде означати поріг, при якому потрібно буде замовляти нову партію товару. Для реалізації задачі оптимального замовлення необхідно знати значення річного попиту, ціну закупівлі за 1 шт., ціна зберігання 1 товару протягом року. За допомогою формули EOQ = sqrt((2 * D * S) / H), де

D – річний попит;

S – ціна закупівлі за 1 шт.;

H – ціна зберігання 1 товару протягом року.

Використавши цю формулу, ми знайдемо оптимальне замовлення кількості товару, щоб мінімізувати загальні витрати загальні витрати на замовлення та зберігання. І третя задача прогнозування попиту на наступний місяць використовує попит за останні два місяці. Ми обираємо значення a від 0 до 1, наприклад 0.4, та підставляємо у формулу: Прогноз = останній місяць * a + передостанній місяць * (1-a). Таким чином ми можемо приблизно дізнатися значення попиту у наступному місяці, що дозволить отримати найбільший прибуток.

Щоб реалізувати ці задачі за допомогою Django, у файлі views.py створимо функцію де ми будемо брати необхідні дані з бази даних товарів, та у циклі записувати результати обчислень у списки для кожного товару. Потім передамо ці списки до файлу html та сформуємо текст, який буде виведений у веб-сторінку «Задачі управління запасами».

3.4. Програмний код

Файл models.py

```
from django.db import models


class Worker(models.Model):
    name = models.CharField(max_length=35,blank = False)
    surname = models.CharField(max_length=35,blank = False)
    salary = models.IntegerField(default=0)


    def __str__(self):
        return f"{self.surname} {self.name}"


class Products(models.Model):
    name = models.CharField(max_length=50,blank = False)
    price = models.IntegerField(default=0)
    company_name = models.CharField(max_length=100,blank = False, null=True)
    demand_per_year = models.IntegerField(default=0)#продажный спрос за год
    price_of_1_item = models.IntegerField(default=0)#стоимость покупки 1 товара
    cost_of_storing_1_product_during_the_year = models.DecimalField(default=0, max_digits=8, decimal_places=1)#стоимость хранения 1 товара в течении года
    restocking_time = models.IntegerField(default=0)#время пополнения запасов
```

```
    demand_this_month = models.IntegerField(default=0)#спрос за этот месяц
    demand_last_month = models.IntegerField(default=0)#спрос за прошлый месяц

    def __str__(self):
        return f"{self.name}"


class Clients(models.Model):
    name = models.CharField(max_length=35,blank = False)
    surname = models.CharField(max_length=35,blank = False)
    age = models.IntegerField(default=0)

    def __str__(self):
        return f"{self.surname} {self.name}"

class Order(models.Model):
    client = models.ForeignKey(Clients,on_delete = models.CASCADE, default=1)
    product = models.ForeignKey(Products,on_delete = models.CASCADE, default=1)
    count_of_products = models.IntegerField(default=0)
    worker = models.ForeignKey(Worker,on_delete = models.CASCADE, default=1)

    def __str__(self):
        return f"Замовлення для {self.client}"
```

Файл admin.py

```
from django.contrib import admin
```

```
from app1.models import Worker, Products, Clients, Order

admin.site.register(Worker)
admin.site.register(Products)
admin.site.register(Clients)
admin.site.register(Order)
```

Файл view.py

```
from django.shortcuts import render
from app1.models import Worker, Products, Clients, Order
import math

def index2_page(request):
    return render(request, 'index2.html')

def workers_page(request):
    all_workers = Worker.objects.all()
    context = {
        'workers': all_workers,
    }
    return render(request, 'workers.html', context)

def products_page(request):
    all_products = Products.objects.all()
    context = {
        'products': all_products,
    }
    return render(request, 'products.html', context)

def client_page(request):
```

```
    all_clients = Clients.objects.all()
    context = {
        'clients': all_clients,
    }
    return render(request, 'clients.html', context)

def order_page(request):
    all_order = Order.objects.all()
    context = {
        'order': all_order,
    }
    return render(request, 'order.html', context)

def tasks(request):
    all_products = Products.objects.all()
    optimal_zapas = []
    optimal_zakaz = []
    prognoz = []
    count = []
    c = 0
    for i in all_products:
        sr1 = i.demand_per_year / 12
        sr2 = i.restocking_time / 30
        res = sr1 * sr2
        optimal_zapas.append(round(res))
        s = i.price_of_1_item
        h = i.cost_of_storing_1_product_during_the_year
        d = i.demand_per_year
        eoq = round(math.sqrt((2*d*s)/h))
        optimal_zakaz.append(eoq)
```

```
            now = i.demand_this_month
            last = i.demand_last_month
            a = 0.4
            res1 = a * now + (1-a) * last
            prognoz.append(round(res1))
            count.append(c)
            c += 1
```

st = 'Перша задача - задача оптимального рівня запасу. Для цього нам потрібно знати середньомісячний попит на товар та час за який можна поповнити запаси.'

```
        str = []
        stra = []
        strb = []
        for i in range(len(count)):
            str1=f'*Товар:{all_products[i].name}'
            str2=f'***Середньомісячний попит на {all_products[i].name}: {round(all_products[i].demand_per_year / 12)}'
            str3=f'***Час за який можна поповнити запаси: {round(all_products[i].restocking_time / 30,2)} місяці'
            str4=f'Маючи такі дані, вирішуємо задачу оптимального рівня запасу. Результат: {optimal_zapas[i]}'
            str5=f'Аналізуючи результат, можна сказати, що для управління запасами слід підтримувати запаси товару лише на рівні щонайменше {optimal_zapas[i]} одиниць. Це означає, що при досягненні запасів до цього рівня слід розміщувати нове замовлення для поповнення запасів.'
            str6=f'Враховуючи середньомісячний попит та час постачання товару, оптимальний рівень запасу дозволить забезпечити безперервну наявність товару та мінімізувати ризики нестачі запасів.'
            str7 = '--------------------------------------------------------------------------------'
            str.append(str1)
```

```
            str.append(str2)
            str.append(str3)
            str.append(str4)
            str.append(str5)
            str.append(str6)
            if i < len(count) - 1:
                str.append(str7)
```

st1 = 'Друга задача - задача оптимального замовлення. Для цього нам потрібно знати ціну за одиницю товару, ціну зберігання 1 товару протягом року та попит за рік.'

```
            for i in range(len(count)):
                str1=f'*Товар:{all_products[i].name}'
                str2=f'***Ціна за одиницю товару: {all_products[i].price_of_1_item}'
                str3=f'***Ціна зберігання 1 товару протягом року: {all_products[i].cost_of_storing_1_product_during_the_year}'
                str4=f'***Річний попит: {all_products[i].demand_per_year}'
                str5=f'Маючи такі дані, вирішуємо задачу оптимального замовлення. Результат: {optimal_zakaz[i]}'
                str6=f'Аналізуючи результат, можна сказати, що це кількість товару, яку слід замовляти щоразу, щоб мінімізувати загальні витрати на купівлю та зберігання товару.'
                str7 = '--------------------------------------------------------------------------------'
                stra.append(str1)
                stra.append(str2)
                stra.append(str3)
                stra.append(str4)
                stra.append(str5)
                stra.append(str6)
                if i < len(count) - 1:
                    stra.append(str7)
```

```
        st2 = 'Третя задача - прогнозування попиту на наступний місяць. Вхідні дані - це попит за два останні місяці.'
        for i in range(len(count)):
            str1=f'*Товар:{all_products[i].name}'
            str2=f'***Попит на перший місяць: {all_products[i].demand_this_month}'
            str3=f'***Попит на другий місяць: {all_products[i].demand_last_month}'
            str5=f'Маючи такі дані, вирішуємо задачу прогнозування попиту на наступний місяць. Результат: {prognoz[i]}'
            str6=f'Аналізуючи результат, можна сказати, число {prognoz[i]}, отримане в результаті вирішення завдання прогнозування попиту на наступний місяць, вказує на очікуваний обсяг попиту на цей товар наступного місяця. Це означає, що на основі наявних даних про попит за два останні місяці та інші фактори, обраний алгоритм прогнозування передбачає, що попит становитиме приблизно {prognoz[i]} одиниць товару.'
            str7 = '--------------------------------------------------------------------------------------------------'
            strb.append(str1)
            strb.append(str2)
            strb.append(str3)
            strb.append(str5)
            strb.append(str6)
            if i < len(count) - 1:
                strb.append(str7)

        context = {
            'products': all_products,
            'optimal_zapas': optimal_zapas,
            'optimal_zakaz': optimal_zakaz,
            'prognoz':prognoz,
            'count': count,
```

```
            'str1': str,
            'st': st,
            'stra': stra,
            'st1': st1,
            'strb': strb,
            'st2': st2,
        }
        return render(request, 'tasks.html', context)
```

Файл urls.py

```
from django.contrib import admin
from django.urls import path
from app1.views import index2_page, index_page, client_page, workers_page, products_page, order_page, tasks

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', index2_page),
    path('main1/', index_page),
    path('orders/', order_page),
    path('clients/', client_page),
    path('workers/', workers_page),
    path('products/', products_page),
    path('tasks/', tasks),
]
```

Файл index2.html

```
{% load static %}
<!DOCTYPE html>
<html lang="uk">
<head>
```

```
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Інструментарій підприємця-початківця</title>
        <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css">
        <style>
          body {
            background-image: url("{% static '6.jpg' %}");
            background-size: cover;
            background-position: center;
            background-repeat: no-repeat;
          }
        </style>
    </head>
    <body>
        <div class="d-flex flex-column flex-md-row align-items-center pb-3 mb-4 border-bottom">
            <h5 style="color: rgba(137, 160, 237, 0.545)">Інструментарій підприємця-початківця</h5>
            <nav class="d-inline-flex mt-2 mt-md-0 ms-md-auto">
              <a class="me-3 py-2 link-body-emphasis text-decoration-none" href="https://tax.gov.ua/nk/">Сайт податкової служби</a>
            <a class="me-3 py-2 link-body-emphasis text-decoration-none" href="/clients">Клієнти</a>
            <a class="me-3 py-2 link-body-emphasis text-decoration-none" href="/workers">Працівники</a>
            <a class="me-3 py-2 link-body-emphasis text-decoration-none" href="/products">Товари</a>
            <a class="me-3 py-2 link-body-emphasis text-decoration-none" href="/orders">Замовлення</a>
            <a class="me-3 py-2 link-body-emphasis text-decoration-none" href="/tasks">Задачі управління запасами</a>
```

```
            </nav>
          </div>
        <p style="text-align: center;">
          <img src="{% static '1_1.png' %}"><br>
          <img src="{% static '2_2.png' %}"><br>
          <img src="{% static '3_3.png' %}"><br>
          <img src="{% static '4_4.png' %}"><br>
          <img src="{% static '5_5.png' %}"><br>
          <img src="{% static '9_9.png' %}" width="875" height="250"><br>
        </p>
      </body>
      </html>
```

Файл clients.html

```
      <!DOCTYPE html>
      <html lang="uk">
      <head>
          <meta charset="UTF-8">
          <meta name="viewport" content="width=device-width, initial-
scale=1.0">
          <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css">
          <title>Клієнти</title>
      {% load static %}
      <head>
          <style>
              body {
                  background-image: url("{% static '16.jpg' %}");
                  background-size: cover;
                  background-position: center;
```

```
            background-repeat: no-repeat;
        }
    </style>
</head>
<body>
    <h1 style="color: #ffffff;">Клієнти:</h1>
    <ul type="circle" >
        {%for i in clients%}
        <li style="padding-left: 20px; color: #ffffff;">{{i}} (Вік: {{i.age}})</li>
        <br>
        {%endfor%}
    </ul>
</body>
</html>
```

Файл order.html

```
{% load static %}
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css">
    <style>
        body {
          background-image: url("{% static '17.jpg' %}");
          background-size: cover;
          background-position: center;
          background-repeat: no-repeat;
        }
```

```
        </style>
    </head>
    <body>
        <h1 style="color: #ffffff;">Замовлення:</h1>
        <ul type="circle" >
            {%for i in order%}
            <li style="padding-left: 20px; color: #ffffff;">{{i}} (Товар: {{i.product}}({{i.product.company_name}}), Кількість товару: {{i.count_of_products}}, Працівник: {{i.worker}})</li>
            <br>
            {%endfor%}
        </ul>
    </body>
    </html>
```

Файл products.html

```
{% load static %}
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css">
    <title>Товари</title>
    <style>
        body {
          background-image: url("{% static '14.jpg' %}");
          background-size: cover;
          background-position: center;
```

```
            background-repeat: no-repeat;
        }
    </style>
</head>
<body>
    <h1>Товари:</h1>
    <ul type="circle"></ul>
        {%for i in products%}
        <li style="padding-left: 20px;">{{i}} (Компанія:
{{i.company_name}}, ціна за одиницю товару: {{i.price}} грн.)</li>
        <br>
        {%endfor%}
</body>
</html>
```

Файл workers.html

```
{% load static %}
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css">
    <title>Працівники</title>
    <style>
        body {
            background-image: url("{% static '15.jpg' %}");
            background-size: cover;
            background-position: center;
            background-repeat: no-repeat;
        }
```

```
    </style>
</head>
<body>
    <h1 style="color: #ffffff;">Працівники:</h1>
    <ul type="circle"></ul>
        {%for i in workers%}
        <li style="padding-left: 20px; color: #ffffff;">{{i}} (Заробітня плата: {{i.salary}} грн.)</li>
        <br>
        {%endfor%}
</body>
</html>
```

Файл tasks.html

```
{% load static %}
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css">
</head>
<body style="background-color: #6da3ba">
    <h1>Задачі управління запасами:</h1>
    <ul type="circle" >
        <li style="padding-left: 20px;"><h1>{{st}}</h1></li>
        {%for i in str1%}
            <p>{{i}} </p>
        {%endfor%}
        <br>
        <li style="padding-left: 20px;"><h1>{{st1}}</h1></li>
```

```
        {%for i in stra%}
           <p>{{i}} </p>
        {%endfor%}
        <br>
        <li style="padding-left: 20px;"><h1>{{st2}}</h1></li>
        {%for i in strb%}
           <p>{{i}} </p>
        {%endfor%}
    </ul>
</body>
</html>
```

3.5. Тестування ППЗ «Інструментарій підприємця - початківця» з використанням технологій Django

Щоб запустити проект Django потрібно перейти у термінал, перейти до місця розташування проєкта за допомогою cd шлях, потім прописати таку команду: python manage.py runserver і натиснути enter, після цього у терміналі з’явиться посилання на головну сторінку: Starting development server at http://127.0.0.1:8000/. Перейшовши за цим посиланням, у вікні вашого браузера з’явиться таке вікно:

Рис. 3.5.1. Головна сторінка

Як ми можемо побачити, головна сторінка схожа на макет із пункту 3.2. Розглянемо цю сторінку детально. Там де стояло число 1 на макеті, знаходиться назва веб-застосунку. Потім, там де була двійка, стоїть посилання на сайт податкової служби. Там де була трійка – посилання на клієнтів,

четвірка – сторінка працівників, п’ятірка – товари, шістка – замовлення, сімка – задачі управляння запасами та вісім – інформація про те, що знаходиться перейшовши за кожним посиланням. Розглянемо все підряд, спочатку натиснемо на посилання «Сайт податкової служби»:

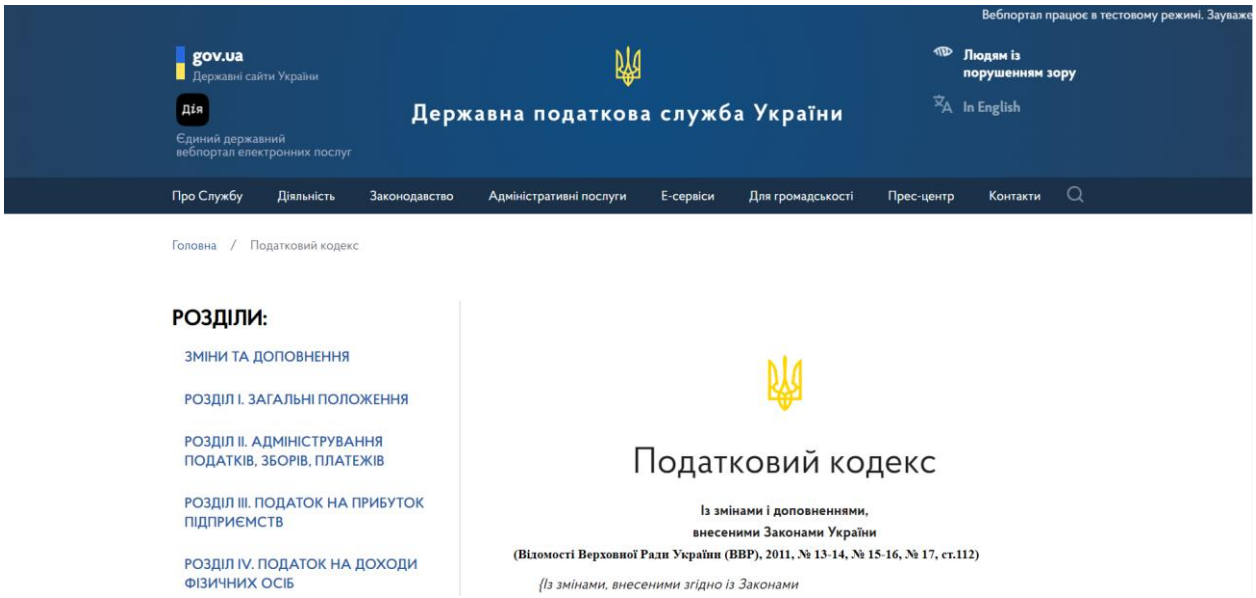

Рис 3.5.2. «Сайт податкової служби»

Як ми можемо побачити, перейшовши за цим посиланням нас перекинуло на сторінку, де написаний податковий кодекс. З цим розібралися. Далі, перейдемо на сторінку із назвою «Клієнти»:

Рис 3.5.3. «Клієнти»

На цій сторінці демонструється список клієнтів, що зареєстровані у базі даних та виводиться інформація про вік клієнтів. Далі перейдемо за посиланням на сторінку «Працівники»:

Рис 3.5.4. «Працівники»

На цій сторінці виводиться інформація про працівників. Виводиться імена та прізвище, а також заробітня плата кожного працівника. Далі перейдемо на сторінку «Товари»:

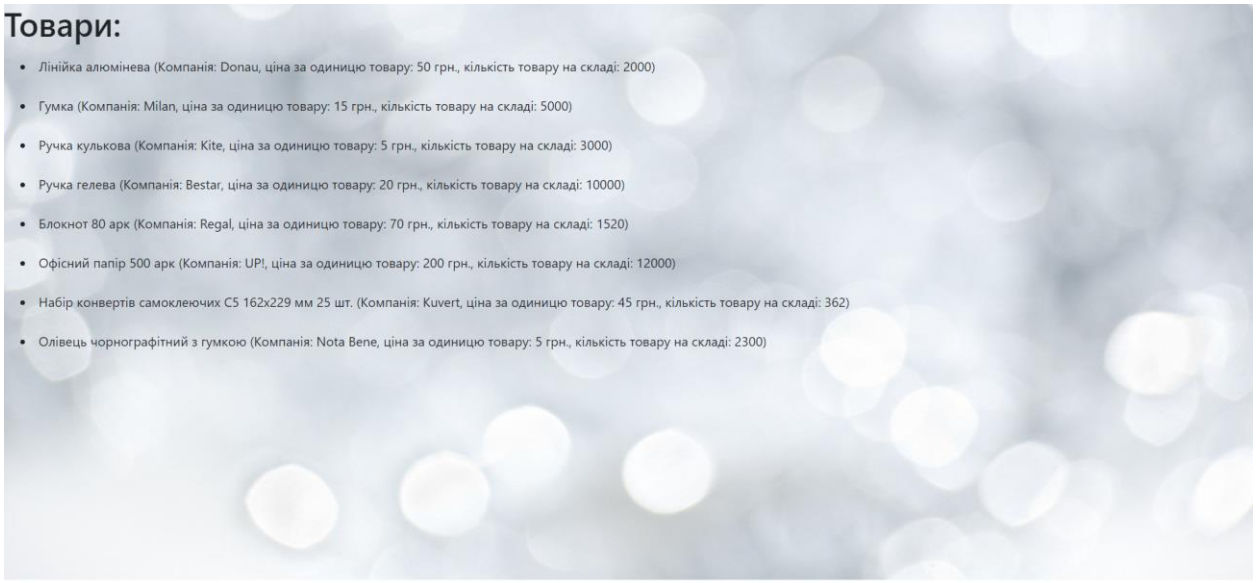

Рис 3.5.5. «Товари»

На цій сторінці виводиться інформація про товари, які продає підприємець. Виводиться назва товару, компанія у якої закуповується товар, ціна за одиницю товару та кількість товару, що знаходиться на складі. Потім подивимося, що знаходиться на сторінці «Замовлення»:

Рис 3.5.6. «Замовлення»

На цій сторінці виводиться інформація по кожному замовленню. Виводиться для якого клієнта це замовлення та виводиться інформація: який товар замовила людина, у якій кількості та ім’я і прізвище працівника, який виконує це замовлення. Залишилося подивитися, що знаходиться на сторінці «Задачі управління запасами»:

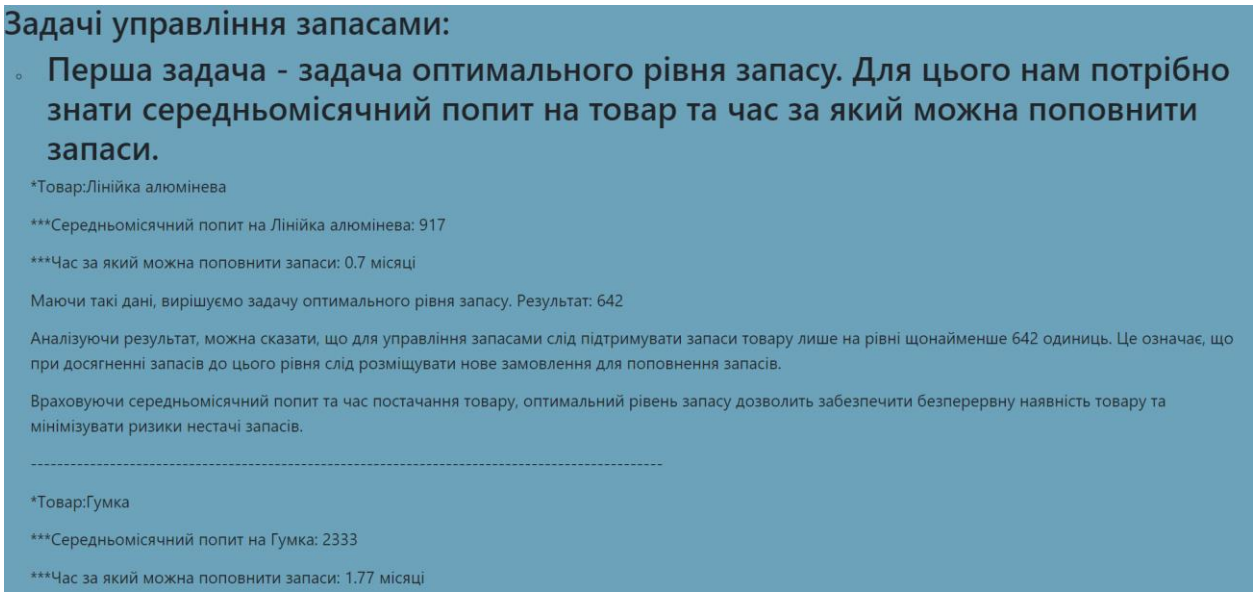

Рис 3.5.7. Задача оптимального рівня запасу

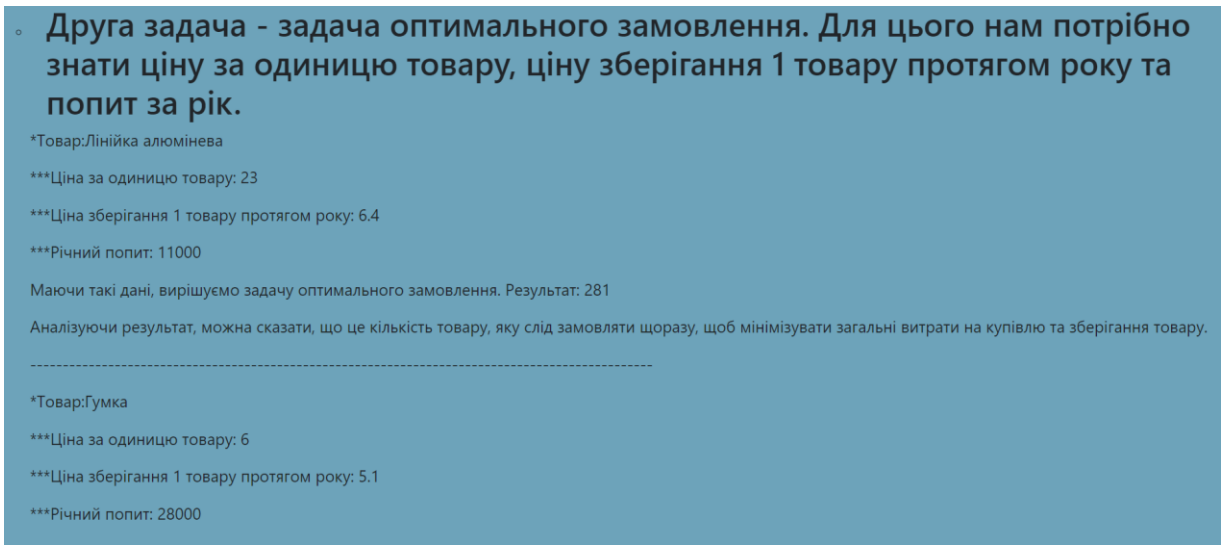

Рис 3.5.8. Задача оптимального замовлення

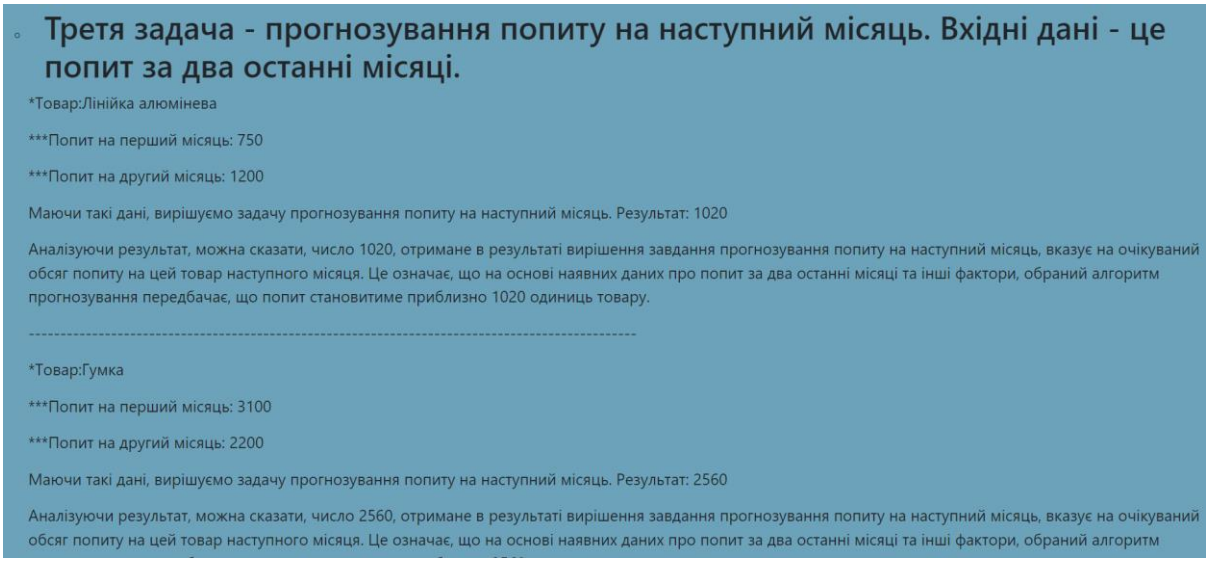

Рис 3.5.9. Задача прогнозування попиту на наступний місяць

Як ми бачимо, на сторінці «Задачі управління запасами» вирішуються три прикладні задачі (задача оптимального рівня запасу, задача оптимального замовлення, задача прогнозування попиту на наступний місяць) для кожного товару. У першій задачі знаходиться мінімальну кількість товару, яке має бути на складі і у разі його досягнення, потрібно буде одразу робити замовлення на нову партію певного товару. У другій задачі знаходиться кількість товару, яку необхідно замовити для мінімізації витрат на купівлю та зберігання товару. І у третій задачі вирішується питання прогнозу попиту на наступний місяць. Результатом вирішення цієї задачі буде приблизно прогнозований попит на наступний місяць, що допоможе підприємцю отримати найбільший прибуток.

Тепер розглянемо де заповнюються бази даних, то як їх заповнювати, корегувати та видаляти. Знаходячись на головній сторінці, потрібно до посилання сайту дописати /admin. Перейшовши за цим посиланням, на перекине на адміністративну сторінку, яка буде виглядати ось так:

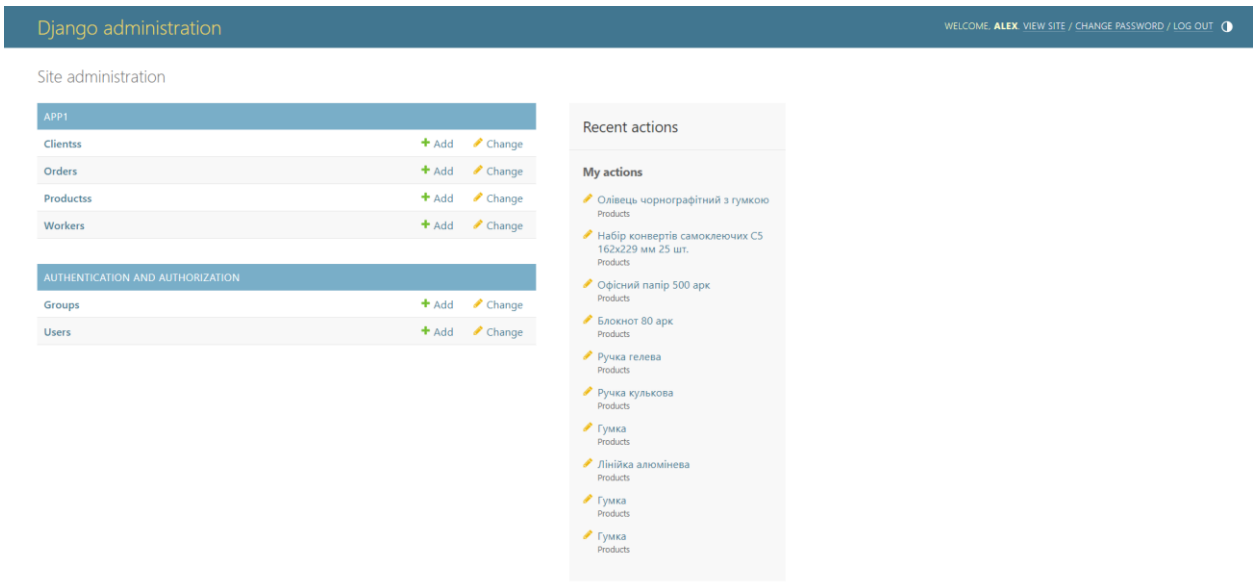

Рис 3.5.10. Адміністративна сторінка

Нас цікавлять бази даних, де знаходиться уся необхідна інформація. У колонці APP1 ідуть назви Clientss, Orders, Products, Workers. Це і є наші бази даних клієнтів, замовлень, товарів, працівників відповідно. Перейшовши наприклад до бази даних клієнтів:

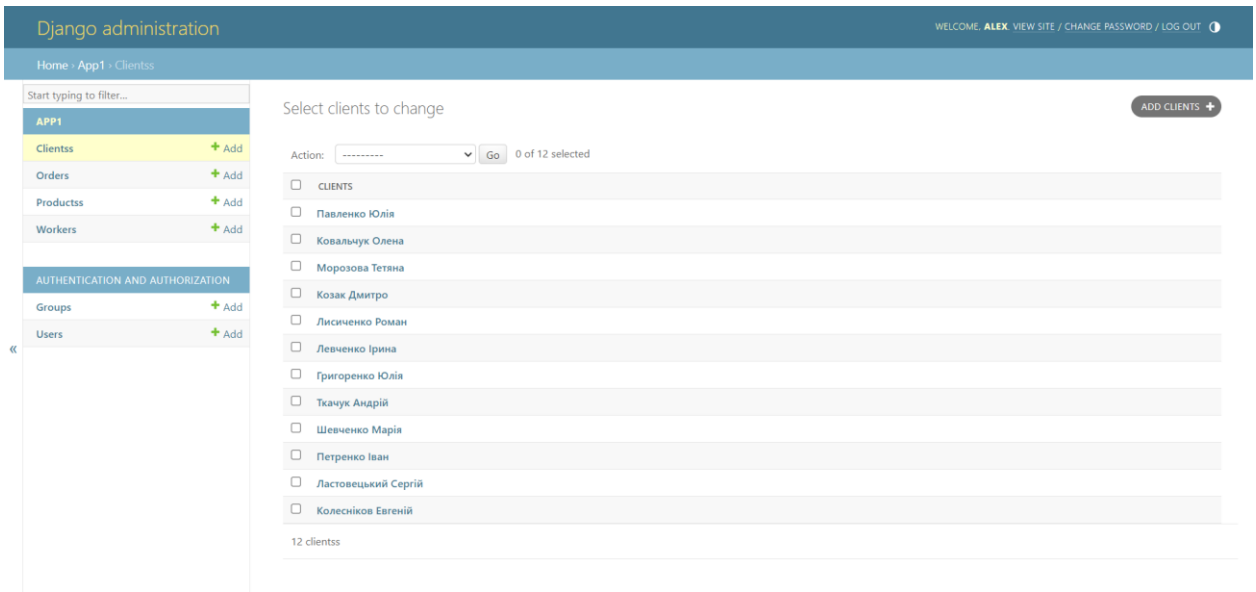

Рис 3.5.11. База даних клієнтів

Тут знаходяться усі клієнти. Щоб додати нового клієнта, праворуч вгорі можна додати нового клієнта. Тепер переглянемо, що буде якщо натиснути наприклад на Павленко Юлію:

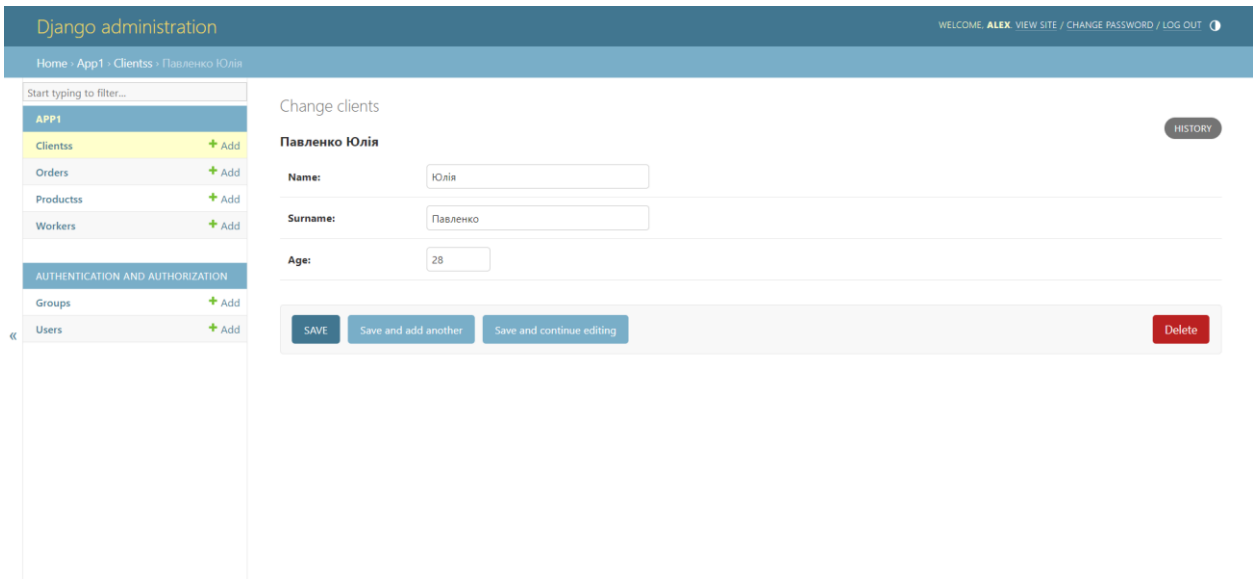

Рис 3.5.12. Приклад клієнта у базі даних

Ми бачимо, що тут зберігається уся інформація про Юлію. Аналогічно дані зберігаються й у всіх інших клієнтів. Щоб редагувати ми просто замінюємо значення у потрібній комірці і натискаємо Save, для того щоб видалити цього клієнта, праворуч знизу є кнопка Delete, тоді клієнт повність видалиться із БД. Аналогічно базі даних клієнтів, робиться й у інших базах даних. Потрібно заповнювати необхідні дані й зберігати й базі даних. Головне вікно автоматично зчитує усю інформацію із бази даних, тож якщо ви щось зміните, то ця зміна відобразиться у певній сторінці.

Висновки

1. Мною була розглянута тема: інструментарій підприємця - початківця з використанням технологій Django.
2. Вивчено базові технології Django.
3. За допомогою цих технологій, реалізував веб-застосунок для підприємця-початківця.
4. Застосунок має функціонал заповнювати бази даних, редагувати їх, переглядати основну інформацію про клієнтів, працівників, товарів та замовлень.
5. Реалізував три прикладні задачі для управління запасами.