Київський національний торговельно-економічний університет

Кафедра комп'ютерних наук та інформаційних систем

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Розробка програмного забезпечення для проведення онлайнового анкетування»

Студента 4 курсу, 10 групи

спеціальності
122 «Комп'ютерні науки»

_____
підпис студента

Нальотов
Володимир
Володимирович

Науковий керівник
кандидат економічних наук, доцент

_____
підпис керівника

Шклярський
Сергій
Михайлович

Гарант освітньої програми
кандидат технічних наук, доцент

_____
підпис керівника

Демідов Павло
Георгійович

Київ 2021

Київський національний торговельно-економічний університет

Факультет інформаційних технологій
Кафедра комп'ютерних наук та інформаційних систем
Спеціальність 122 «Комп'ютерні науки»

Затверджую
Зав. кафедри _____Пурський О.І.
«20» грудня 2020р.

Завдання
на випускну кваліфікаційну роботу (проект) студента

Нальотов Володимир Володимирович
(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи (проекту)

«Розробка програмного забезпечення для проведення онлайнового анкетування»
Затверджена наказом ректора від *«15» грудня 2020 р. № 3780*
2. Строк здачі студентом закінченої роботи *31 травня2021 року*
3. Цільова установка та вихідні дані до роботи
Мета роботи: розробка програмного забезпечення для проведення онлайнового анкетування
Об'єкт дослідження: процеси розробки програмного забезпечення
Предмет дослідження: інформаційні системи і технології в системі проведення онлайнового анкетування
4. Перелік графічного матеріалу рисунки, таблиці, лістинг-код, програмне забезпечення, презентація
5. Консультанти по роботі із зазначенням розділів, за якими здійснюється консультування:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв

1	Шклярський С.М.	15.12.2020 р.	15.12.2020 р.
2	Шклярський С.М.	15.12.2020 р.	15.12.2020 р.
3	Шклярський С.М.	15.12.2020 р.	15.12.2020 р.

6. Зміст випускної кваліфікаційної роботи (проекту) (перелік питань за кожним розділом)

ВСТУП

РОЗДІЛ 1. ТЕОРЕТИЧНІ АСПЕКТИ ПРОВЕДЕННЯ АНКЕТУВАННЯ В СУЧАСНОМУ СЕРЕДОВИЩІ

1.1. Поняття та зміст анкетування населення

1.1.1. Тип запитань

1.1.2. Послідовність запитань

1.1.3. Основні правила побудови опитування

1.1.4. Багатопозиційні ваги

1.1.5. Режими адміністрування анкет

1.1.6. Проблеми з анкетами

1.2. Методи і особливості проведення анкетування

1.2.1. За кількістю респондентів

1.2.2. По повноті охоплення

1.2.3. За типом контактів з респондентом

1.2.4. Онлайн опитування

Висновки до розділу 1

РОЗДІЛ 2. МОДЕЛЬ ПРОВЕДЕННЯ АНКЕТУВАННЯ В ОНЛАЙН ФОРМАТІ

2.1. Формування системи показників для проведення онлайнового анкетування

2.2. Модель проведення онлайнового опитування

2.2.1. Відповіді

2.2.2. Адміністрування

2.2.3. Якість

2.2.4. Етика

Висновки до розділу 2

РОЗДІЛ 3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ПРОВЕДЕННЯ ОНЛАЙНОВОГО АНКЕТУВАННЯ

3.1. Інформаційно-логічна модель проведення онлайнового анкетування

3.1.1. Діаграма варіантів використання

3.1.2. Діаграма класів

3.2. Програмна реалізація проведення онлайнового анкетування

3.2.1. Мова програмування Python

3.2.2. Фреймворк Django

3.2.3. База даних SQLite

3.2.4. Розробка графічного інтерфейсу користувача

ВИСНОВКИ

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

ДОДАТКИ

7. Календарний план виконання роботи

№ пор.	Назва етапів випускного кваліфікаційного проекту	Строк виконання етапів роботи	
		За планом	фактично
1	2	3	4
1	*Вибір теми випускного кваліфікаційного проекту*	*05.10.2020*	*05.10.2020*
2	*Розробка та затвердження завдання на випускний кваліфікаційний проект*	*18.12.2020*	*18.12.2020*
3	*Вступ*	*03.02.2021*	*03.02.2021*
4	*Розділ 1.*	*26.02.2021*	*26.02.2021*
5	*Розділ 2.*	*06.04.2021*	*06.04.2021*
6	*Розділ 3.*	*12.05.2021*	*12.05.2021*
7	*Висновки*	*14.05.2021*	*14.05.2021*
8	*Здача випускного кваліфікаційного проекту на кафедру науковому керівнику*	*20.05.2021*	*20.05.2021*
9	*Попередній захист випускного кваліфікаційного проекту*	*26.05.2021*	*26.05.2021*
10	*Виправлення зауважень, зовнішнє рецензування випускного кваліфікаційного проекту*	*27.05.2021*	
12	*Представлення готового зшитого випускного кваліфікаційного проекту на кафедру*	*31.05.2021*	
13	*Публічний захист випускного кваліфікаційного проекту*	*За розкладом роботи ЕК*	

8. Дата видачі завдання «22» грудня 2020 р.

9. Керівник випускної кваліфікаційної роботи (проекту)
Шклярський С.М.
(прізвище, ініціали, підпис)

10. Гарант освітньої програми Демідов П.Г.
(прізвище, ініціали, підпис)

11. Завдання прийняв до виконання студент-дипломник
Нальотов В.В.
(прізвище, ініціали, підпис)

12. Відгук керівника випускної кваліфікаційної роботи (проекту)

Керівник випускної кваліфікаційної роботи (проекту)
_____ 30.05.2020 р.
(підпис, дата)

13. Висновок про випускну кваліфікаційну роботу (проект)

Випускна кваліфікаційна робота (проект) студента _____
(прізвище, ініціали)
може бути допущена до захисту в екзаменаційній комісії.

Гарант освітньої програми_____Демідов П.Г.____
(підпис, прізвище, ініціали)

Завідувач кафедри _____Пурський О.І.___
(підпис, прізвище, ініціали)
«_____»_____2021 р.

АНОТАЦІЯ

Тема дипломної роботи: «Програмне забезпечення для проведення онлайнового опитування»

Виконав: Нальотов Володимир Володимирович

У випускній кваліфікаційній роботі розглядається питання створення системи проведення онлайнового опитування. Дипломна робота складається з восьми частин, а саме зі: вступу; першого розділу, в якому розглядається і описуються теоретико-методологічні основи дослідження, поняття та зміст онлайнового опитування, методи і особливості проведення онлайнового опитування; другого розділу, у якому проведено побудову моделі системи проведення онлайнового опитування, проведено аналіз системи показників для проведення онлайнового опитування, сформовано моделі проведення тестування; третього розділу, в якому відбувається вибір засобів програмної реалізації системи, її проектування за допомогою діаграм та, власне, розробка інформаційної системи; висновків, списку використаних джерел та додатків.

Випускна кваліфікаційна робота є науково-практичним дослідженням з питання «Створення програмного забезпечення для проведення онлайнового опитування». Метою кваліфікаційної випускної роботи виступає розроблення системи проведення онлайнового опитування засобами мови програмування Python і фреймворку Django.

Досліджено сучасні методи і засоби створення опитувань, виявлено відомі типи опитувань та їх особливості, виявлено методи і особливості проведення опитувань.

Завдяки обґрунтуванню теоретичних і методологічних положень спроектовано функціональні вимоги до інформаційної системи, проведено

аналіз системи показників для проведення онлайнового опитування, сформовано моделі проведення онлайнового опитування.

На основі результатів, отриманих в ході моделювання системи проведення онлайнового опитування, було спроектовано і розроблено інформаційну систему онлайнового опитування, яка в повній мірі виконує закладений в неї функціонал та готова до використання в умовах реального онлайнового опитування в будь-якій предметній області. .

В ході проведеного дослідження було виявлено низку інструментів та ключових етапів впровадження інформаційних систем і комп'ютерних технологій в тематику тестування знань.

Апробація отриманих результатів допоможе вітчизняним спеціалістам сформувати єдиний алгоритм щодо впровадження і формування конкурентної стратегії, а також у функціонування технологій онлайнового опитування на вітчизняному та закордонному ринках.

До ключових слів належать наступні: веб-сайт, опитування, HTML, CSS, Python, Django.

ЗМІСТ

ВСТУП

Кожного дня діджиталізація різноманітних процесів стає все більш актуальною тенденцією в сучасному світі. Кожного дня в різноманітні процеси впроваджуються сучасні технології.

На хвилі впровадження технологічних рішень в процеси людського життя з’явилися загальнодоступні ресурси для онлайнового анкетування, які все більш щільно входять в життя сучасної людини.

Виходячи з актуальності описаного явища, яку неможливо недооцінити, об’єктом дослідження стало явище онлайнового опитування.

Предметом дослідження, в свою чергу, є методи і інструменти розроблення ресурсів для проведення онлайнового опитування.

Метою даної роботи було обрано створення програмного забезпечення для проведення онлайнового опитування.

Для виконання поставленої мети слід виконати наступні завдання:

- Проаналізувати предметну область
- Обрати засоби реалізації
- Створити діаграму варіантів використання
- Створити діаграму класів
- Спроектувати графічний інтерфейс користувача

Основні методи дослідження — аналіз літературних та інших джерел, комп’ютерне моделювання, комп’ютерне проектування, створення програмного забезпечення.

РОЗДІЛ 1. ТЕОРЕТИЧНІ АСПЕКТИ ПРОВЕДЕННЯ АНКЕТУВАННЯ В СУЧАСНОМУ СЕРЕДОВИЩІ

1.1. Поняття та зміст анкетування населення

Анкета - це інструмент дослідження, що складається з ряду питань (або інших типів підказок) з метою збору інформації від респондентів. Анкета була винайдена Лондонським статистичним товариством у 1838 р. [1] [2]

Хоча анкети часто розроблені для статистичного аналізу відповідей, це не завжди так.

Анкети мають переваги перед деякими іншими типами опитувань тим, що вони дешеві, не вимагають стільки зусиль від запитувача, як усні або телефонні опитування, і часто мають стандартизовані відповіді, що спрощують збір даних [3]. Однак такі стандартизовані відповіді можуть розчарувати користувачів, оскільки можливі відповіді можуть не точно відображати бажані відповіді [4]. Анкети також різко обмежені тим фактом, що респонденти повинні вміти читати питання та відповідати на них. Таким чином, для деяких демографічних груп проведення опитування за допомогою анкетування може бути нереально здійсненним.

Можна розрізнити опитувальники із запитаннями, що вимірюють окремі змінні, та опитувальники із запитаннями, які об’єднані у шкалу чи індекс. Анкети з питаннями, що вимірюють окремі змінні, можуть, наприклад, включати запитання щодо:

- уподобання (наприклад, політична партія)
- поведінка (наприклад, споживання їжі)
- факти (наприклад, стать)

Анкети з питаннями, які об'єднані у шкалу чи індекс, включають, наприклад, запитання, які вимірюють:

- приховані риси
- ставлення (наприклад, до імміграції)
- індекс (наприклад, соціально-економічний статус)

1.1.1. Тип запитань

Зазвичай анкета складається з ряду питань, на які респондент повинен відповісти у встановленому форматі. Розрізняють питання відкритого та закритого типу. Відкрите запитання пропонує респонденту сформулювати власну відповідь, тоді як закрите запитання відповідає обирати відповідь із заданої кількості варіантів. Варіанти відповіді на закрите запитання повинні бути вичерпними та взаємовиключними. Розрізняють чотири типи шкал відповідей на закриті питання:

- Дихотомічний, де респондент має два варіанти
- Номінально-політомічний, де респондент має більше двох невпорядкованих варіантів
- Порядково-політомічний, де респондент має більше двох упорядкованих варіантів
- (Обмежений) Безперервний, де респондент отримує суцільну шкалу

Відповідь респондента на відкрите питання згодом кодується у шкалу відповідей. Прикладом відкритого питання є запитання, коли випробуваний повинен закінчити речення (пункт завершення речення). [8]

1.1.2. Послідовність запитань

Загалом, запитання повинні логічно переходити від одного до іншого. Щоб досягти найкращих показників відповіді, запитання повинні переходити

від найменш чутливих до найбільш чутливих, від фактичних та поведінкових до ставлення та від більш загальних до більш конкретних.

Зазвичай існує потік, якого слід дотримуватися при складанні опитувальника щодо порядку, в якому задаються питання. Порядок такий:

1. Екрани
2. Розминки
3. Переходи
4. Пропускає
5. Важко
6. Класифікація

Екрани використовуються як метод скринінгу, щоб рано з'ясувати, чи повинен хтось заповнювати анкету чи ні. Відповіді на розминки прості, допомагають зацікавити опитування і можуть навіть не стосуватися цілей дослідження. Питання переходу використовуються для того, щоб змусити різні області добре зливатися. Пропуски включають запитання, схожі на "Якщо так, то відповідайте на питання 3. Якщо ні, то продовжуйте запитання 5." Складні питання до кінця, оскільки респондент перебуває в "режимі відповіді". Крім того, під час заповнення анкети в Інтернеті, індикатори прогресу дають респонденту зрозуміти, що вони майже закінчили, тому вони охочіше відповідають на складніші запитання. Класифікація чи демографічне запитання мають бути в кінці, оскільки зазвичай вони можуть відчувати себе особистими запитаннями, що спричинить респондентів дискомфорт і не бажає закінчувати опитування. [9]

1.1.3. Основні правила побудови опитування

- Використовуйте твердження, які тлумачаться однаково представниками різних субпопуляцій зацікавленої сукупності.
- Використовуйте твердження, коли особи, які мають різні думки чи риси, даватимуть різні відповіді.

- Подумайте про наявність категорії "відкритих" відповідей після списку можливих відповідей.
- Використовуйте лише один аспект конструкції, яка вас цікавить, для кожного елемента.
- Використовуйте позитивні твердження та уникайте негативів чи подвійних негативів.
- Не робіть припущень щодо респондента.
- Використовуйте чіткі та зрозумілі формулювання, легко зрозумілі для всіх рівнів освіти
- Використовуйте правильну орфографію, граматику та розділові знаки.
- Уникайте предметів, які містять більше одного запитання на предмет (наприклад, чи любите ви полуницю та картоплю?).
- Питання не повинно бути упередженим або навіть вести учасника до відповіді.

1.1.4. Багатопозиційні ваги

У рамках соціальних досліджень та практики опитувальники найчастіше використовуються для збору кількісних даних за допомогою багатопозиційних шкал із наступними характеристиками: [10]

- Для кожної досліджуваної змінної подається кілька тверджень або запитань (мінімум ≥3; зазвичай ≥5).
- Кожне твердження або запитання має супровідний набір рівновіддалених точок відповіді (зазвичай 5-7).
- Кожна точка відповіді має супровідний словесний якір (наприклад, "повністю згоден"), що зростає зліва направо.
- Словесні якорі повинні бути збалансовані, щоб відображати рівні інтервали між точками відповіді.

- У сукупності набір балів відповідей та супровідні словесні закріплення називаються шкалою оцінок. Однією з дуже часто використовуваних шкал оцінок є шкала Лікерта.
- Зазвичай для наочності та ефективності в анкеті представлений один набір якорів для декількох шкал оцінок.
- У сукупності висловлювання чи запитання із супровідною шкалою рейтингу називають предметом.
- Коли кілька елементів надійно і достовірно вимірюють одну і ту ж змінну, вони в сукупності позначаються як багатоелементна шкала або психометрична шкала.
- Для багатоелементної шкали слід встановити наступні типи надійності та валідності: внутрішня надійність, надійність тестування та повторного тестування (якщо змінна передбачається стабільною з часом), валідність вмісту, валідність конструкції та валідність критерію.
- Факторний аналіз використовується в процесі розробки масштабу.
- Анкети, що використовуються для збору кількісних даних, зазвичай складаються з декількох багатоелементних шкал, а також вступного та заключного розділу.

1.1.5. Режими адміністрування анкет

Основні способи введення анкет включають: [8]

- Очна анкета, де інтерв’юер презентує предмети усно.
- Адміністрування анкет з паперу та олівця, де предмети представлені на папері.
- Комп’ютеризоване адміністрування анкет, де предмети представлені на комп’ютері.
- Адаптивне комп’ютеризоване адміністрування анкет, де на комп’ютері представлений вибір предметів, і на основі відповідей на ці елементи

комп’ютер вибирає такі елементи, оптимізовані для оцінюваних здібностей або особливостей тестуваного.

1.1.6. Проблеми з анкетами

Хоча анкети недорогі, швидкі та прості в аналізі, часто анкета може мати більше проблем, ніж переваг. Наприклад, на відміну від інтерв’ю, люди, які проводять дослідження, можуть ніколи не дізнатися, чи відповідач зрозумів питання, яке йому задавали. Крім того, оскільки запитання настільки специфічні, що задають дослідники, отримана інформація може бути мінімальною. [11] Часто такі анкети, як Індикатор типу Майерса-Бриггса, дають занадто мало варіантів відповіді; респонденти можуть відповісти на будь-який варіант, але повинні вибрати лише одну відповідь. Анкети також дають дуже низькі показники прибутку, незалежно від того, чи це електронна анкета. Інша проблема, пов’язана зі ставками повернення, полягає в тому, що часто люди, які повертають анкету, - це ті, хто має справді позитивну чи справді негативну точку зору і хоче, щоб їх думка була почута. Люди, які, швидше за все, неупереджені в будь-якому випадку, зазвичай не реагують, оскільки це не варте їх часу.

Одне з ключових занепокоєнь анкет полягає в тому, що вони можуть містити досить великі похибки вимірювань. Ці помилки можуть бути випадковими або систематичними. Випадкові помилки спричинені ненавмисними помилками респондентів, інтерв'юерів та / або кодерів. Систематична помилка може виникнути, якщо систематично реагують респонденти на шкалу, яка використовується для формулювання питання опитування. Таким чином, точне формулювання питання опитування та його масштаб є вирішальним, оскільки вони впливають на рівень похибки вимірювання. [13]

Крім того, якщо анкети не збираються із застосуванням обґрунтованих методів вибірки, часто результати можуть бути нерепрезентативними для сукупності, оскільки така хороша вибірка має вирішальне значення для отримання репрезентативних результатів на основі анкет. [14]

1.2. Методи і особливості проведення анкетування

1.2.1. За кількістю респондентів

- Індивідуальне анкетування - опитується один респондент;
- Групове анкетування - опитуються декілька респондентів;
- Аудиторне анкетування - методична та організаційна різновид анкетування, що складається в одночасному заповненні анкет групою людей, зібраних в одному приміщенні відповідно до правил вибіркової процедури;
- Масове анкетування - беруть участь від сотні до декількох тисяч респондентів (на практиці робота трудомістка, а результати менш коректні).

1.2.2. По повноті охоплення

- Суцільне - опитування всіх представників вибірки;
- Вибіркове - опитування частини представників вибірки.

1.2.3. За типом контактів з респондентом

- Очне - проводиться в присутності дослідника-анкетёра;
- Заочне - анкетёр відсутня:
    - Розсилка анкет поштою;
    - Публікація анкет в пресі;
    - Публікація анкет в Інтернеті;
    - Вручення і збір анкет за місцем проживання, роботи і т. Д.

1.2.4. Онлайн опитування

З ростом популярності Інтернету все більш затребуваним способом збору даних стає онлайн анкетування. Дизайн онлайн опитувальників часто впливає на результат опитування. До таких факторів дизайну відносять якість

керівництва опитувальників, доступні формати представлення даних (питань), способи управління, рівень опрацювання та етичні складові опитувальника. Ряд сайтів дає безкоштовну можливість створити онлайн опитувальник і зібрати дані.

Комп'ютерне веб-опитування (CAWI) - це метод опитування в Інтернеті, при якому допитуваний дотримується сценарію, наданого на веб-сайті. Анкети складаються в програмі для створення веб-інтерв'ю. Програма дозволяє опитувальнику містити картинки, аудіо- та відеокліпи, посилання на різні веб-сторінки тощо. Веб-сайт може налаштувати хід опитування на основі наданих відповідей, а також відомостей, уже відомих про учасника. Це вважається більш дешевим способом опитування, оскільки не потрібно залучати людей для проведення опитувань, на відміну від телефонних опитувань за допомогою комп'ютера. Зі збільшенням використання Інтернету анкети в Інтернеті стали популярним способом збору інформації. Дизайн онлайн-опитувальника суттєво впливає на якість зібраних даних. Існує багато факторів при розробці онлайн-анкети; керівні принципи, доступні формати запитань, адміністративні питання, питання якості та етики слід переглянути. Інтернет-анкети слід розглядати як підмножину більш широкого кола методів онлайн-дослідження.

Є кілька причин, чому хтось використовував опитувальники в Інтернеті як бажаний метод тестування. Деякі переваги та недоліки цього методу узагальнені нижче: [1] [2]

Переваги:

- Адміністратор має більшу гнучкість у відображенні питань. Запитання можна відобразити за допомогою: [1]
    - Прапорці
    - Розкривні меню
    - Спливаючі меню
    - Екрани довідки

- Підменю

- Інтернет-форум дозволяє швидше отримувати відповіді від суб'єктів. [3] [4]
- Цей спосіб також дешевший у вживанні, оскільки відсутні витрати, пов'язані із придбанням паперу чи інших матеріалів для друку. Також зменшуються поштові витрати. [5]
- Оскільки дані збираються в центральній базі даних, час на аналіз згодом скорочується [6].
- Виправити помилки в онлайновій анкеті простіше, оскільки адміністратору не потрібно передруковувати всі анкети для розповсюдження. [1]

Недоліки:

- Не всі мають доступ до Інтернету, тому рівень відповіді обмежений. [1]
- Багато людей не сприймають заповнення анкет через Інтернет. [7]
- Дослідження показують, що демографічні показники, які відповідають на запрошення через анкети в Інтернеті, як правило, упереджені для молодих людей [8].

Висновки до розділу 1

Після написання даного розділу, в якому в свою чергу здійснено аналіз за літературними джерелами стосовно порушеного питання. Власне, уся зібрана інформація, а саме створення теоретичної бази стосовно онлайн опитувань, дала змогу оцінити ситуацію та сформувати базову картину та план, щодо створення інформаційної системи.

Таким чином, перший розділ бакалаврської роботи дав змогу зрозуміти та ознайомитись з наявною теоретичною інформацією стосовно функціонування онлайн опитувань.

РОЗДІЛ 2. МОДЕЛЬ ПРОВЕДЕННЯ АНКЕТУВАННЯ В ОНЛАЙН ФОРМАТІ

2.1. Формування системи показників для проведення онлайнового анкетування

Інтернет-анкету потрібно ретельно продумати перед її запуском. Існує кілька важливих парадигм, які слід враховувати при створенні анкети в Інтернеті. [1]

Збір та встановлення пріоритетів даних

- Цілі первинного розслідування потрібно переглянути, щоб визначити, яку інформацію потрібно збирати. [9]
- Необхідна інформація має бути неупередженим, упорядкована за значимістю. Теми не повинні наводити людину на хибні висновки. [10]

Формат анкети в Інтернеті

- Анкета повинна починатися з короткого вступу, який повідомляє суб'єкту, чому проводиться анкетування. [10]
- Запитання для опитувальника слід створювати у найбільш підходящому форматі, що полегшує розуміння. [1]
- При створенні макету анкети в Інтернеті для зменшення складності слід використовувати “розумне розгалуження”. Наприклад, якщо суб'єкт вибирає "так" для питання, анкета автоматично переходить до наступного відповідного питання і навпаки. [5]
- У кінці анкети слід включити коротку подяку "5". [5]

Довжина анкети

- Як правило, анкета не повинна перевищувати 5 хвилин. [10]
    - Як правило, 4 запитання з декількома варіантами займають 1 хвилину. [10]
    - Одне питання з короткою відповіддю еквівалентно 3 питанням із множинним вибором. [10]

Прототипування

- Перед публікацією в Інтернеті зразок анкети повинен бути розданий принаймні п’ятьом особам. Після заповнення анкети слід отримати відгук від учасників. [1]
    - Потрібно зібрати інформацію щодо того, чи зрозуміли вони основний момент анкети. [3]
    - Важливо розрізнити, чи виникали у учасників труднощі з будь-яким із запитань. [7]
    - Відгуки суб’єктів слід використовувати для внесення будь-яких необхідних змін до анкети. [3]

При розробці анкети слід пам’ятати про метод оцінки при виборі формату відповіді. У цьому розділі є різні формати відповідей, які можна використовувати в онлайнових анкетах. [1]

Радіокнопки.

Респондент повинен натиснути на коло, яке відповідає бажаній відповіді. Точка посередині з’явиться після того, як буде вибрано відповідь. Можна вибрати лише одну відповідь. [1]

- Рекомендується, коли вибір відповідей взаємовиключний.
- Відповіді за замовчуванням надавати не слід. Якщо надається відповідь за замовчуванням, це може бути помилковим як відповідь, якщо респондент вирішить пропустити питання.
- Вимагати точності клацання. [1]

Рис. 2.1. Приклад радіокнопок

Прапорці

Респондент повинен натиснути поле біля відповіді, яке відповідає бажаному вибору. Після вибору відповіді у вікні з’явиться галочка. Можна вибрати декілька відповідей. [1]

- Якщо варіантів багато, рекомендується проста матриця. [7]
- При використанні прапорців, якщо можна встановити кілька відповідей, це слід вказати в інструкціях. [7]
- Якщо «нічого з перерахованого вище» не потрібно, надайте йому перемикач, щоб уникнути помилкової перевірки цього вибору у випадку, якщо обрана інша відповідь. [1]

Рис. 2.2. Приклад використання прапорців

Спадні списки

Респондент повинен натиснути стрілку в крайній правій частині вікна. [1] Після натискання відобразиться дисплей зі списком відповідей. [3] Якщо відображається велика кількість відповідей, праворуч може з’явитися смужка прокрутки. [3] Респондент може натиснути виділений розділ списку, щоб вибрати відповідь. [1] Потім ця відповідь з’явиться у вікні. Для цього типу запитань можна вибрати лише одну відповідь. [1]

- Хороший варіант для довгих списків, таких як держава / країна проживання. [3]
- Слід уникати тих предметів, де швидше друкувати. Наприклад, рік народження. [3]
- При розробці випадаючих вікон не робіть перший варіант видимим для респондента. Це може ввести в оману, коли не можна вибрати жодної відповіді. [7]

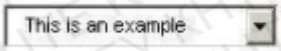


Рис. 2.3. Приклад спадного списку

Питання відкритого типу

Питання відкритого типу - це ті, що дозволяють респондентам відповісти своїми словами. В Інтернет-опитуванні текстові поля отримують запит на запитання, щоб респонденти могли ввести свою відповідь. Питання відкритого типу шукають вільної відповіді і мають на меті визначити, що є головним у

розумі респондента. Їх добре використовувати, коли запитують про ставлення чи почуття, симпатії та антипатії, згадування пам’яті, думки чи додаткові коментарі.

Респондент повинен натиснути всередині текстового поля, щоб отримати курсор усередині поля. Коли курсор блимає всередині поля, можна ввести відповідь на запитання. [7]

- Зробіть розмір текстового поля відповідно до бажаного та необхідного обсягу інформації від респондента. [1]
- Надайте стислі та чіткі інструкції щодо введення.

Рейтингові шкали

Респондент повинен вибрати одне значення зі шкали можливих варіантів; наприклад, бідний, справедливий, хороший чи відмінний. Рейтингові шкали дозволяють особі, яка проводить опитування, вимірювати та порівнювати набори змінних.

- Якщо ви використовуєте рейтингові шкали, будьте послідовними протягом усього опитування. Використовуйте однакову кількість балів на шкалі та переконайтеся, що значення високого та низького значень залишаються незмінними протягом усього опитування.
- Використовуйте непарну цифру в шкалі оцінок, щоб полегшити аналіз даних. Зміна рейтингової шкали навколо заплутає учасників опитування, що призведе до ненадійних відповідей.
- Обмежте кількість пунктів у запитаннях щодо рейтингу чи шкали рейтингу менше ніж десятьма. Ці запитання можуть стати важкими для читання після десяти варіантів. Більш довгі запитання щодо рейтингу чи рейтингу також можуть спричинити проблеми з відображенням у деяких середовищах.

2.2. Модель проведення онлайнового опитування

2.2.1. Відповіді

Частота відповідей часто досить низька, і існує небезпека того, що вони будуть продовжувати знижуватися через надмірне опитування веб-користувачів.

Джон Кросник стверджує, що наступні три фактори визначають успішність анкетування та ймовірність досягнення гідних рівнів відповіді.

- Респондентська здатність
- Мотивація респондента
- Складність завдання / дизайн опитувальника [12]

Бошняк і Тутен стверджують, що існує принаймні сім способів реагування на опитування в Інтернеті. [13]

Вони встановлюють наступну типологію:

1. Повні відповіді - це ті респонденти, які переглядають усі питання та відповідають на всі питання.
2. Невідповідачі підрозділів - це ті особи, які не беруть участі в опитуванні. Існує дві можливі варіації одиниці, яка не відповідає. Такій особі може бути технічно перешкоджено брати участь, або вона може цілеспрямовано відмовитись після відображення екрана привітання, але до перегляду будь-яких питань.
3. Випадаючі, що кидають відповіді, складаються з осіб, які дають відповіді на ті питання, що відображаються, але звільняються перед тим, як заповнити опитування.
4. Луркери переглядають усі питання опитування, але не відповідають на жодне з них.

5. Залишаються ті, хто кидає навчання, являє собою комбінацію 3 і 4. Такий учасник переглядає деякі питання, не даючи відповіді, але також кидає опитування до того, як дійде до кінця.
6. Невідповідачі, які не відповідають на запитання, переглядають всю анкету, але лише відповідають на деякі запитання.
7. Відсутні елементи, які не відповідають, представляють собою суміш 3 і 6. Особи, що демонструють цю поведінку відповідей, переглядають деякі питання, відповідають на деякі, але не на всі переглянуті питання, а також кидають до закінчення опитування.

2.2.2. Адміністрування

Після того, як анкета розроблена, вона повинна бути введена до відповідної сукупності вибірки для збору даних. [8] Залучення відповідної цільової аудиторії часто вимагає реклами. Для залучення учасників використовуються різні методи

- дошки оголошень
- масові електронні листи
- реклама в комерційних районах
- поштою
- грошові заохочення
- знижки на товари компанії

Це, як правило, допомагає залучити охочих учасників, які в кінцевому підсумку надають якісніші дані на відміну від неохочих.

Розташування адміністрації для анкети в Інтернеті може бути фактором адміністрування, якщо потрібне конкретне середовище. Тихе середовище може знадобитися для питань, які вимагають певної концентрації. [14] Можливо, анкету потрібно буде вводити у відокремленому середовищі, щоб захистити конфіденційну інформацію, надану учасником. [9] У цих випадках також може

знадобитися додати заходи безпеки в програмному забезпеченні. [5] На відміну від цього, анкети в Інтернеті також можуть бути дуже неформальними та розслабленими, і їх можна проводити в комфорті когось удома. [1]

2.2.3. Якість

Якість опитувальника можна виміряти за значенням отриманих даних та задоволенням учасників. [1] Щоб підтримувати високу якість довжини анкети, слід враховувати стислість та послідовність запитань. [14] По-перше, анкети повинні бути лише стільки, скільки їм потрібно. [3] [4] Лаконічності можна досягти шляхом видалення зайвих і нерелевантних питань, що може додати учаснику розчарування, але не мати значення для дослідження. [8] Нарешті, розміщення запитань у логічній послідовності також дає учасникам кращу ментальну карту під час заповнення анкети. [3] Безладне переміщення між випробовуваними та отримання відповідей у неінтуїтивній послідовності може заплутати учасника. [1]

2.2.4. Етика

Етичні питання слід враховувати під час збору даних від цільової аудиторії. Нижче наведено загальні речі, про які слід пам’ятати, розглядаючи права та інтереси учасника. [1]

- Учасники не повинні зобов'язуватися відповідати на будь-яке із запитань. [9]
- Спонукання взяти участь в опитуванні слід використовувати економно. [9]
- Анкети повинні мати можливість бути анонімними. [5]
- Конфіденційність повинна покладатися на певні анкети. Ідентифікація може знадобитися в анкетах, які потребують подальшого вивчення. Хоча в цьому випадку адміністратор може вибрати ідентифікаційні номери, а

не імена. У цьому випадку учасник анкети повинен повністю розуміти, для чого використовується номер і чому він там. [14]

- Запитання повинні мати можливість «не знаю» або варіант, який позначає нейтралітет, щоб учасник відчував, що він / вона має змогу заявляти про незнання або нейтралітет, щоб не було надано неточних даних. [1]
- Запитання не повинні обманювати учасника. Вони повинні бути сформульовані чітко; учасник повинен почуватись комфортно і точно знати, на що він або вона реагує. [7]
- Учасник у більшості випадків повинен знати, чому проводиться анкетування та для чого буде використана інформація. [3]
- У деяких випадках опитувальник повинен переглянути комітет з питань етики або стороння сторона. [1] Це особливо важливо, якщо анкета передбачає надання конфіденційної інформації або тема, яка може спричинити дискомфорт у деяких учасників.

Висновки до розділу 2

Мета даного розділу полягала в розробці моделі онлайн опитування, формуванні системи показників і виявленні основних структурних систем онлайн опитування, їх складових частин та особливостей.

Модель і система показників, розроблені в даному розділі, в подальшому зіграють ключову роль в створенні власної інформаційної системи, адже стануть базовим фундаментом побудови системи і чітким вказанням щодо її структури.

Таким чином, другий розділ бакалаврської роботи дав змогу виробити підхід до розробки майбутньої системи і зрозуміти її загальну структуру.

РОЗДІЛ 3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ПРОВЕДЕННЯ ОНЛАЙНОВОГО АНКЕТУВАННЯ

3.1. Інформаційно-логічна модель проведення онлайнового анкетування

3.1.1. Діаграма варіантів використання

Діаграма використання найпростіша - це представлення взаємодії користувача із системою, яка показує взаємозв'язок між користувачем та різними випадками використання, в яких користувач бере участь. Діаграма випадків використання може ідентифікувати різні типи користувачів системи та різні випадки використання, і часто вона супроводжується також іншими типами діаграм. Варіанти використання представлені кругами або еліпсами.

Незважаючи на те, що сам випадок використання може детально вивчити кожну можливість, діаграма прикладів використання може допомогти забезпечити огляд системи на більш високому рівні. Раніше вже було сказано, що "схеми використання - це принципи вашої системи".

Через їх спрощений характер, схеми використання можуть бути хорошим інструментом комунікації для зацікавлених сторін. Креслення намагаються імітувати реальний світ і дають зацікавленій стороні уявлення про те, як буде розроблена система.

Метою діаграми використання є відображення динамічного аспекту системи. Додаткові схеми та документація можуть бути використані для забезпечення повного функціонального та технічного уявлення про систему. Вони забезпечують спрощене та графічне представлення того, що система насправді повинна робити.

Елементи:

- рамки системи (англ. system border) - прямокутник із назвою у верхніх частинах та еліпсами (прецедентами) всередині. Часто може бути опущено без корисної інформації про полезну інформацію,
- актор (англ. актор) - стилізований людський персонаж, обозначаючий набір ролей користувача (розуміється в широкому змісті: людина, внешняя сутність, клас, інша система), взаємодіючого з деякою сутністю (системною, підсистемою, класом). Актори не можуть бути пов'язані між собою з іншим (за вимкнення відносин щодо обробки / дослідження),
- прецедент - еліпс із надписом, що означає виконувану систематичну дію (може включати можливі варіанти), що призводить до спостерігаємих акторами результатів. Надпис може бути ім'ям або описом (з точки зору актора) того, "що" робить система (а не "як"). Ім’я прецедента зв’язано з неперервним (атомарним) сценарієм - конкретною послідовністю дій, ілюстріруючою повіддю. Під час сценарію актори обмінюються із систематичними повідомленнями. Сценарій може бути приведений на діаграмі прецедентів у відео UML-коментарі. З одним прецедентом може бути пов'язано кілька різних сценаріїв

На рисунках 3.1 і 3.2 зображено діаграми варіантів використання, які описують можливі дії користувача в системі.

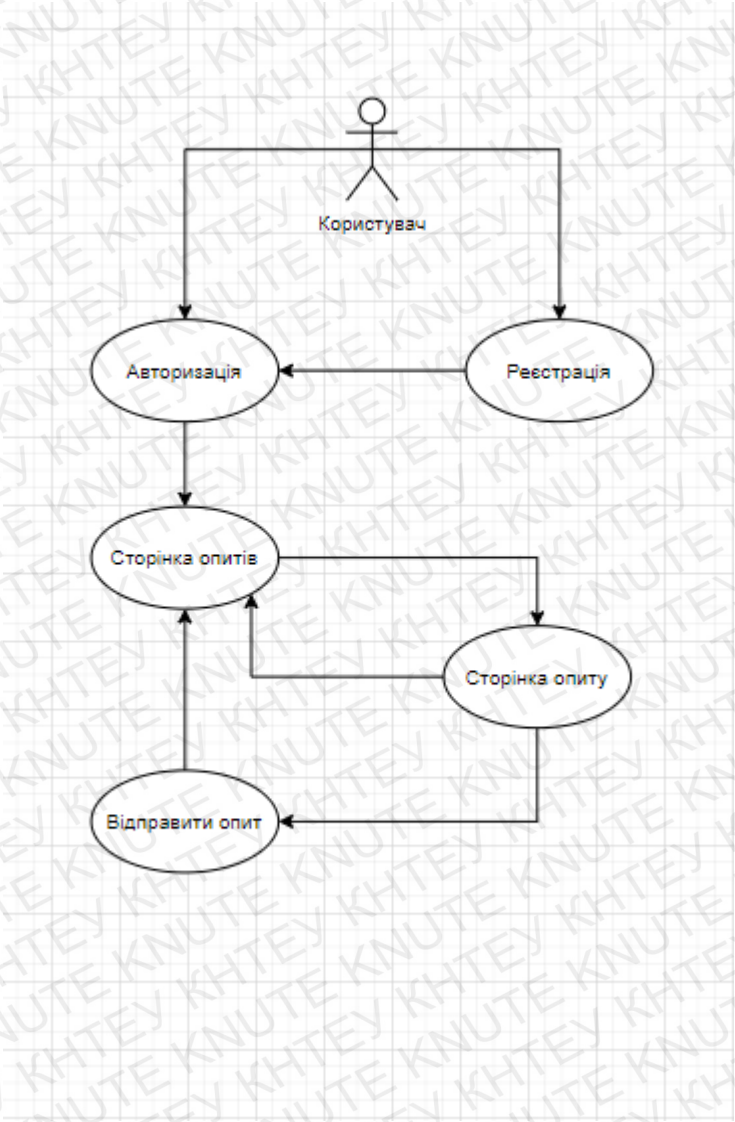


Рис. 3.1 — Діаграма варіантів використання для користувача

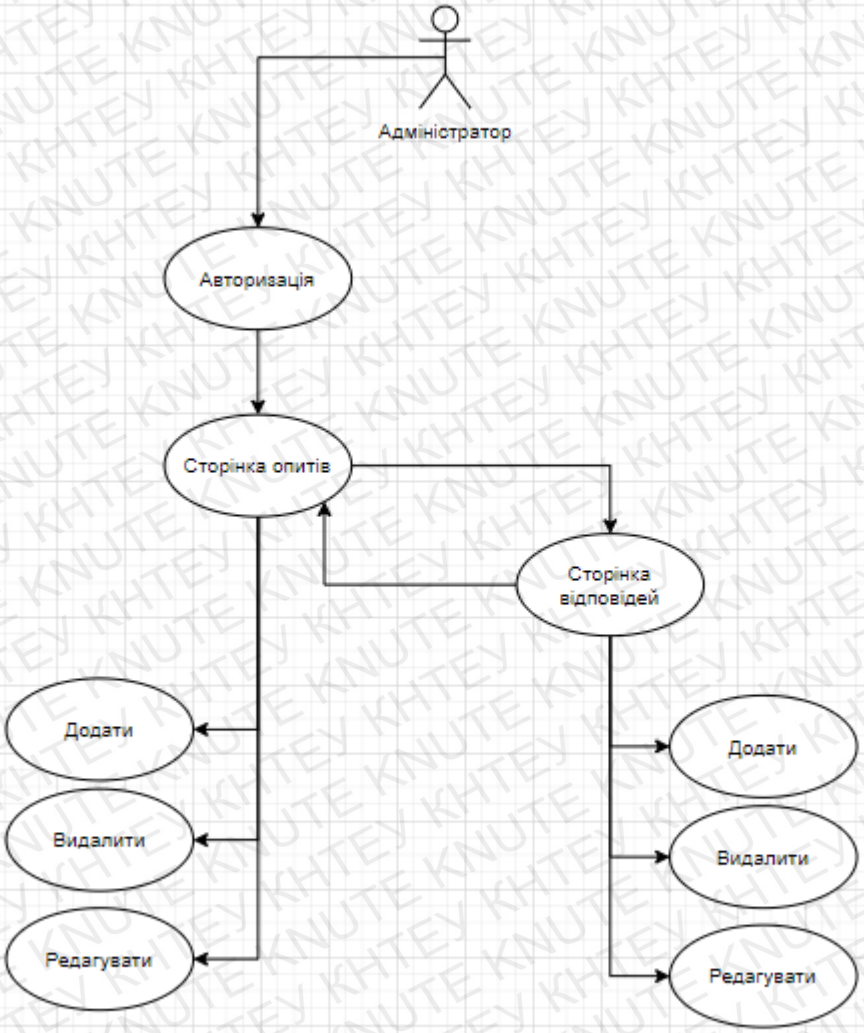


Рис. 3.2 — Діаграма варіантів використання для користувача

3.1.2. Діаграма класів

У програмній інженерії діаграма класів в Уніфікованій мові моделювання (UML) - це тип статичної структурної діаграми, що описує структуру системи, показуючи класи системи, їх атрибути, операції (або методи) та взаємозв'язки між об'єктами.

Діаграма класів є основним будівельним елементом об’єктно-орієнтованого моделювання. Він використовується для загального концептуального моделювання структури програми та для детального моделювання переведення моделей у програмовий код. Діаграми класів також можуть бути використані для моделювання даних. Класи на діаграмі класів представляють як основні елементи, взаємодії в програмі, так і класи, що програмуються.

На схемі класи представлені вікнами, які містять три відділення:

- У верхньому відділенні міститься назва класу. Надруковано жирним шрифтом і відцентровано, а перша літера написана великими літерами.
- Середній відсік містить атрибути класу. Вони вирівняні за лівим краєм, а перша буква мала.
- У нижньому відділенні містяться операції, які може виконувати клас. Вони також вирівняні за лівим краєм, а перша буква - мала.

При проектуванні системи ряд класів ідентифікується та згруповується у схему класів, яка допомагає визначити статичні відносини між ними. При детальному моделюванні класи концептуального проекту часто поділяються на ряд підкласів.

Залежність - це семантичний зв’язок між залежними та незалежними елементами моделі. Він існує між двома елементами, якщо зміни у визначенні одного елемента (сервера або цілі) можуть спричинити зміни для іншого (клієнта або джерела). Ця асоціація є односпрямованою. Залежність відображається у вигляді штрихової лінії з відкритою стрілкою, яка вказує від клієнта до постачальника.

Для подальшого опису поведінки систем ці діаграми класів можуть бути доповнені діаграмою стану або машиною стану UML.

Асоціація представляє родину посилань. Двійкова асоціація (з двома кінцями) зазвичай представляється у вигляді рядка. Асоціація може пов’язувати будь-яку кількість класів. Асоціація з трьома ланками називається потрійною асоціацією. Асоціацію можна назвати, а кінці асоціації можна прикрасити іменами ролей, показниками власності, кратністю, видимістю та іншими властивостями.

Існує чотири різні типи асоціацій: двонаправлена, односпрямована, агрегаційна (включає агрегацію композиції) та рефлексивна. Двонаправлені та односпрямовані асоціації є найбільш поширеними.

Наприклад, клас польоту асоціюється з класом літака двонаправлено. Асоціація представляє статичне відношення, яке ділиться між об'єктами двох класів.

Агрегація є варіантом взаємозв'язку "має"; агрегація є більш конкретною, ніж асоціація. Це асоціація, яка представляє частково цілі або часткові стосунки. Як показано на зображенні, професор "має" клас для викладання. Як тип асоціації, агрегація може бути названа та мати ті самі прикраси, що і асоціація. Однак агрегація не може включати більше двох класів; це має бути бінарна асоціація. Крім того, навряд чи існує різниця між агрегаціями та асоціаціями під час реалізації, і діаграма може взагалі пропустити відносини агрегування. [7]

Агрегація може відбуватися, коли клас є колекцією або контейнером інших класів, але вміщені класи не мають сильної залежності життєвого циклу від контейнера. Вміст контейнера все ще існує, коли контейнер знищений.

В UML він графічно представлений у вигляді порожнистої форми ромба на вміщуючому класі одним рядком, що зв’язує його із вміщеним класом. Сукупність - це семантично розширений об’єкт, який у багатьох операціях трактується як одиниця, хоча фізично він складається з декількох менших об’єктів.

Приклад: Бібліотека та студенти. Тут студент може існувати без бібліотеки, зв'язок між студентом і бібліотекою є агрегацією.

Це вказує на те, що один із двох пов’язаних класів (підклас) вважається спеціалізованою формою іншого (супер тип), а суперклас - узагальненням підкласу. На практиці це означає, що будь-який екземпляр підтипу є також екземпляром суперкласу. Зразкове дерево узагальнень цієї форми зустрічається в біологічній класифікації: людина - це підклас вищих приматів, який є підкласом ссавців тощо. Зв’язок найлегше зрозуміти за допомогою фрази „А - це В” (людина - це ссавець, ссавець - тварина).

Графічне представлення UML узагальнення - це форма порожнистого трикутника на кінці суперкласу рядка (або дерева рядків), що зв’язує його з одним або кількома підтипами.

Відносини узагальнення також відомі як спадщина або відносини "є".

Суперклас (базовий клас) у відносинах узагальнення також відомий як "батьківський", суперклас, базовий клас або базовий тип.

Підтип у відносинах спеціалізації також відомий як "дочірній", підклас, похідний клас, похідний тип, клас успадкування або тип успадкування.

Зверніть увагу, що ці стосунки нічим не схожі на біологічні стосунки батьків та дітей: використання цих термінів надзвичайно поширене, але може ввести в оману.

А - це тип В

Наприклад, "дуб - це тип дерева", "автомобіль - це тип транспортного засобу"

Узагальнення може бути показано лише на діаграмах класів та на діаграмах використання.

При моделюванні UML взаємозв'язок реалізації - це взаємозв'язок між двома елементами моделі, в яких один елемент моделі (клієнт) реалізує

(реалізує або виконує) поведінку, яку вказує інший елемент моделі (постачальник).

Графічне представлення UML реалізації - це порожниста форма трикутника на кінці інтерфейсу штрихової лінії (або дерева рядків), яка з'єднує її з одним або кількома реалізаторами. Проста головка стрілки використовується на кінці інтерфейсу штрихової лінії, що з'єднує її з користувачами. У діаграмах компонентів використовується графічна умова «м'яч і сокет» (реалізатори виставляють кульку або льодяник, тоді як користувачі показують сокет). Реалізації можна показати лише на діаграмах класів або компонентів. Реалізація - це взаємозв'язок між класами, інтерфейсами, компонентами та пакетами, що з'єднує елемент клієнта з елементом постачальника. Зв'язок реалізації між класами / компонентами та інтерфейсами показує, що клас / компонент реалізує операції, пропоновані інтерфейсом.

Залежність - це слабша форма зв'язку, яка вказує на те, що один клас залежить від іншого, оскільки він використовує його в певний момент часу. Один клас залежить від іншого, якщо незалежний клас є змінною параметра або локальною змінною методу залежного класу. Це відрізняється від асоціації, де атрибут залежного класу є екземпляром незалежного класу. Іноді відносини між двома класами дуже слабкі. Вони взагалі не реалізовані зі змінними-членами. Швидше вони можуть бути реалізовані як аргументи функції-члена.

інший. Ці відносини зазвичай описуються як "А має В" (у матері-кота є кошенята, у кошенят - мати-кішка).

Представлення UML асоціації - це лінія, що з'єднує два пов'язані класи. На кожному кінці рядка є додаткові позначення. Наприклад, ми можемо вказати, використовуючи наконечник стрілки, що загострений кінець видно з хвоста стрілки. Ми можемо вказати власність шляхом розміщення кульки, ролі, яку відіграють елементи цього кінця, вказавши ім'я ролі та множинність

екземплярів цієї сутності (діапазон кількості об'єктів, які беруть участь в асоціації з точки зору іншого кінця).

Класи сутності моделюють довгоживучу інформацію, якою обробляє система, а іноді і поведінку, пов'язану з цією інформацією. Їх не слід ідентифікувати як таблиці баз даних чи інших сховищ даних.

Вони намальовані як кола з короткою лінією, прикріпленою до нижньої частини кола. Як варіант, їх можна намалювати як звичайні класи із позначенням стереотипу «сутність» над назвою класу.

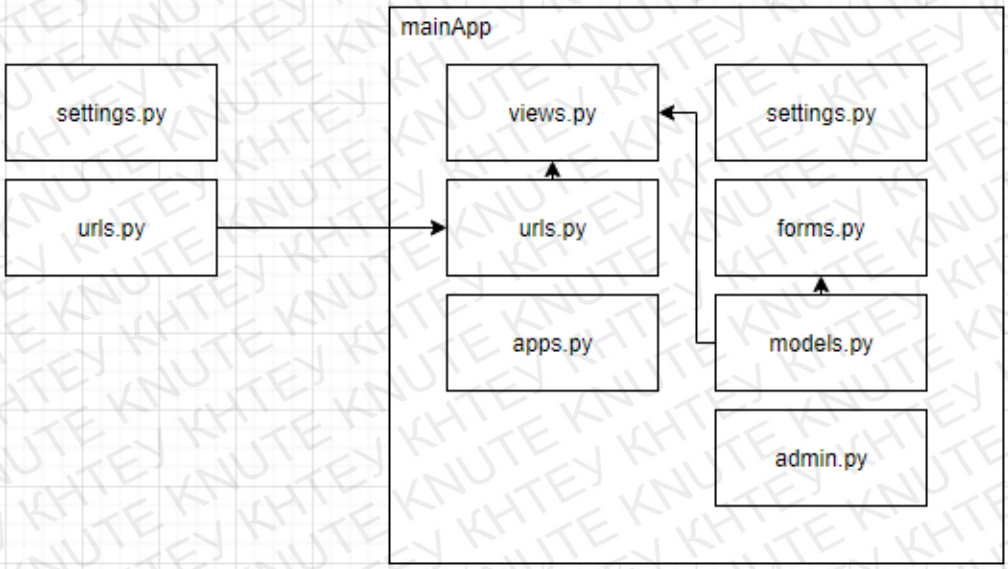


Рис. 3.2 — Діаграма класів

3.2. Програмна реалізація проведення онлайнового анкетування

3.2.1. Мова програмування Python

Python - інтерпретована мова програмування високого рівня та загального призначення. Філософія дизайну Python наголошує на читабельності коду завдяки помітному використанню значних відступів. Його мовні конструкції та об’єктно-орієнтований підхід мають на меті допомогти програмістам написати чіткий логічний код для малих та великих проектів.

Python динамічно набирається та збирається сміття. Він підтримує декілька парадигм програмування, включаючи структуроване (зокрема, процедурне), об'єктно-орієнтоване та функціональне програмування. Python часто описують як мову, що включає батареї, завдяки своїй повній стандартній бібліотеці.

Python стабільно входить до числа найпопулярніших мов програмування.

Python був задуманий наприкінці 1980-х Гвідо ван Россумом з Centrum Wiskunde & Informatica (CWI) в Нідерландах як спадкоємець мови програмування ABC, натхненної SETL, здатною обробляти винятки та взаємодіяти з Операційна система Amoeba. Його реалізація розпочалась у грудні 1989 р. Ван Россум взяв на себе виключну відповідальність за проект, як провідний розробник, до 12 липня 2018 року, коли він оголосив про свої "постійні канікули", виконуючи обов'язки як «Доброзичливий життєвий диктатор» Пітона, титул, який йому надала спільнота Пітона, щоб відобразити його тривалий час. строкове зобов'язання як головний керівник проекту. Зараз він ділиться своїм керівництвом як член наглядової ради з п'яти осіб. У січні 2019 року активні розробники ядра Python обрали Бретта Кеннона, Ніка Коглана, Барі Варшаву, Керол Віллінг та Ван Россума членами керівної ради з п’яти членів. З тих пір Гвідо ван Россум відкликав свою кандидатуру на посаду Керівної ради 2020 року.

Python 2.0 був випущений 16 жовтня 2000 року з багатьма основними новими можливостями, включаючи збирач сміття, що виявляє цикл, та підтримку Unicode.

Python 3.0 був випущений 3 грудня 2008 року. Це був серйозний перегляд мови, який не є повністю сумісним із зворотною стороною. Багато його основних функцій були передані до версій Python 2.6.x та 2.7.x. Випуски Python 3 включають утиліту 2to3, яка автоматизує (принаймні частково) переклад коду Python 2 на Python 3.

Дата закінчення терміну служби Python 2.7 спочатку була встановлена на 2015 рік, а потім перенесена на 2020 рік через занепокоєння тим, що велика кількість існуючого коду не може бути легко перенесена на Python 3. Більше виправлень безпеки та інших удосконалень для нього випускати не буде. У кінці життя Python 2 підтримується лише Python 3.6.x і пізніші версії.

Python 3.9.2 та 3.8.8 були прискорені, оскільки всі версії Python мали проблеми із безпекою, що призвело до можливого віддаленого виконання коду та отруєння веб-кешу.

Python - мова програмування з багатьма парадигмами. Об'єктно-орієнтоване програмування та структуроване програмування повністю підтримуються, і багато його функцій підтримують функціональне програмування та аспектно-орієнтоване програмування (в тому числі метапрограмуванням та метаоб'єктами (магічні методи)). Багато інших парадигм підтримуються за допомогою розширень, включаючи дизайн за контрактом та логічне програмування.

Python використовує динамічне введення тексту та комбінацію підрахунку посилань та збирач сміття, що визначає цикл, для управління пам’яттю. Він також має динамічну роздільну здатність імен (пізнє прив'язування), яка зв'язує імена методів та змінних під час виконання програми.

Дизайн Python пропонує певну підтримку функціонального програмування за традицією Lisp. Він має функції фільтрування, картографування та зменшення; перелічити розуміння, словники, набори та вирази генераторів. Стандартна бібліотека має два модулі (itertools та functools), що реалізують функціональні інструменти, запозичені у Haskell та Standard ML.

Основна філософія мови узагальнена в документі Zen of Python (PEP 20), який включає такі афоризми, як:

- Красиве - краще, ніж потворне.
- Явне краще, ніж неявне.
- Просте - краще, ніж складне.
- Складний - це краще, ніж складний.
- Підрахунок читабельності.

Замість того, щоб усі його функціональні можливості були вбудовані в його ядро, Python був розроблений таким чином, щоб бути дуже розширюваним (з модулями). Ця компактна модульність зробила його особливо популярним як засіб додавання програмованих інтерфейсів до існуючих додатків. Бачення Ван Россума щодо малої основної мови з великою стандартною бібліотекою та легко розширюваним перекладачем випливало з його розчарувань у ABC, який підтримував протилежний підхід. Python прагне до більш простого, менш захараченого синтаксису та граматики, одночасно надаючи розробникам можливість вибору методології кодування. На відміну від девізу Perl "існує більше ніж один спосіб зробити це", Python охоплює "повинен бути один - і бажано лише один - очевидний спосіб зробити це" філософія дизайну. [69] Алекс Мартеллі, співробітник Фонду програмного забезпечення Python і автор книги про Python, пише, що "Описувати щось як" розумне "не вважається компліментом у культурі Python" .

Розробники Python прагнуть уникнути передчасної оптимізації та відкидають виправлення для некритичних частин еталонної реалізації CPython,

які пропонують незначне збільшення швидкості ціною ясності. Коли швидкість важлива, програміст Python може переміщати критично важливі для часу функції до модулів розширення, написаних мовами, такими як C, або використовувати PyPy, своєчасний компілятор. Також доступний Cython, який перекладає скрипт Python на C і здійснює прямі виклики API рівня C в інтерпретатор Python.

Важливою метою розробників Python є підтримка його задоволення від використання. Це відображається в назві мови - данина поваги британській гумористичній групі Монті Пайтон - і в іноді грайливих підходах до навчальних посібників та довідкових матеріалів, таких як приклади, що стосуються спаму та яєць (із відомого етюду Монті Пайтона) стандартних foo and bar.

Поширеним неологізмом у спільноті Python є пітонічний, який може мати широкий діапазон значень, пов’язаних зі стилем програми. Сказати, що код пітонічний, означає сказати, що він добре використовує ідіоми Python, що він природний або демонструє вільну мову, що відповідає мінімалістичній філософії Python та наголосу на читабельності. На відміну від цього, код, який важко зрозуміти або читається як груба транскрипція з іншої мови програмування, називається непітонічним.

Користувачів та шанувальників Python, особливо тих, хто вважається обізнаним чи досвідченим, часто називають Pythonistas. Python має бути мовою, яка легко читається. Його форматування візуально не захаращене, і в ньому часто використовуються англійські ключові слова, де інші мови використовують розділові знаки. На відміну від багатьох інших мов, він не використовує фігурні дужки для розмежування блоків, а крапка з комою після операторів дозволена, але рідко, якщо взагалі використовується. Він має менше синтаксичних винятків та особливих випадків, ніж C або Pascal.

Python використовує пробіли, а не фігурні дужки або ключові слова, для відмежування блоків. Збільшення відступу відбувається після певних

тверджень; зменшення відступу означає кінець поточного блоку. Таким чином, візуальна структура програми точно відображає семантичну структуру програми. Цю функцію іноді називають правилом "поза стороною", яке мають деякі інші мови, але в більшості мов відступ не має семантичного значення. Рекомендований розмір відступу - чотири пробіли.

Оператори Python включають (серед іншого):

- Оператор присвоєння, використовуючи один знак рівності =.
- Оператор if, який умовно виконує блок коду, разом з else та elif (скорочення else-if).
- Оператор for, який переглядає ітерабельний об’єкт, захоплюючи кожен елемент до локальної змінної для використання вкладеним блоком.
- Оператор while, який виконує блок коду, доки його умова відповідає істині.
- Оператор try, що дозволяє вилученням, що містяться у вкладеному блоці коду, ловити та обробляти за винятком речень; він також гарантує, що код очищення в блоці нарешті завжди буде виконуватися незалежно від того, як блок виходить.
- Оператор raise, що використовується для отримання вказаного винятку або повторного підняття спійманого винятку.
- Оператор class, який виконує блок коду та приєднує його локальний простір імен до класу для використання в об'єктно-орієнтованому програмуванні.
- Оператор def, що визначає функцію або метод.
- Оператор with з Python 2.5, випущений у вересні 2006 р. [82], який охоплює блок коду в контекстному менеджері (наприклад, отримання блокування перед запуском блоку коду та звільнення

блокування після цього, або відкриття файлу, а потім закриваючи його), дозволяючи поведінку, схожу на отримання ресурсів - це ініціалізація (RAII), і замінює загальну ідіому try / нарешті.

- Оператор break, виходить із циклу.
- Оператор continue, пропускає цю ітерацію і продовжує наступний пункт.
- Оператор del видаляє змінну, що означає, що посилання з імені на значення видаляється, і спроба використання цієї змінної спричинить помилку. Видалену змінну можна перепризначити.
- Оператор pass, яка виконує функцію NOP. Це синтаксично потрібно для створення порожнього блоку коду.
- Оператор assert, який використовується під час налагодження для перевірки умов, які мають застосовуватися.
- Оператор yield, який повертає значення з функції генератора. З Python 2.5, yield також є оператором. Ця форма використовується для реалізації спільних програм.
- Оператор return, який використовується для повернення значення з функції.
- Оператор import, який використовується для імпорту модулів, функції чи змінні яких можна використовувати в поточній програмі. Є три способи використання імпорту: імпорт <ім'я модуля> [як <псевдонім>] або з <ім'я модуля> імпорт * або з <ім'я модуля> імпорт <визначення 1> [як <псевдонім 1>], <визначення 2> [як <псевдонім 2>], ...

Оператор присвоєння (=) діє шляхом прив'язки імені як посилання на окремий динамічно розподілений об'єкт. Потім змінні можуть бути відскочені в будь-який час до будь-якого об’єкта. У Python ім'я змінної є загальним власником посилання і не має фіксованого типу даних, пов’язаного з ним.

Однак у даний момент часу змінна буде посилатися на якийсь об'єкт, який матиме тип. Це називається динамічним набором тексту і протиставляється статично набраним мовам програмування, де кожна змінна може містити лише значення певного типу.

Python не підтримує оптимізацію хвостових викликів або першокласні продовження, і, за словами Гвідо ван Россума, він ніколи не буде. Однак краща підтримка подібних до корутинів функціональних можливостей надається в 2.5, розширюючи генератори Python. До 2.5 генератори були ледачими ітераторами; інформація передавалась в односпрямованому напрямку з генератора. З Python 2.5 можна передавати інформацію назад у функцію генератора, а з Python 3.3 інформацію можна передавати через кілька рівнів стека.

Такі бібліотеки, як NumPy, SciPy та Matplotlib, дозволяють ефективно використовувати Python у наукових обчисленнях, а спеціалізовані бібліотеки, такі як Biopython та Astropy, забезпечують функціональність, що залежить від домену. SageMath - математичне програмне забезпечення з інтерфейсом ноутбука, програмоване на Python: його бібліотека охоплює багато аспектів математики, включаючи алгебру, комбінаторику, числову математику, теорію чисел і числення. OpenCV має палітурні прив'язки з багатим набором функцій для комп'ютерного зору та обробки зображень.

Python зазвичай використовується в проектах штучного інтелекту та машинному навчанні за допомогою таких бібліотек, як TensorFlow, Keras, Pytorch та Scikit-learn. Як мова сценаріїв з модульною архітектурою, простим синтаксисом та інструментами обробки багатого тексту, Python часто використовується для обробки природної мови. Python успішно вбудований у багато програмних продуктів як мову сценаріїв, включаючи програмне забезпечення методом скінченних елементів, таке як Abaqus, 3D-параметричний моделер, такий як FreeCAD, пакети 3D-анімації, такі як 3ds Max, Blender, Cinema 4D, Lightwave, Houdini, Maya, modo , MotionBuilder, Softimage,

композитор візуальних ефектів Nuke, програми для 2D-зображень, такі як GIMP, Inkscape, Scribus і Paint Shop Pro, та музичні нотатні програми, такі як автори та капели. Налагоджувач GNU використовує Python як гарний принтер для показу складних структур, таких як контейнери C ++. Esri пропагує Python як найкращий вибір для написання сценаріїв в ArcGIS. Він також використовувався в декількох відеоіграх і був прийнятий як перша з трьох доступних мов програмування в Google App Engine, інші дві - Java і Go.

Багато операційних систем включають Python як стандартний компонент. Він постачається з більшістю дистрибутивів Linux, AmigaOS 4 (з використанням Python 2.7), FreeBSD (як пакет), NetBSD, OpenBSD (як пакет) та macOS і може використовуватися з командного рядка (терміналу). Багато дистрибутивів Linux використовують інсталятори, написані на Python: Ubuntu використовує інсталятор Ubiquity, тоді як Red Hat Linux і Fedora використовують інсталятор Anaconda. Gentoo Linux використовує Python у своїй системі управління пакунками Portage.

Python широко використовується в галузі інформаційної безпеки, в тому числі в розробці експлойтів.

Більшість програм Sugar для одного ноутбука на дитину XO, які зараз розробляються в Sugar Labs, написані на Python. Проект одноплатної комп'ютерної програми Raspberry Pi прийняв Python як основну мову програмування користувачів.

LibreOffice включає Python і має намір замінити Java на Python. Його постачальник сценаріїв Python є основною функцією з версії 4.0 від 7 лютого 2013 року.

Вихоядчи зі сфери використання і величезного функціоналу мови Python, який був описаний вище, можна зробити висновок про його мультифунціональність. Саме через це дана мова програмування була обрана для виконання даної роботи.

3.2.2. Фреймворк Django

Django - це безкоштовний веб-фреймворк на основі Python, що відповідає архітектурному зразку model-template-views (MTV). [9] [10] Він підтримується Django Software Foundation (DSF), американською незалежною організацією, створеною як некомерційна організація 501 (c) (3).

Основна мета Django - полегшити створення складних веб-сайтів, керованих базами даних. Структура підкреслює багаторазовість та "підключеність" компонентів, менше коду, низьку зв'язок, швидкий розвиток та принцип "не повторюватися". [11] Python використовується всюди, навіть для налаштувань, файлів та моделей даних. Django також надає додатковий адміністративний інтерфейс створення, читання, оновлення та видалення, який генерується динамічно за допомогою самоаналізу та налаштовується за допомогою моделей адміністратора.

Деякі добре відомі сайти, які використовують Django, включають PBS, [12] Instagram, [13] Mozilla, [14] The Washington Times, [15] Disqus, [16] Bitbucket, [17] і Nextdoor. [18]

3.2.3. База даних SQLite

Бази даних використовуються для структурованого зберігання даних, а SQLite - це популярний формат баз даних, що з'являється у багатьох мобільних системах, а також традиційних операційних системах. SQLite - це технологічна бібліотека, яка реалізує самостійний, безсерверний, механізм баз даних SQL з нульовою конфігурацією. Код SQLite знаходиться у відкритому доступі і, отже, безкоштовний для будь-яких цілей, комерційних або приватних.

SQLite - це вбудований механізм баз даних SQL. На відміну від більшості інших баз даних SQL, SQLite не має окремого серверного процесу. SQLite читає та пише безпосередньо на звичайні файли диска. Повна база даних SQL з

кількома таблицями, індексами, тригерами та поданнями міститься в одному дисковому файлі. Як правило, SQLite працює швидше, чим більше пам'яті ви йому надаєте. Тим не менше, продуктивність, як правило, досить хороша навіть у середовищах з низьким обсягом пам'яті. Залежно від того, як він використовується, SQLite може бути швидшим, ніж прямий ввід / вивід файлової системи [33].

SQLite дуже ретельно перевіряється перед кожним випуском і має репутацію дуже надійного. Більшість вихідних кодів SQLite присвячені виключно тестуванню та верифікації. Автоматизований набір тестів запускає мільйони і мільйони тестових випадків, що включають сотні мільйонів окремих операторів SQL, і забезпечує 100% покриття тестуванням філій. SQLite витончено реагує на помилки розподілу пам'яті та помилки вводу-виводу диска. Все це перевіряється автоматизованими тестами за допомогою спеціальних тестових джгутів, які імітують відмови системи. Звичайно, навіть при всьому цьому тестуванні все ще існують помилки. Але на відміну від деяких подібних проектів (особливо комерційних конкурентів), SQLite відкрито та чесно ставиться до всіх помилок і надає списки помилок та щохвилинні хронології змін коду.

Отож, для наглядного огляду переваг взято одного з конкурентів SQLite, а саме SQL Server, і в свою чергу було здійснено їхнє коротке порівняння:

- Модель ціноутворення:
    - SQLite безкоштовний;
    - SQL Server - це рекламний ролик із обмеженою безкоштовною версією;
- Серверні операційні системи:
    - SQLite не потребує сервера;
    - SQL Server працює на Linux та Windows;
- API та інші методи доступу:

    - SQLite підтримує такі драйвери:
        - ADO.NET;
        - JBDC;
        - ODBC;
    - SQL Server підтримує:
        - OLE DB;
        - ADO.NET;
        - JBDC;
        - ODBC;
        - Табличний потік даних (TDS);
- Підтримувані мови програмування:
    - SQL Server підтримує такі мови програмування (C#, C++, Delphi, Go, Java, JavaScript (Node.js), PHP, Python, R, Ruby, Visual Basic);
    - SQLite підтримує майже будь-які мови програмування, про які ви можете подумати (Actionscript, Ada, Basic, C, C#, C++, D, Delphi, Forth, Fortran, Haskell, Java, JavaScript, Lisp, Lua, MatLab, Objective-C, OCaml, Perl, PHP, PL/SQL, Python, R, Ruby, Scala, Scheme, Smalltalk, Tcl);
- Підтримка збереженої процедури:
    - SQLite не підтримує збережену процедуру;
    - SQL Server має його з мовами Transact SQL та .NET;
- Методи розділення:
    - SQLite не підтримує;
    - У SQL Server таблиці можуть розподілятися між кількома файлами (горизонтальне розділення);
- Методи реплікації:
    - SQLite не підтримує;

- SQL Server підтримує;

- Контроль доступу користувачів:
  - SQLite не має концепції контролю доступу користувачів;

SQL Server має чіткі права доступу відповідно до стандарту SQL.

3.2.4. Розробка графічного інтерфейсу користувача

Графічний інтерфейс користувача складається з 4 основних сторінок, а саме:

- Сторінка авторизації (представляє собою власну реалізацію стандартної для Django форми авторизації, яка містить в собі поля для вводу логіну і паролю, кнопку входу та посилання на сторінку реєстрації)
- Сторінка реєстрації (представляє собою власну реалізацію стандартної для Django форми реєстрації, яка містить в собі поля для вводу логіну і паролю, пошти, кнопку реєстрації та посилання на сторінку авторизації)
- Сторінка активних опитів (містить в собі список активних опитів, які являють собою список значень з моделі опитів, представлених у вигляді назв. При виклику відповідного view функція спочатку дістає із БД всі данні, а потім передає їх у шаблон під час рендеру, який в свою чергу за допомогою циклу Django, вбудованого в HTML-код, виводить почергово назву опиту для кожного пункту списку)
- Сторінка опиту (працює аналогічно до сторінки активних опитів, за виключенням більш складної структури виводу (подвійні вкладені цикли) та більшої кількості полів)

Скріншоти даних сторінок можна знайти в додатку Б.

ВИСНОВКИ

Метою даної роботи було розроблення програмного забезпечення для проведення онлайнового анкетування.

Для виконання поставленої мети було виконано наступні завдання:

- Проаналізувати предметну область
- Обрати засоби реалізації
- Створити діаграму варіантів використання
- Створити діаграму класів
- Спроектувати графічний інтерфейс користувача

Узагальнення результатів теоретичної та дослідно-експериментальної роботи дає підстави для таких висновків:

Проаналізовано теоретичні підходи та стан наукової розробленості питання розробки систем проведення онлайнових опитувант, що свідчать недостатність розробленості питання та наявність великих пробілів в дослідженні.

Розроблено модель Web-системи для онлайнового опитування яка складається з модулів публікації опитування, реєстрації та авторизації користувачів, проходження опитування, відправки бланку відповідей адміністратору.

З’ясовано особливості використання програмно-апаратних засобів Web-системи для онлайнового опитування, що являють собою чітку необхідність в постійні підтримці адекватного функціонування серверу веб-додатку з метою постійно працездатності всієї системи.

Розроблено та експериментально перевірено технологію розробки Web-системи для онлайнового опитування мовою програмування Python, що дало змогу забезпечити можливість використовувати отриману систему в будь-якій

сфері, наприклад при зборі статистичних даних, прийняті рішень всередині групи, або зборі психологічних портретів користувачів.

Завдяки чіткому виконанню завдань, поставлених на початку роботи, в результаті виконання роботи було отримано повноцінну систему, що здатна виконувати закладений в неї функціонал та готова до використання в реальних умовах.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Gault, RH (1907). "A history of the questionnaire method of research in psychology". Research in Psychology. 14 (3): 366–383. doi:10.1080/08919402.1907.10532551.
2. A copy of the instrument was published in the Journal of the Statistical Society, Volume 1, Issue 1, 1838, pages 5–13. "Fourth Annual Report of the Council of the Statistical Society of London". JSTOR i315562.
3. "The Roma have a much younger population". OECD Economic Surveys: Slovak Republic. 2019-02-05. doi:10.1787/d8c7c39a-en. ISBN 9789264311350. ISSN 1999-0588.
4. "questions-answers-the-international-criminal-court-may-2010". doi:10.1163/2210-7975_hrd-0162-0046.
5. Fox, Adam, Parochial Queries: Printed Questionnaires and the Pursuit of Natural:Knowledge in the British Isles, 1650–1800, Edinburgh University
6. Smedts HP, de Vries JH, Rakhshandehroo M, et al. (February 2009). "High maternal vitamin E intake by diet or supplements is associated with congenital heart defects in the offspring". BJOG. 116 (3): 416–23. doi:10.1111/j.1471-0528.2008.01957.x. PMID 19187374. S2CID 22276050.
7. Hogervorst, J. G.; Schouten, L. J.; Konings, E. J.; Goldbohm, R. A.; Van Den Brandt, P. A. (2007). "A Prospective Study of Dietary Acrylamide Intake and the Risk of Endometrial, Ovarian, and Breast Cancer". Cancer Epidemiology, Biomarkers & Prevention. 16 (11): 2304–2313. doi:10.1158/1055-9965.EPI-07-0581. PMID 18006919. Retrieved 2013-02-18.
8. Mellenbergh, G.J. (2008). Chapter 10: Tests and Questionnaires: Construction and administration. In H.J. Adèr & G.J. Mellenbergh (Eds.) (with contributions by D.J. Hand), Advising on Research Methods: A consultant's companion (pp. 211–236). Huizen, The Netherlands: Johannes van Kessel Publishing.

9. Burns, A. C., & Bush, R. F. (2010). Marketing Research. Upper Saddle River, NJ: Pearson Education.
10. Robinson, M. A. (2018). Using multi-item psychometric scales for research and practice in human resource management. Human Resource Management, 57(3), 739–750. https://dx.doi.org/10.1002/hrm.21852 (open-access)
11. Kaplan, R. M., & Saccuzzo, D. P. (2009). Psychological testing: Principles, applications, and issues. Belmont, CA: Wadsworth
12. Alwin, D. F. (2007). Margins of error: A study of reliability in survey measurement. Hoboken, Wiley
13. Saris, W. E. and Gallhofer, I. N. (2014). Design, evaluation, and analysis of questionnaires for survey research. Second Edition. Hoboken, Wiley.
14. Moser, Claus Adolf, and Graham Kalton. "Survey methods in social investigation." Survey methods in social investigation. 2nd Edition (1971).

ДОДАТКИ

Додаток А. Лістинг програмного коду

```
from django.db import models
from django.contrib.auth.models import User
class questionnaireData(models.Model):
    questionnaire_name = models.TextField('Название опросника')
    questions_text = models.TextField('Вопросы')
    questions_answers = models.TextField('Ответы')
    def __str__(self):
        return self.questionnaire_name
    class Meta:
        verbose_name = 'Опрос'
        verbose_name_plural = 'Опросы'
class questionnaireAnswers(models.Model):
    questionnaire_name_fk = models.ForeignKey(questionnaireData, on_delete=models.CASCADE)
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    answers = models.TextField('Ответы')
    def __str__(self):
        return self.user.username + "/" + self.questionnaire_name_fk.questionnaire_name
    class Meta:
        verbose_name = 'Ответ'
        verbose_name_plural = 'Ответы'
from django.contrib import admin
from .models import questionnaireAnswers, questionnaireData
admin.site.register(questionnaireAnswers)
admin.site.register(questionnaireData)
from django.shortcuts import render, redirect
from django.contrib.auth import authenticate, login, logout
from django.contrib import messages
from .forms import CreateUserForm
from .models import questionnaireAnswers, questionnaireData
from django.contrib.auth.models import User
def loginPage(request):
    if request.method == 'POST':
        username = request.POST.get('username')
        password = request.POST.get('password')
        user = authenticate(request, username=username, password=password)

        if user is not None:
            login(request, user)
            return redirect('index')
        else:
            messages.info(request, 'Имя пользователя или пароль введены неверно')
    context = {}
    return render(request, 'login.html', context)

def regist(request):
    form = CreateUserForm
    if request.method == "POST":
        form = CreateUserForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('login')
    context = {'form': form}
    return render(request, 'register.html', context)

def logoutUser(request):
    logout(request)
    return redirect('login')
```

```
def index(request):
    if request.user.is_authenticated:
        questionnaire_list = questionnaireData.objects.all()
        return render(request, 'index.html', {'questionnaire_list': questionnaire_list})
    else:
        return redirect('login')

def questionnaire_page(request, name):
    questionnaireCurr = questionnaireData.objects.get(questionnaire_name=name)
    questName = questionnaireCurr.questionnaire_name
    questions = questionnaireCurr.questions_text.split('/')
    tempAns = questionnaireCurr.questions_answers.split('/')
    tempQA = []
    for i in range(len(questions)):
        tempQA.append(questions[i] + '*' + tempAns[i])

    quest_ans = []
    for i in range(len(tempQA)):
        quest_ans.append(tempQA[i].split('*'))
    answers = []
    if request.method == 'POST':
        for i in range(1, len(quest_ans) + 1):
            answers.append(request.POST.get('quest' + str(i)))
        addAnswersToDB(answers, questName, request)
        return redirect('index')
    return render(request, 'questionnaire_page.html', {'questName': questName, 'quest_ans': quest_ans})

def addAnswersToDB(answers, name, req):
    quest = questionnaireData.objects.get(questionnaire_name=name)
    strAnsw = ""
    for str in answers:
        strAnsw += str + "\n"
    answ = questionnaireAnswers(questionnaire_name_fk=quest, user=req.user, answers=strAnsw)
    answ.save()

from django.urls import path

from . import views

urlpatterns = [
    path('', views.index, name='index'),
    path('register/', views.regist, name='register'),
    path('login/', views.loginPage, name='login'),
    path('logout/', views.logoutUser, name='logout'),
    path('questionnaire_page/<str:name>', views.questionnaire_page, name='questionnaire_page'),

]
<!DOCTYPE html>
<html>

<head>
        <title>Авторизация</title>
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
        <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.1/css/all.css" integrity="sha384-
gfdkjb5BdAXd+lj+gudLWI+BXq4IuLW5IT+brZEZsLFm++aCMlF1V92rMkPaX4PP" crossorigin="anonymous">

        <style>
                body,
                html {
                        margin: 0;
                        padding: 0;
                        height: 100%;
                        background: #7abecc !important;
```

```
        }
        .user_card {
                width: 350px;
                margin-top: auto;
                margin-bottom: auto;
                background: #74cfbf;
                position: relative;
                display: flex;
                justify-content: center;
                flex-direction: column;
                padding: 10px;
                box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
                -webkit-box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
                -moz-box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
                border-radius: 5px;

        }

        .form_container {
                margin-top: 20px;
        }

        #form-title{
                color: #fff;

        }

        .login_btn {
                width: 100%;
                background: #33ccff !important;
                color: white !important;
        }
        .login_btn:focus {
                box-shadow: none !important;
                outline: 0px !important;
        }
        .login_container {
                padding: 0 2rem;
        }
        .input-group-text {
                background: #f7ba5b !important;
                color: white !important;
                border: 0 !important;
                border-radius: 0.25rem 0 0 0.25rem !important;
        }
        .input_user,
        .input_pass:focus {
                box-shadow: none !important;
                outline: 0px !important;
        }

        #messages{
                background-color: grey;
                color: #fff;
                padding: 10px;
                margin-top: 10px;
        }
    </style>

</head>
<body>
    <div class="container h-100">
        <div class="d-flex justify-content-center h-100">
            <div class="user_card">
                <div class="d-flex justify-content-center">
```

```
                        <h3 id="form-title">АВТОРИЗАЦИЯ</h3>
                    </div>
                    <div class="d-flex justify-content-center form_container">
                        <form method="POST" action="">
            {% csrf_token %}
                            <div class="input-group mb-3">
                                <div class="input-group-append">
                                    <span class="input-group-text"><i class="fas fa-
user"></i></span>
                                </div>
                                <input type="text" name="username" placeholder="Логин..."
class="form-control">
                            </div>
                            <div class="input-group mb-2">
                                <div class="input-group-append">
                                    <span class="input-group-text"><i class="fas fa-
key"></i></span>
                                </div>
                                    <input type="password" name="password"
placeholder="Пароль..." class="form-control" >
                            </div>

                            <div class="d-flex justify-content-center mt-3 login_container">
                                    <input class="btn login_btn" type="submit"
value="Войти">
                                </div>
                        </form>

                    </div>
        {% for message in messages %}
          <p id="message">{{message}}</p>
        {% endfor %}
                    <div class="mt-4">
                        <div class="d-flex justify-content-center links">
                            Нет аккаунта? <a href="{% url 'register' %}" class="ml-
2">Зарегистрироваться</a>
                        </div>

                    </div>
                </div>
            </div>
        </div>
</body>

</html>
<!DOCTYPE html>
<html lang = "ru">
  <head>
        <meta charset="utf-8">

    <title>
        Опросы
    </title>

    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.1/css/all.css" integrity="sha384-
gfdkjb5BdAXd+lj+gudLWI+BXq4IuLW5IT+brZEZsLFm++aCMlF1V92rMkPaX4PP" crossorigin="anonymous">

    {% load static %}
        <link rel="stylesheet" type="text/css" href="{% static 'index/css/style.css' %}">
  <style>
```

```
                }

                ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  overflow: hidden;
  background-color: #333;
}

li {
  float: left;
  border-right:1px solid #bbb;
}

li:last-child {
  border-right: none;
}

li a {
  display: block;
  color: white;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
}

li a:hover:not(.active) {
  background-color: #111;
}

.active {
  background-color: #4CAF50;
}
.usrName {
float:right;
vertical-align:5px;
color:white;
display: block;
text-align: center;
padding: 14px 16px;
}
        </style>

  </head>
  <body>
<ul>
  <li><a href="{% url 'index' %}">Главная</a></li>
    <li><a href="{% url 'admin:index' %}">Администрирование</a></li>
    <li  style="float:right" st><a href="{% url 'logout' %}">Выйти</a></li>
  <li class="usrName">Добро пожаловать, {{request.user}}</li>

</ul>

  <div class="container h-100">
                <div class="d-flex justify-content-center h-100">
                        <div class="user_card">
                                <div class="d-flex justify-content-center">


                                        <h3 id="form-title">Открытые опросы</h3>

                                </div>

  {% if questionnaire_list %}
```

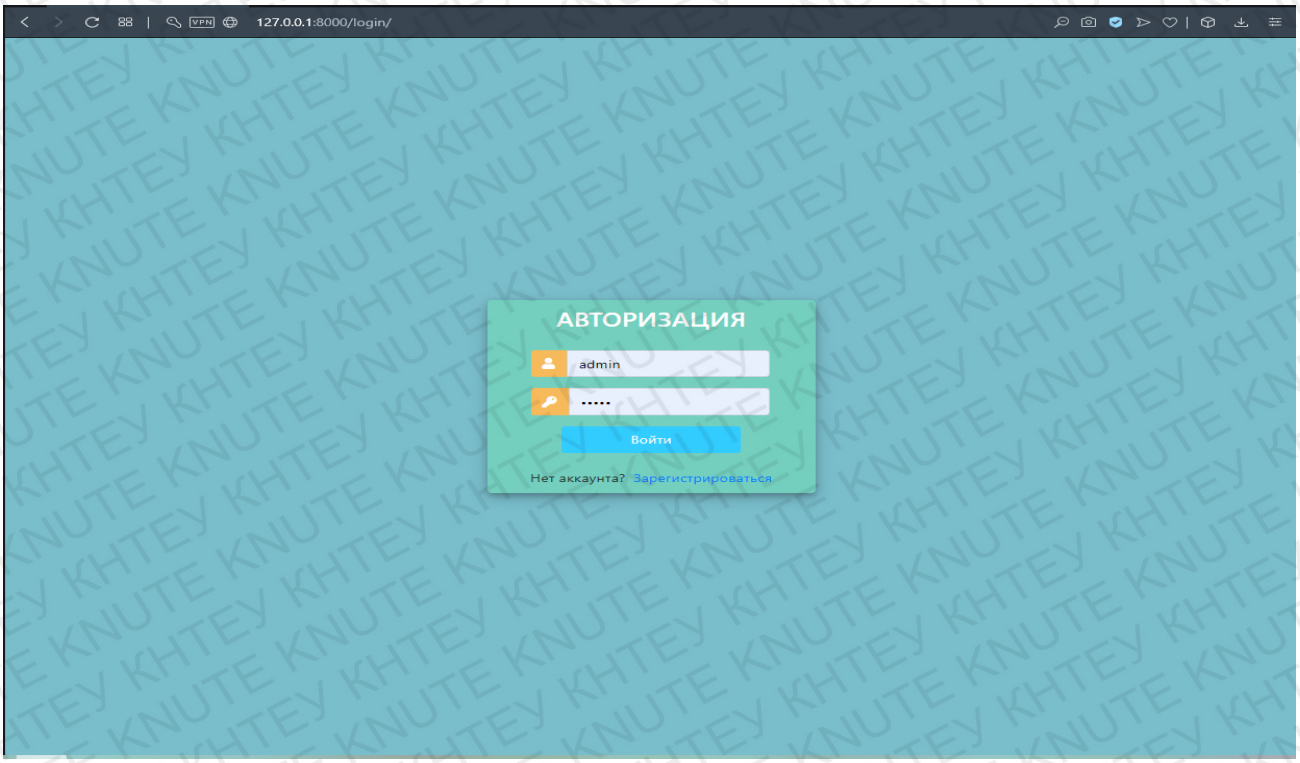
```
  {% for a in questionnaire_list %}

    <h4 align="center"> <a href="{% url 'questionnaire_page' a.questionnaire_name %}">{{a.questionnaire_name}}</a></h4><br>

  {% endfor %}

 {% endif %}

        </div>
      </div>
  </div>

  </body>
</html>
<!DOCTYPE html>
<html>

<head>
        <title>Регистрация</title>
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
        <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.1/css/all.css" integrity="sha384-gfdkjb5BdAXd+lj+gudLWI+BXq4IuLW5IT+brZEZsLFm++aCMlF1V92rMkPaX4PP" crossorigin="anonymous">

        <style>
                body,
                html {
                        margin: 0;
                        padding: 0;
                        height: 100%;
                        background: #7abecc !important;
                }
                .user_card {
                        width: 350px;
                        margin-top: auto;
                        margin-bottom: auto;
                        background: #74cfbf;
                        position: relative;
                        display: flex;
                        justify-content: center;
                        flex-direction: column;
                        padding: 10px;
                        box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
                        -webkit-box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
                        -moz-box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
                        border-radius: 5px;

                }

                .form_container {
                        margin-top: 20px;
                }

                #form-title{
                        color: #fff;
                }
                .login_btn {
                        width: 100%;
                        background: #33ccff !important;
                        color: white !important;
                }
                .login_btn:focus {
                        box-shadow: none !important;
```

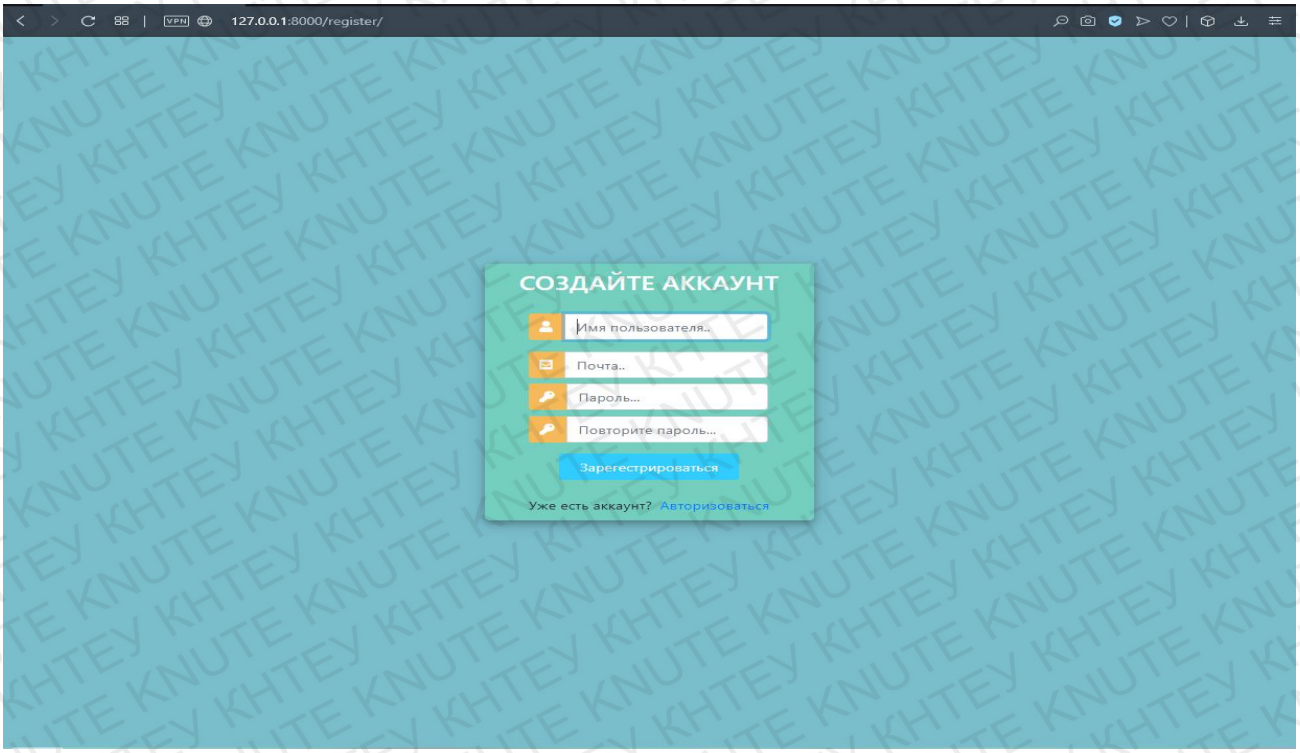
```
                outline: 0px !important;
            }
            .login_container {
                padding: 0 2rem;
            }
            .input-group-text {
                background: #f7ba5b !important;
                color: white !important;
                border: 0 !important;
                border-radius: 0.25rem 0 0 0.25rem !important;
            }
            .input_user,
            .input_pass:focus {
                box-shadow: none !important;
                outline: 0px !important;
            }

        </style>

</head>
<body>
    <div class="container h-100">
        <div class="d-flex justify-content-center h-100">
            <div class="user_card">
                <div class="d-flex justify-content-center">
                    <h3 id="form-title">СОЗДАЙТЕ АККАУНТ</h3>
                </div>
                <div class="d-flex justify-content-center form_container">

                    <form method="POST" action="">
            {% csrf_token %}
                        <div class="input-group mb-3">
                            <div class="input-group-append">
                                <span class="input-group-text"><i class="fas fa-
user"></i></span>
                            </div>
                {{form.username}}
                        </div>
                        <div class="input-group mb-2">
                            <div class="input-group-append">
                                <span class="input-group-text"><i class="fas fa-
envelope-square"></i></span>
                            </div>
                            {{form.email}}
                        </div>
                        <div class="input-group mb-2">
                            <div class="input-group-append">
                                <span class="input-group-text"><i class="fas fa-
key"></i></span>
                            </div>
                            {{form.password1}}
                        </div>
                        <div class="input-group mb-2">
                            <div class="input-group-append">
                                <span class="input-group-text"><i class="fas fa-
key"></i></span>
                            </div>
                            {{form.password2}}
                        </div>

                        <div class="d-flex justify-content-center mt-3 login_container">
                            <input class="btn login_btn" type="submit"
value="Зарегестрироваться">
                        </div>
                    </form>
```

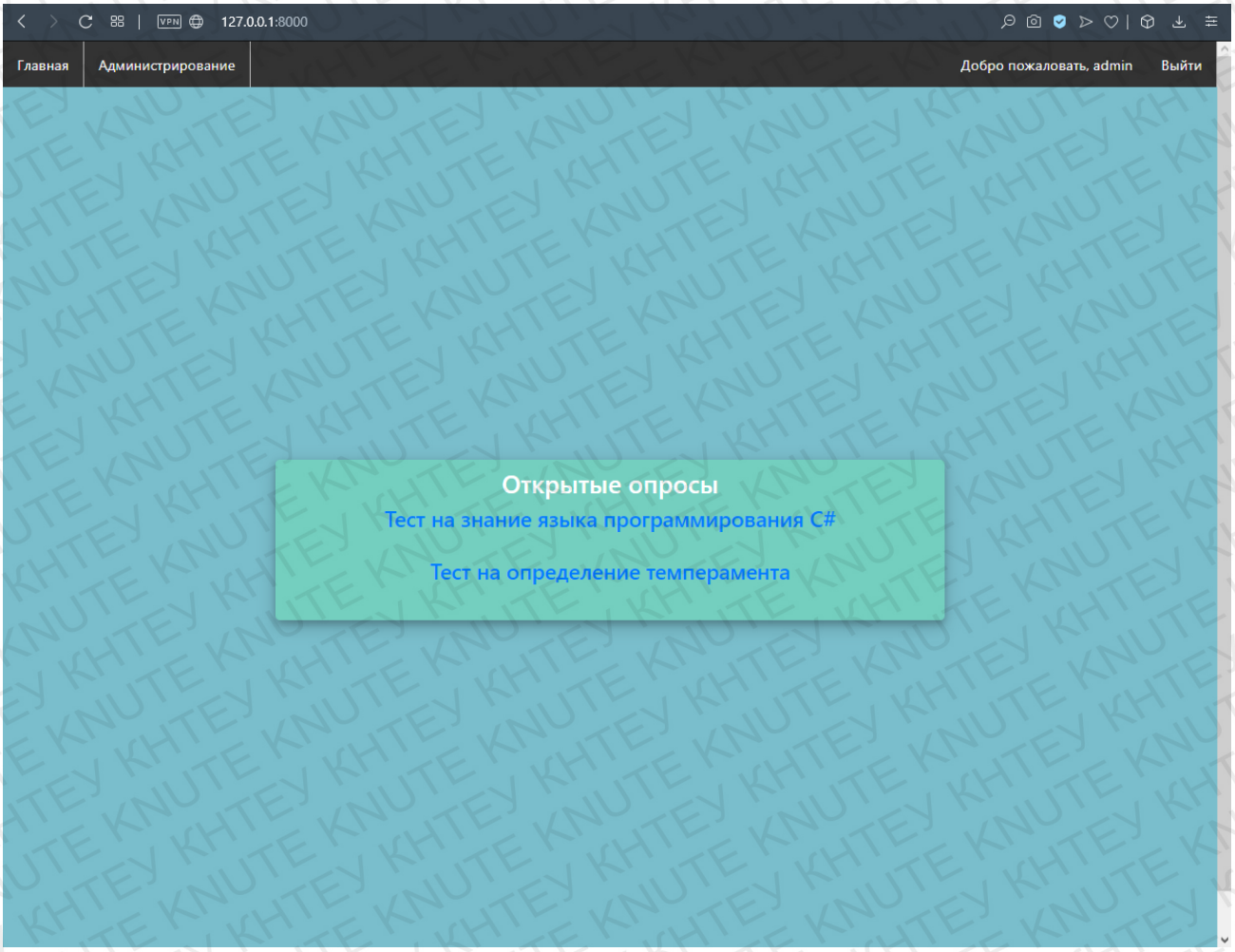
```
                                </div>
                                        {{form.errors}}
                                <div class="mt-4">
                                        <div class="d-flex justify-content-center links">
                                                Уже есть аккаунт? <a href="{% url 'login' %}" class="ml-2">Авторизоваться</a>
                                        </div>
                                </div>
                        </div>
                </div>
        </div>
        <script>

                var form_fields = document.getElementsByTagName('input')
                form_fields[1].placeholder='Имя пользователя..';
                form_fields[2].placeholder='Почта..';
                form_fields[3].placeholder='Пароль...';
                form_fields[4].placeholder='Повторите пароль...';


                for (var field in form_fields){
                        form_fields[field].className += ' form-control'
                }
        </script>
</body>
</html>


<!DOCTYPE html>
<html lang = "ru">
 <head>
        <meta charset="utf-8">

  <title>
        Продукты
  </title>


  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.1/css/all.css" integrity="sha384-gfdkjb5BdAXd+lj+gudLWI+BXq4IuLW5IT+brZEZsLFm++aCMlF1V92rMkPaX4PP" crossorigin="anonymous">

   {% load static %}
        <link rel="stylesheet" type="text/css" href="{% static 'index/css/style.css' %}">
 <style>
                body,
                html {
                        margin: 0;
                        padding: 0;
                        height: 100%;
                        background: #7abecc !important;
                }
                .user_card {
                        width: 750px;
                        margin-top: auto;
                        margin-bottom: auto;
                        background: #74cfbf;
                        position: relative;
                        display: flex;
                        justify-content: center;
                        flex-direction: column;
                        padding: 10px;
                        box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
```

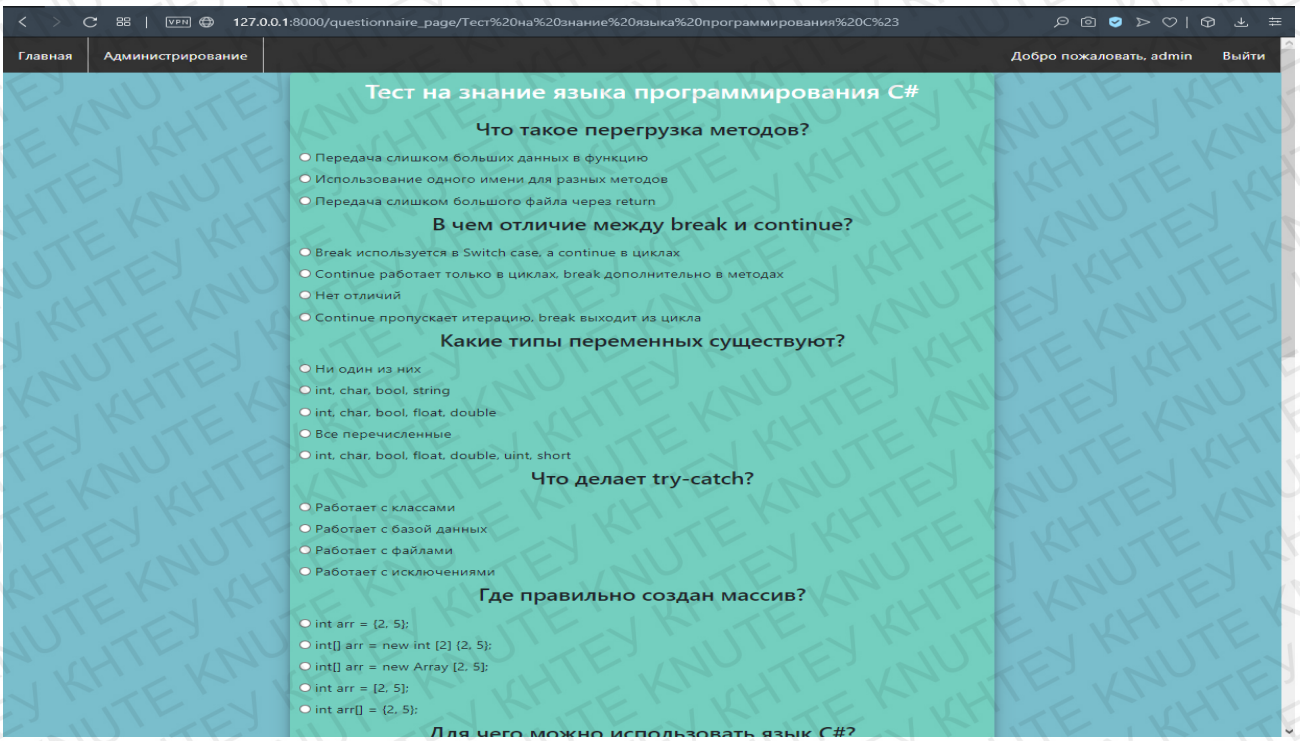
```
                -webkit-box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
                -moz-box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
                border-radius: 5px;

            {% for a in quest_ans %}
              {% for b in a %}
        <fieldset>
              {% if forloop.first %}
            <legend> <h4 align="center">{{b}}</h4> </legend>
              {% else %}
              <input type="radio" id="quest{{forloop.parentloop.counter}}.ans{{forloop.counter}}"
name="quest{{forloop.parentloop.counter}}" value="{{b}}">
            <label for="quest{{forloop.parentloop.counter}}.ans{{forloop.counter}}">{{b}}</label>
              {% endif %}
            {% endfor %}
        </fieldset>
            {% endfor %}


                                                    <div class="d-flex justify-content-center mt-3 login_container">
                                                        <input class="btn login_btn" type="submit"
value="Закончить тест">

                                                    </div>
                                            </form>


 {% endif %}

        </div>
      </div>
  </div>
  </body>
</html>
```

Додаток Б. Скріншоти графічного інтерфейсу користувача






127.0.0.1:8000
Главная
Администрирование
Добро пожаловать, admin
Выйти
Открытые опросы
Тест на знание языка программирования C#
Тест на определение темперамента


127.0.0.1:8000/questionnaire_page/Тест%20на%20знание%20языка%20программирования%20C%23
Главная
Администрирование
Добро пожаловать, admin
Выйти
Тест на знание языка программирования C#
Что такое перегрузка методов?
Передача слишком больших данных в функцию
Использование одного имени для разных методов
Передача слишком большого файла через return
В чем отличие между break и continue?
Break используется в Switch case, а continue в циклах
Continue работает только в циклах, break дополнительно в методах
Нет отличий
Continue пропускает итерацию, break выходит из цикла
Какие типы переменных существуют?
Ни один из них
int, char, bool, string
int, char, bool, float, double
Все перечисленные
int, char, bool, float, double, uint, short
Что делает try-catch?
Работает с классами
Работает с базой данных
Работает с файлами
Работает с исключениями
Где правильно создан массив?
int arr = {2, 5};
int[] arr = new int [2] {2, 5};
int[] arr = new Array [2, 5];
int arr = [2, 5];
int arr[] = {2, 5};