

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)
на тему:
«Розробка веб-сайту «Фінансовий помічник» з
калькулятором іпотеки»

Виконала студентка групи ЗІПЗ 20-1
КУРАКІНА Валерія Олексіївна

Керівник роботи:
ЄФРЕМОВ Юрій Миколайович

Рецензент:
ЧИЖМОТРЯ Олена Геннадіївна

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії
програмного забезпечення

Тетяна ВАКАЛЮК

«26» лютого 2024 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **КУРАКІНА Валерія Олексіївна**

Керівник роботи: **ЄФРЕМОВ Юрій Миколайович**

Тема роботи: **«Розробка веб-сайту «Фінансовий помічник» з калькулятором іпотеки»**, затверджена Наказом закладу вищої освіти від **«23» лютого 2024 р., №75/с**

Вихідні дані для роботи: **методи та засоби розробки веб-сайту, розробка калькулятора іпотеки.**

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Єфремов Ю. М.	27.02.2024р.	27.02.2024р.
2	Єфремов Ю. М.	14.04.2024р.	14.04.2024р.
3	Єфремов Ю. М.	04.05.2024р.	04.05.2024р.

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з програмного комплексу веб-додатку «Фінансовий помічник» з калькулятором іпотеки. Пояснювальна записка до випускної роботи містить: 66 сторінки, 21 ілюстрацію, 30 джерел, 2 додатки та 1 таблицю.

Мета випускної кваліфікаційної роботи бакалавра полягає в розробці і створенні веб-додатку, спрямованого на надання допомоги користувачам при виборі іпотечної програми для придбання нерухомості в Україні. Додаток буде забезпечувати користувачів інтерактивним калькулятором, що дозволить зручно та швидко розрахувати вартість іпотечних платежів, враховуючи умови та відсоткові ставки різних кредитних програм.

У процесі роботи були визначені основні завдання для розробки системи, проаналізовані аналоги створеного програмного продукту та обрані засоби для його реалізації. Було розроблено структуру бази даних, вибраний сучасний технологічний стек, і розроблена концепція програмного продукту. Також була описана програмна реалізація і результати тестування.

КЛЮЧОВІ СЛОВА: ВЕБ-ДОДАТОК, ОДНОСТОРІНКОВИЙ ДОДАТОК, ФОРМА ЗВ'ЯЗКУ, ГРАФІК, КАЛЬКУЛЯТОР, КРЕДИТ, ІПОТЕКА.

					ІПЗ.КР.Б - 121 - 24 - ПЗ					
Змн.	Арк.	№ докум.	Підпис	Дата	Розробка веб-сайту «Фінансовий помічник» з калькулятором іпотеки Пояснювальна записка			Літ.	Арк.	Аркушів
Розроб.		Куракіна В.О.								
Керівник		Єфремов Ю. М.							3	64
								Житомирська політехніка, група ЗІПЗ 20-1		
Рецензент		Чижмoтря О.Г.								
Зав. каф.		Вакалюк Т.А.								

ABSTRACT

Bachelor's thesis consists of a web application software suite called "Financial Assistant" with a mortgage calculator. The explanatory note to the thesis includes: 66 pages, 21 illustrations, 30 references and 1 table.

The purpose of the thesis is to develop and create a web application aimed at assisting users in choosing a mortgage program for real estate acquisition in Ukraine. The application will provide users with an interactive calculator, allowing them to conveniently and quickly calculate the cost of mortgage payments, taking into account the terms and interest rates of various credit programs.

During the work, the main tasks for system development were determined, analogs of the created software product were analyzed, and tools for its implementation were chosen. The structure of the database was developed, a modern technological stack was selected, and a concept of the software product was developed. The software implementation and test results were also described.

KEYWORDS: WEB APPLICATION, SINGLE-PAGE APPLICATION, CONTACT FORM, GRAPHIC, CALCULATOR, CREDIT, MORTGAGE.

					ІІЗ.КР.Б - 121 - 24 - ІІЗ	Арк.
						4

Зміст

РЕФЕРАТ.....	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	6
ВСТУП.....	7
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ВЕБ-ДОДАТКУ ФІНАНСОВИЙ ПОМІЧНИК З КАЛЬКУЛЯТОРОМ ІПОТЕКИ	9
1.1 Постановка задачі	9
1.2 Аналіз аналогів програмного продукту	10
1.3 Засоби розробки	18
1.4 Визначення архітектури веб-додатку	19
1.5 Визначення структури клієнтської частини веб-додатку	21
Висновки до розділу 1	23
РОЗДІЛ 2. ПРОЕКТУВАННЯ ДОДАТКУ «ФІНАНСОВИЙ ПОМІЧНИК» З КАЛЬКУЛЯТОРОМ ІПОТЕКИ	24
2.1 Визначення варіантів використання та налаштування параметрів додатку	24
2.2 Розробка бази даних	26
2.3 Проектування та реалізація алгоритмів роботи додатку	27
2.4 Реалізація функціоналу додатку	28
2.4.1 Реалізація основної архітектури додатку	28
2.4.2 Розробка калькулятора.....	33
Висновки до розділу 2	45
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ДОДАТКОМ.....	46
3.1 Порядок встановлення та налаштування параметрів додатку	46
3.2 Структура інтерфейсу веб-додатку та порядок роботи	47
3.3 Тестування роботи веб-додатку.....	56
Висновки до розділу 3	57
ВИСНОВКИ	58
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	59
ДОДАТКИ.....	63

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

CSS (Cascading Style Sheets, каскадні таблиці стилів) – мова опису зовнішнього вигляду HTML-документа. Це одна з базових технологій в сучасному інтернеті. Практично жоден сайт не обходиться без CSS, тому HTML і CSS діють в єдиній зв'язці.

HTML (від англ. HyperText Markup Language – мова гіпертекстової розмітки) – стандартизована мова розмітки веб-сторінок у Всесвітній павутині, за допомогою якої верстальники безпосередньо створюють структуру web-сторінок.

МРА – багатосторінковий додаток.

PHP (Hypertext Preprocessor) – скриптова мова загального призначення, інтенсивно застосовується для розробки веб-додатків. В даний час підтримується переважною більшістю хостинг-провайдерів і є одним з лідерів серед мов, що застосовуються для створення динамічних веб-сайтів.

SPA – односторінковий додаток.

ПЗ – Програмне забезпечення.

ВСТУП

Актуальність теми. В сучасному світі, де фінансова грамотність стає все більш важливою складовою успішного управління особистими фінансами, розробка інструментів, які допомагають ефективно управляти фінансами, стає актуальним завданням. У цьому контексті розробка веб-додатку «Фінансовий помічник» з калькулятором іпотеки набуває особливого значення.

Розробка веб-додатку «Фінансовий помічник» має велику актуальність у сучасному світі, оскільки він відповідає наростаючому попиту на інструменти фінансового планування та управління особистими фінансами.

З ростом цін на житло, іпотечні кредити стають все більш доступними та розповсюдженими серед населення. У зв'язку з цим, з'являється потреба у зручних інструментах для розрахунку вартості кредиту, щомісячних виплат та інших фінансових параметрів. Багато людей мають обмежені знання з фінансової грамотності та не володіють достатнім рівнем знань для ефективного управління своїми фінансами. Розробка інструменту, який надасть зручний доступ до фінансової інформації та порад, може значно полегшити процес фінансового планування та прийняття рішень.

У сучасному світі все більше людей використовують Інтернет та мобільні додатки для здійснення фінансових операцій та планування. Розробка веб-додатку, який надасть швидкий та зручний доступ до інструментів фінансового обліку та калькулятора іпотеки, стає необхідністю у цьому контексті. Зростаюча усвідомленість важливості фінансової грамотності спонукає багатьох людей до пошуку інструментів, які допоможуть їм розуміти складності фінансових процесів, таких як іпотечні кредити.

Мета випускної кваліфікаційної роботи бакалавра полягає в розробці веб-додатку, який надасть користувачам зручний та ефективний інструмент для обчислення параметрів іпотечного кредиту та надання фінансових порад.

Для досягнення поставленої мети необхідно виконати такі завдання:

1. Провести аналіз функціональних можливостей іпотечних калькуляторів, методів розрахунку іпотечних платежів та статистичних даних, які можуть бути збережені в базі даних.
2. Вибрати інструменти для проектування.
3. Розробити веб-сайт «Фінансовий помічник» з калькулятором іпотеки.

Об'єктом дослідження є розробка веб-додатку «Фінансовий помічник» з калькулятором іпотеки.

Предметом дослідження є використання веб-орієнтованих технологій задля створення веб-додатку «Фінансовий помічник».

Розроблений веб-додаток може служити практичним інструментом для розрахунку іпотечних кредитів, який може бути адаптований для різних програм кредитів з можливістю подальшого розширення функціоналу.

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ВЕБ-ДОДАТКУ ФІНАНСОВИЙ ПОМІЧНИК З КАЛЬКУЛЯТОРОМ ІПОТЕКИ

1.1 Постановка задачі

Зараз дуже актуальним є придбання будівлі в кредит. Зручним є наявність сервісу за допомогою, якого можна отримати всі розрахунки та безкоштовно проконсультуватись з спеціалістом.

Мета полягає в розробці і створенні веб-додатку, спрямованого на надання допомоги користувачам при виборі іпотечної програми для придбання нерухомості в Україні. Додаток буде забезпечувати користувачів інтерактивним калькулятором, що дозволить зручно та швидко розрахувати вартість іпотечних платежів, враховуючи умови та відсоткові ставки різних кредитних програм.

У кваліфікаційній роботі необхідно створити веб-сайт, корисний для людей, які хочуть оформити кредит на придбання будинку, а також хочуть отримати консультацію з цього приводу.

Для виконання цього завдання основними кроками є:

- Визначення вимог та функцій додатку;
- Проектування дизайну додатку;
- Створення дизайну для основної частини додатку;
- Проектування та розробка архітектури додатку;
- Розробка архітектури для серверної частини додатку;
- Розробка архітектури для клієнтської частини додатку;
- Розробка компонентів, сторінок та функціоналу додатку;
- Функціонал зберігання інформації
- Розробка форми зв'язку з спеціалістом.

1.2 Аналіз аналогів програмного продукту

Було проведено аналіз аналогів програмного продукту серед наявних веб- додатків в Інтернеті та визначено їх переваги та недоліки.

Одним з сайтів, які мають у своєму функціоналі фінансовий калькулятор є **Finance.ua**. Іпотечний калькулятор на Finance.ua дозволяє розрахувати щомісячні платежі, загальну суму виплат та процентні витрати. Також є можливість порівняти пропозиції різних кредиторів. Він враховує додаткові збори, податки та страхування. Інтегрований з базою даних процентних ставок.

1. Функціональність:

- Тип калькулятора: калькулятор на Finance.ua є повнофункціональним калькулятором іпотечного кредиту, який дозволяє користувачам розрахувати щомісячні платежі, загальну суму виплат, процентні витрати, а також порівняти пропозиції різних кредиторів.
- Вхідні дані: користувачі можуть вказати суму кредиту, процентну ставку, термін кредиту, тип платежу, додаткові збори, податки та страхування.
- Обчислення: калькулятор використовує стандартні формули для розрахунку іпотечних платежів, враховуючи всі relevantні фактори.
- Результати: результати відображаються чітко та зрозуміло, включаючи щомісячні платежі, загальну суму виплат, процентні витрати, а також графік амортизації.
- Інтеграція з іншими сервісами: калькулятор інтегрований з базою даних процентних ставок Finance.ua, що дозволяє користувачам порівнювати пропозиції різних кредиторів.

2. Технологічний стек:

- Програмування: калькулятор розроблений на мові JavaScript, використовуючи фреймворк React [1].

- Фронт-енд: інтерфейс користувача калькулятора чистий, сучасний та зручний у використанні. Він відповідає сучасним веб-стандартам та принципам UX/UI дизайну.
- Бази даних: калькулятор використовує базу даних Finance.ua для зберігання інформації про процентні ставки та інші дані.
- Безпека: калькулятор використовує HTTPS для захисту даних користувачів [2].
- Масштабованість: калькулятор може обробляти велику кількість користувачів та запитів, завдяки використанню хмарних сервісів.

3. Користувацький досвід:

- Інтуїтивність: калькулятор дуже інтуїтивний та простий у використанні. Користувачі можуть легко знайти та використовувати всі опції та вхідні дані.
- Швидкість: калькулятор працює швидко та без затримок.
- Доступність: калькулятор доступний для людей з інвалідністю та відповідає стандартам WCAG [3].
- Мобільність: калькулятор адаптований для мобільних пристроїв та зручний у використанні на смартфоні або планшеті.
- Локалізація: калькулятор доступний українською та російською мовами.

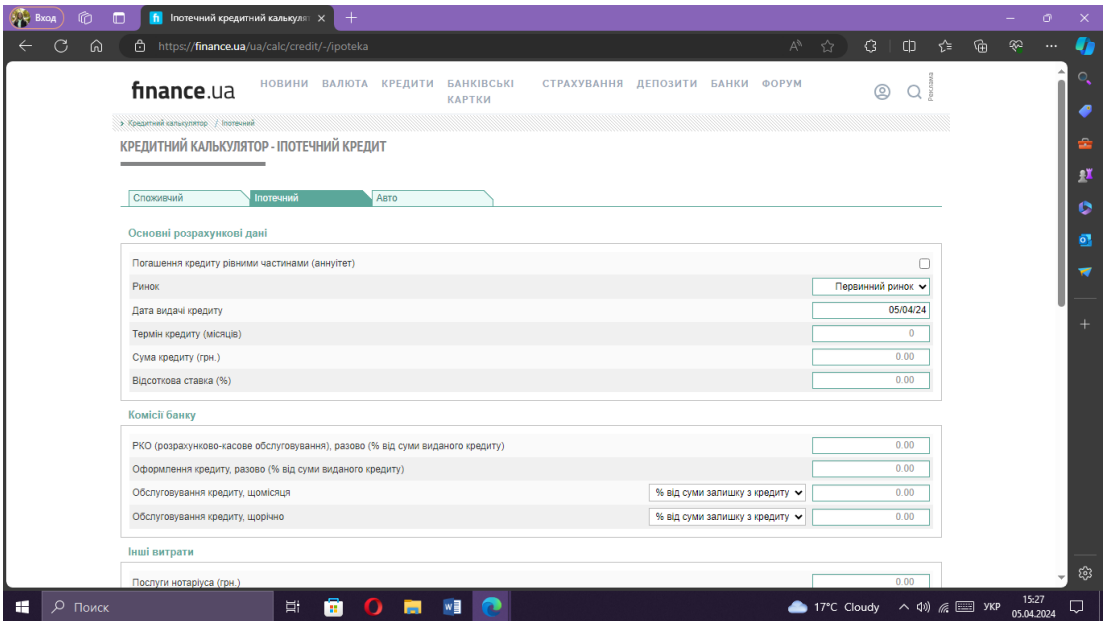


Рис. 1.1. Зовнішній вигляд сайту Finance.ua [4]

Наступним аналогом є сайт **bankchart.com.ua**. Тут також є калькулятор для розрахунку іпотеки, а також пропонуються банківські послуги.

Професійний аналіз іпотечного кредитного калькулятора онлайн в 2024 (bankchart.com.ua)

1. Функціональність:

- Тип калькулятора: калькулятор є базовим калькулятором іпотечного кредиту, який дозволяє користувачам розрахувати щомісячні платежі, загальну суму виплат та процентні витрати.
- Вхідні дані: користувачі можуть вказати суму кредиту, процентну ставку, термін кредиту та тип платежу.
- Обчислення: калькулятор використовує стандартні формули для розрахунку іпотечних платежів, враховуючи всі релевантні фактори.
- Результати: результати відображаються чітко та зрозуміло, включаючи щомісячні платежі, загальну суму виплат, процентні витрати, а також графік амортизації.
- Інтеграція з іншими сервісами: калькулятор не інтегрований з базою даних процентних ставок, тому користувачам потрібно вручну вводити процентну ставку свого кредиту.

2. Технологічний стек:

- Програмування: калькулятор розроблений на мові JavaScript.
- Фронт-енд: інтерфейс користувача калькулятора простий, але не дуже сучасний. Він відповідає основним веб-стандартам, але не використовує сучасні принципи UX/UI дизайну.
- Бази даних: калькулятор не використовує базу даних.
- Безпека: калькулятор використовує HTTPS для захисту даних користувачів.
- Масштабованість: калькулятор може обробляти невелику кількість користувачів та запитів.

3. Користувацький досвід:

- Інтуїтивність: калькулятор досить інтуїтивний, але інтерфейс користувача не дуже зручний.
- Швидкість: калькулятор працює швидко та без затримок.
- Доступність: калькулятор не доступний для людей з інвалідністю.
- Мобільність: калькулятор не адаптований для мобільних пристроїв.

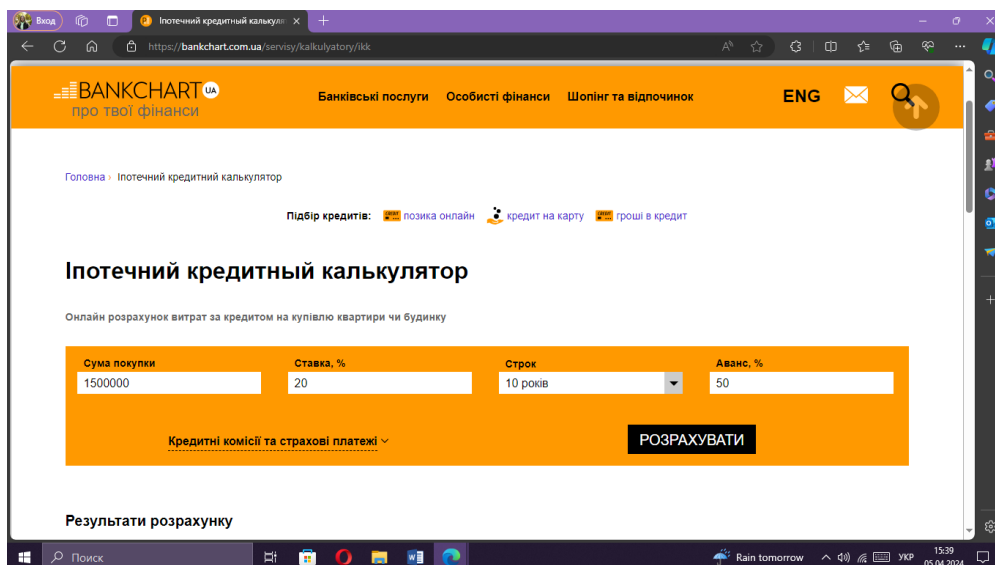


Рис. 1.2. Зовнішній вигляд веб-сайту [5]

Останнім аналогом, який буде розглянуто є Іпотечний калькулятор. Розрахунок вартості іпотеки та графіка платежів за іпотечним кредитом (fin-calc.org.ua). На ньому представлена велика кількість калькуляторів для розрахунку.

1. Функціональність:

- Тип калькулятора: сайт fin-calc.org.ua пропонує широкий спектр калькуляторів, включаючи калькулятор іпотечного кредиту, який дозволяє користувачам розрахувати щомісячні платежі, загальну суму виплат, процентні витрати, а також порівняти пропозиції різних кредиторів.
- Вхідні дані: користувачі можуть вказати суму кредиту, процентну ставку, термін кредиту, тип платежу, додаткові збори, податки та страхування.

					ІПЗ.КР.Б - 121 - 24 - ПЗ	Арк.
						13

- Обчислення: калькулятор використовує стандартні формули для розрахунку іпотечних платежів, враховуючи всі релевантні фактори.
- Результати: результати відображаються чітко та зрозуміло, включаючи щомісячні платежі, загальну суму виплат, процентні витрати, а також графік амортизації.
- Інтеграція з іншими сервісами: калькулятор інтегрований з базою даних процентних ставок fin-calc.org.ua, що дозволяє користувачам порівнювати пропозиції різних кредиторів.

2. Технологічний стек:

- Програмування: калькулятор розроблений на мові JavaScript, використовуючи фреймворк React.
- Фронт-енд: інтерфейс користувача калькулятора чистий, сучасний та зручний у використанні. Він відповідає сучасним веб-стандартам та принципам UX/UI дизайну.
- Бази даних: калькулятор використовує базу даних fin-calc.org.ua для зберігання інформації про процентні ставки та інші дані.
- Безпека: калькулятор використовує HTTPS для захисту даних користувачів.
- Масштабованість: калькулятор може обробляти велику кількість користувачів та запитів, завдяки використанню хмарних сервісів.

3. Користувацький досвід:

- Інтуїтивність: калькулятор дуже інтуїтивний та простий у використанні. Користувачі можуть легко знайти та використовувати всі опції та вхідні дані.
- Швидкість: працює швидко та без затримок.
- Доступність: доступний для людей з інвалідністю та відповідає стандартам WCAG.

– Мобільність: калькулятор адаптований для мобільних пристроїв та зручний у використанні на смартфоні або планшеті.

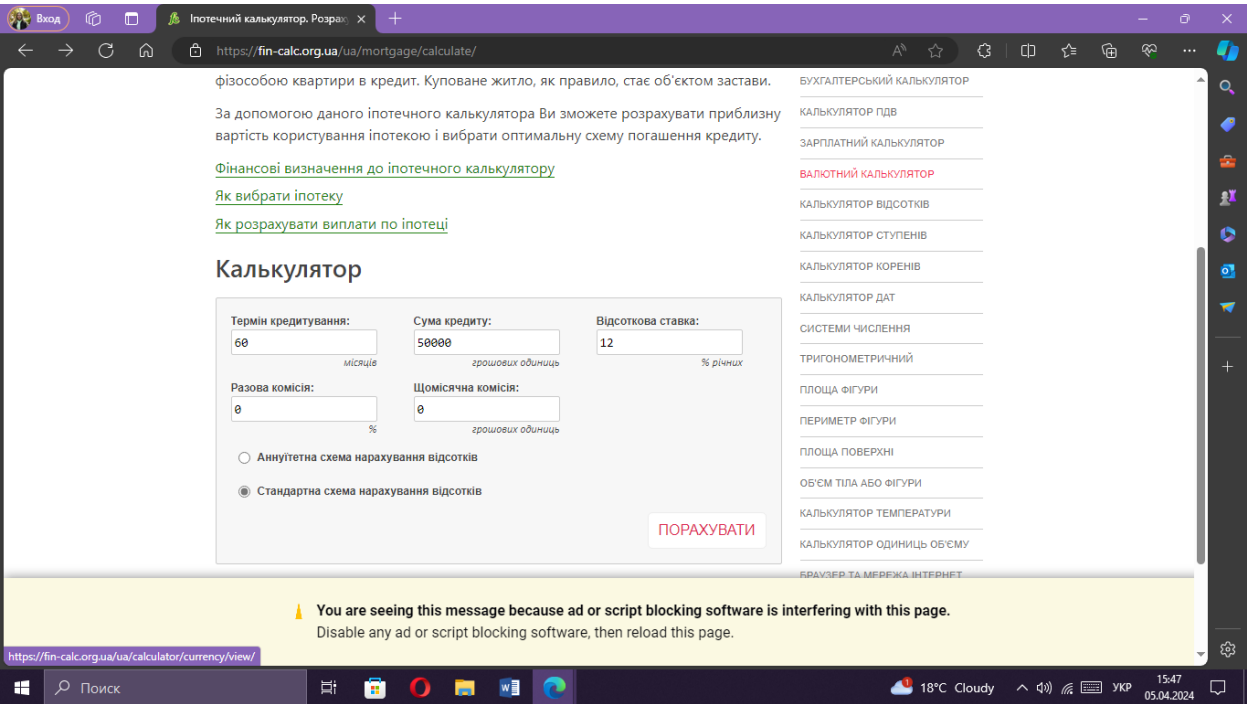


Рис. 1.3. Зовнішній вигляд веб-сайту [6]

Загалом, всі три калькулятори іпотеки, які буди представлені є корисними інструментами для людей, які шукають іпотеку в Україні.

Іпотечний кредитний калькулятор на Finance.ua є найкращим з трьох калькуляторів, завдяки його широкому спектру функцій, інтуїтивно зрозумілому інтерфейсу користувача, інтеграції з базою даних процентних ставок та загальній надійності.

bankchart.com.ua є базовим калькулятором, який може бути корисним для швидкого розрахунку іпотечних платежів. Однак він має ряд недоліків, таких як відсутність інтеграції з базою даних процентних ставок, несучасний інтерфейс.

Фінансовий калькулятор (fin-calc.org.ua) є ще одним чудовим калькулятором, який має багато функцій, простий у використанні та інтегрований з базою даних процентних ставок. Він трохи менш інтуїтивно зрозумілий, ніж калькулятор на Finance.ua, але все ж таки є цінним інструментом.

Згідно з результатами порівняння аналогів, було виявлено всі переваги та недоліки задля того, щоб гарно спроектувати додаток розрахунку кредитів.

Таблиця порівняння веб-додатків аналогів

Таблиця 1.1

Характеристика	Finance.ua	Bankchart.com.ua	Fin-calc.org.ua
Тип калькулятора	Повнофункціональний калькулятор	Базовий калькулятор	Повнофункціональний калькулятор
Вхідні дані	Сума кредиту, процентна ставка, термін кредиту, тип платежу, додаткові збори, податки, страхування.	Сума кредиту, процентна ставка, термін кредиту, тип платежу.	Сума кредиту, процентна ставка, термін кредиту, тип платежу, додаткові збори, податки, страхування.
Результати	Щомісячні платежі, загальна сума виплат, процентні витрати, графік амортизації.	Щомісячні платежі, загальна сума виплат, процентні витрати, графік амортизації.	Щомісячні платежі, загальна сума виплат, процентні витрати, графік амортизації.
Інтеграція з іншими сервісами	Інтеграція з базою даних процентних ставок Finance.ua.	Не інтегрований з базою даних процентних ставок.	Інтеграція з базою даних процентних ставок fin-calc.org.ua
Фронт-енд	Чистий, сучасний, зручний у використанні інтерфейс, відповідає сучасним веб-стандартам і	Простий, але не дуже сучасний інтерфейс, відповідає	Чистий, сучасний, зручний у використанні інтерфейс, відповідає

	принципам UX/UI дизайну.	основним веб-стандартам.	сучасним веб-стандартам і принципам UX/UI дизайну.
Бази даних	Використовує базу даних Finance.ua	Не використовує базу даних.	Використовує базу даних fin-calc.org.ua
Безпека	HTTPS	HTTPS	HTTPS
Масштабованість	Обробка великої кількості користувачів і запитів за рахунок використання хмарних сервісів.	Обробка невеликої кількості користувачів і запитів.	Обробка великої кількості користувачів і запитів за рахунок використання хмарних сервісів.
Інтуїтивність	Дуже інтуїтивний і простий у використанні.	Досить інтуїтивний, але інтерфейс не дуже зручний.	Дуже інтуїтивний і простий у використанні.
Мобільність	Адаптований для мобільних пристроїв.	Не адаптований для мобільних пристроїв.	Адаптований для мобільних пристроїв.

Беручи до уваги всі аспекти, можна ретельно проаналізувати та інтегрувати найкращі функції й можливості даних веб-сайтів у розроблюваний продукт. Такий комплексний підхід гарантує, що наш він не тільки відповідає різноманітним потребам користувачів, але й використовує сильні сторони цих відомих додатків.

1.3 Засоби розробки

У даному розділі буде детально розглянуто інструменти та технології, які були використані під час розробки веб-сайту. Сайт був розроблений у середовищі Visual Studio Code. Розмітка та стилі реалізовані за допомогою HTML5 та CSS3, анімація та зворотня робота сайту виконана за допомогою Java Script, обробка форм реалізований на PHP . Далі буде представлений огляд даних технологій.

HTML5 є останньою версією стандартної та поширеної мови розмітки гіпертексту. Це потужна версія, яка має ряд переваг та нових удосконалених функцій. Мова є дуже зручною для використання так як, завдяки цій версії має семантичні елементи, які дозволяють створювати футер, секції, статті та верхнє меню. Це допомагає у тому, щоб сайт виглядав структуровано та мав логічний вигляд [7].

CSS є мовою стилів, що використовується для оформлення та визначення зовнішнього вигляду веб-документів, написаних у мові розмітки, такій як HTML або XHTML. Використання CSS дозволяє розділити структуру веб-сторінки від її стилю, що забезпечує більшу гнучкість та легкість зміни вигляду сторінки без необхідності зміни її вмісту [8].

CSS є однією з найважливіших технологій для розробки веб-сторінки проекту, що дозволяє створювати привабливий та зручний для користувачів вигляд веб-документів.

При створенні зовнішньо привабливого вигляду веб-сайту був також використаний фреймворк Bootstrap 5. Він один з найпопулярніших та потужних фреймворків для розробки заснованих на веб-технологіях проектів.

Основна мета Bootstrap - це спростити та прискорити процес розробки веб-сайтів та веб-додатків, забезпечуючи готові компоненти і стилізовані елементи для швидкої побудови привабливого та відзивчивого інтерфейсу. Особливість Bootstrap полягає в тому, що він пропонує "мобільно-першу" філософію, що означає, що розробка зосереджується на мобільних пристроях і планшетах, а після цього адаптується до десктопів [9].

За допомогою цієї технології була використана гнучка та потужна система сітки, яка дозволяє створювати веб-дизайн, який легко адаптується до різних розмірів екранів. Сітка базується на 12-ти колонках і розділяє екран на різні частини для ефективного розміщення контенту. Був використаний широкий набір готових компонентів, таких як кнопки, форми, меню, картки, модальні вікна, пагінація та багато іншого. Вони стилізовані та гармонійно поєднуються один з одним, що дозволяє легко додавати функціональність до веб-сторінок. Bootstrap підтримує всі сучасні браузері.

Створення робочої Ajax PHP контактної форми - це важлива частина розробки веб-сайту “Фінансовий помічник”, яка дозволяє користувачам легко зв'язатися з консультантами. Ajax дозволяє надсилати дані на сервер без перезавантаження сторінки, що забезпечує більш зручний та плавний досвід для користувачів. Створення робочої Ajax PHP контактної форми включає створення HTML-структури форми, написання JavaScript-коду для збору та відправки даних, створення PHP-скрипту обробки форми та обробки відповіді з сервера [10].

Для створення динамічних інтерактивних елементів представленого веб-сайту використовується Java Script. JavaScript є слабкотипізованою мовою, що дозволяє змінювати типи даних змінних без явного вказівника типу. Використовується для взаємодії з DOM (Document Object Model) веб-сторінки, тим самим дозволяючи змінювати вміст та структуру сторінки під час її відображення у браузері. JavaScript також використовується для валідації форм, створення анімацій, взаємодії з сервером за допомогою AJAX-запитів, створення веб-додатків та багатьох інших завдань, пов'язаних з веб-розробкою [11].

1.4 Визначення архітектури веб-додатку

Завдання визначення архітектури додатку є однією з ключових частин кваліфікаційної роботи. Відповідно до сучасних тенденцій та зручності для

користувачів, рекомендується розробити веб-додаток. Такий підхід завжди залишається актуальним і забезпечує зручність взаємодії.

Оскільки дані потрібно зберігати, систематизувати та редагувати обмеженою групою користувачів, найкращим варіантом буде використання клієнт-серверної архітектури для додатку. Це забезпечить централізоване зберігання та управління даними на сервері, тоді як клієнтська частина додатку відповідатиме за взаємодію з користувачем, забезпечуючи ефективну та інтуїтивну роботу з інтерфейсом.

Зважаючи на вимоги та функціональність додатку, розглянемо різні архітектурні підходи для серверної частини:

Модульна архітектура визначається як розподіл програмного забезпечення на відокремлені модулі з чітко визначеними функціональними областями. Кожен модуль виконує специфічні завдання і може бути розроблений та протестований окремо від інших. Однак, при розгляді функціональних вимог для додатку, що орієнтований на створення іпотечного калькулятора, можна зрозуміти, що модульна архітектура для серверної частини може бути не найкращим вибором. У даному випадку важливо, щоб система працювала як цілісний блок, оскільки додаток не передбачає значних масштабних обсягів, які виправдовували б розділення серверної частини на велику кількість модулів. Замість цього, оптимальним може бути підхід, який дозволяє інтегрувати всі функціональності в єдину систему, забезпечуючи її цілісність та ефективність роботи [12].

Мікросервісна архітектура передбачає поділ програмної системи на невеликі, самостійні сервіси, які співпрацюють через легко змінювані та масштабовані компоненти. Кожен сервіс фокусується на виконанні певної функції і може бути розроблений, розгорнутий та масштабований окремо від інших. Проте, для іпотечного додатку мікросервісна архітектура може виявитися менш оптимальним вибором. Це зумовлено тим, що мікросервісна архітектура є найбільш ефективною для масштабних додатків, де різні функції можуть діяти незалежно одна від одної. У даному випадку, коли частини веб-

					ІПЗ.КР.Б - 121 - 24 - ПЗ	Арк.
						20

програми є взаємозалежними і важко роздільними, застосування мікросервісної архітектури може призвести до зайвої складності та сповільнення розробки додатку [13].

Монолітна архітектура є класичним методом розробки серверних додатків, де вся функціональність та логіка програми об'єднані в один великий блок, відомий як моноліт. У цьому підході весь код, бази даних, користувацькі інтерфейси та інші компоненти інтегровані в один проект і розгортаються як єдиний додаток. Хоча у монолітній архітектурі масштабування окремих компонентів може бути важким через їх взаємозалежність, для власного додатку, який не потребує складного масштабування, це може бути прийнятним варіантом. Такий підхід спрощує розробку та управління програмою, особливо коли вона має невеликий обсяг і не вимагає великої інфраструктури для ефективної роботи [14].

Отже, монолітна архітектура була обрана для розробки веб-додатку з калькулятором для іпотеки з урахуванням його потреб та спрощення розробки.

1.5 Визначення структури клієнтської частини веб-додатку

Після визначення серверної архітектури важливо обрати подібний підхід для клієнтської частини. Вибір технологічного стеку для клієнтської частини повинен гармонійно поєднуватися з серверною архітектурою. Також, важливо обрати сучасні технології, які добре інтегруються з серверною частиною.

Залишається визначити вид клієнтського додатку. Розглянуті два види клієнтських додатків: SPA (односторінковий додаток) та МРА (багатосторінковий додаток). У SPA весь контент завантажується один раз при першому відкритті додатку, а потім взаємодія з ним відбувається без перезавантаження сторінки. Усі необхідні дані та шаблони сторінок завантажуються одночасно, а зміни на сторінці здійснюються динамічно за допомогою JavaScript [15].

Для розробки SPA використовуються фреймворки та бібліотеки, такі як React, Next.js, Angular, Vue.js. Основною перевагою SPA є швидкість та

зручність, він зменшує навантаження на сервер, передаючи необхідний контент лише один раз, а потім змінюючи сторінку динамічно на клієнтському боці. Це сприяє зменшенню навантаження на сервер за рахунок зниження кількості запитів [16].

Хоча SPA має свої переваги, є і певні недоліки. По-перше, початкове завантаження може зайняти більше часу, особливо якщо додаток містить великий обсяг даних або залежить від зовнішніх ресурсів. Деякі старі браузерери можуть не підтримувати сучасні функції JavaScript, що використовуються в SPA. Крім того, через динамічне завантаження контенту, пошукові системи можуть мати труднощі з індексацією сторінок, що негативно впливає на SEO. Однак ці недоліки не повністю відмінюють переваги, а просто потребують уваги при розробці та використанні SPA.

MPA, або багатосторінковий додаток, функціонує за традиційним принципом, при якому кожна сторінка перезавантажується при переході на неї. В MPA кожна сторінка має унікальну URL-адресу і завантажується окремо з сервера. Це дозволяє кожній сторінці мати свої власні ресурси (HTML, CSS, JavaScript), які завантажуються автономно. Це дозволяє кожній сторінці мати власні ресурси (HTML, CSS, JavaScript), які завантажуються незалежно.

Переваги MPA полягають у його меншій складності розробки порівняно з SPA, оскільки кожна сторінка може бути розроблена окремо. Також MPA має кращу підтримку для старих браузерів і легко індексується пошуковими системами.

MPA має свої недоліки. Коли користувач переходить на нову сторінку, сервер повинен відправити повний HTML-код сторінки, що може призвести до перевантаження сервера, особливо при великому навантаженні або значній кількості користувачів. Завантаження повної сторінки може займати багато часу, що впливає на враження користувачів [17].

Порівнюючи односторінкові додатки (SPA) з традиційними багатосторінковими додатками (MPA), можна побачити, що переваги SPA

значно перевищують їхні недоліки, що робить їх сучаснішим та зручнішим вибором.

Тому вирішено, що клієнтська частина буде реалізована як односторінковий додаток (SPA). Цей підхід спрощує процес розробки та прискорює завантаження сторінок.

Висновки до розділу 1

У розділі аналізу вимог до програми та предметної області було визначено завдання та етапи їх виконання. Також було обрано стек технологій для розробки додатку. Проведено докладний порівняльний аналіз аналогічних додатків з метою виявлення їх переваг та недоліків. Цей аналіз дозволив виділити найкращі аспекти кожного додатку, які можуть бути використані при проектуванні власного додатку для калькулятора іпотеки.

Також було проведено оцінку різних архітектурних підходів для веб-додатків. В результаті вибрано клієнт-серверний підхід. Для серверної частини обрано монолітну архітектуру, а для клієнтської частини – односторінковий додаток (SPA). Цей вибір найбільш відповідає поставленим завданням і забезпечить можливість масштабування та розвитку додатку у майбутньому.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ДОДАТКУ «ФІНАНСОВИЙ ПОМІЧНИК» З КАЛЬКУЛЯТОРОМ ІПОТЕКИ

2.1 Визначення варіантів використання та налаштування параметрів додатку

Для забезпечення безперебійної роботи системи та відповідності встановленим стандартам і специфікаціям веб-орієнтованої системи веб-калькулятора, потрібно врахувати високорівневі вимоги. Ці вимоги гарантують неперервне функціонування системи та дотримання всіх необхідних стандартів і специфікацій.

Бізнес вимоги:

- Головна мета проекту полягає у спрощенні процесу розрахунку іпотечного кредиту для клієнтів.
- Представлення проекту: проект буде реалізовано у вигляді сайту, що має калькулятор іпотеки, розрахунок щомісячних платежів, графік платежів по рокам та загальний розрахунок.

Вимоги клієнтів:

- Можливість обчислювати кредит за різними ставками актуальними в Україні.
- Можливість врахування таких особливостей як: вартість нерухомості, початковий внесок, термін кредиту, податок та страховка, місячний чистий дохід.
- Можливість отримати загальну інформацію по кредитах.
- Легке та зручне внесення необхідної інформації для проведення розрахунків.
- Вимоги персоналу веб-додатку (адміністраторів):
- Можливість перегляду, створення, редагування та видалення даних, що стосуються інформації про іпотечні кредити.
- Зв'язок з користувачами для уточнення деталей їхніх розрахунків чи надання необхідної інформації.

Характеристика об'єкта комп'ютеризації:

– Користувачі зможуть отримати доступ до калькулятора іпотеки та інших інструментів для оцінки своїх фінансових можливостей та планування майбутніх витрат.

Функціональні вимоги:

– Зберігання та керування інформацією про іпотечні кредити.

Нефункціональні вимоги:

- Зрозумілий інтерфейс:
 - Для забезпечення легкості використання веб-додатку необхідно впровадити інтуїтивно зрозумілий інтерфейс з чітким та лаконічним дизайном.
 - Відповідні підказки та пояснення повинні бути доступні для кожного елементу інтерфейсу, допомагаючи користувачам зрозуміти, як користуватися кожною функцією без додаткового зусилля.
- Надійність та доступність системи:
 - Система повинна забезпечувати стабільну роботу з мінімальним часом відгуку навіть при великому навантаженні.
 - Максимальна продуктивність системи вимагає ефективного використання ресурсів сервера та оптимізації запитів до бази даних.
 - Аналіз вимог користувачів визначив можливість розширення функціональності додатку для розрахування кредиту. На рисунку 2.1. подано діаграму використання програмної системи, яка відображає різноманітні способи використання додатку користувачами.

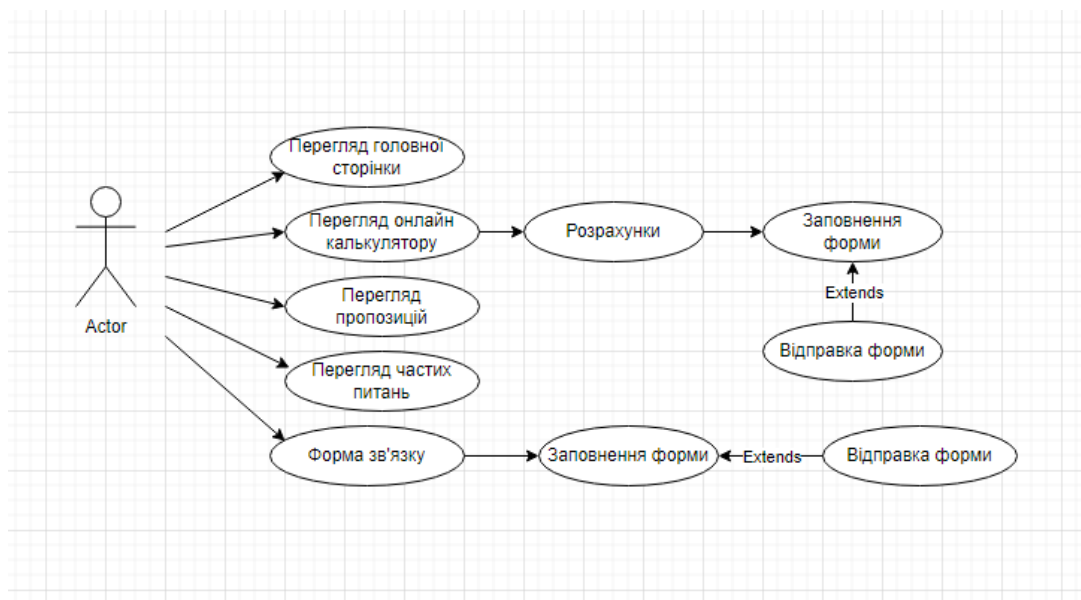


Рис. 2.1. Варіанти використання додатку

2.2 Розробка бази даних

У якості СУБД було обрано Microsoft SQL Server, адже вона відома своєю стабільністю, масштабованістю та великими можливостями [18].

Було створено базу даних з назвою creditsdb. Потім використано цю базу даних для подальшої роботи. Далі код створює таблицю requests з деякими полями та їх характеристиками:

- id INT AUTO_INCREMENT PRIMARY KEY: це унікальний числовий ідентифікатор кожного запису в таблиці. Він автоматично збільшується для кожного нового запису завдяки ключовому слову AUTO_INCREMENT.
- customer_name VARCHAR(255) NOT NULL: рядок, який містить ім'я клієнта. VARCHAR(255) вказує на те, що це текстове поле, яке може містити до 255 символів. NOT NULL означає, що це поле обов'язкове для заповнення.
- email VARCHAR(50) NOT NULL: поле для електронної пошти клієнта. Аналогічно до customer_name, воно має обмеження на довжину 50 символів і також є обов'язковим для заповнення.
- phone VARCHAR(15) NOT NULL: поле для телефонного номера клієнта. Також обмежене 15 символами і обов'язкове для заповнення.

- total_cost INT UNSIGNED NOT NULL: ціле число, яке представляє загальну вартість кредиту. INT вказує на цілочисельний тип даних, UNSIGNED - на те, що значення не може бути від'ємним, і NOT NULL означає, що поле обов'язкове для заповнення.
- first_payment INT UNSIGNED NOT NULL: ціле число, яке представляє суму першого внеску за кредит. Також це поле не може бути від'ємним і обов'язкове для заповнення.
- credit_time TINYINT UNSIGNED NOT NULL: невелике ціле число, яке визначає строк кредитування у місяцях. Обов'язкове для заповнення.
- selected_program FLOAT UNSIGNED NOT NULL: число з плаваючою комою, яке відображає обраний програмний варіант. Обов'язкове для заповнення.
- time_of_request DATETIME NOT NULL: поле типу DATETIME, яке відображає час подання запиту. Також не може бути порожнім.

Ця таблиця зберігає інформацію про запити на кредити, включаючи особисті дані клієнтів, деталі про кредит (вартість, термін, сума першого внеску), обраний програмний варіант та час подання запиту.

2.3 Проектування та реалізація алгоритмів роботи додатку

Було прийнято рішення використовувати для клієнтської частини системи звичайний алгоритм односторінкового додатку (SPA). Послідовість роботи виглядає так:

Ініціалізація: цей етап включає перевірку конструктивної цілісності веб-сайту, наявності необхідних ресурсів та стану сервера. Це може включати перевірку доступності сервера, перевірку наявності всіх необхідних компонентів веб-сайту.

Завдання/технічне обслуговування: після успішної ініціалізації веб-сайт переходить до виконання завдань або технічного обслуговування. Це включає обробку запитів від користувачів, обробку даних, оновлення вмісту

веб-сайту або проведення технічного обслуговування для підтримки оптимальної роботи веб-сайту.

Тест для коректності: після виконання завдань або технічного обслуговування проводиться тест на коректність. Це важливий етап, який забезпечує, що всі завдання виконані правильно, і веб-сайт працює належним чином. Якщо тест не пройдено, процес повертається до етапу ініціалізації.

Клієнтська частина алгоритму завершена, запит на сервер: якщо тест на коректність пройдено успішно, то клієнтська частина алгоритму вважається завершеною. Потім відправляється запит на сервер для подальшої обробки.

Запит оброблено та надіслано: після отримання відповіді від сервера, запит вважається обробленим та надісланим. Після цього веб-сайт переходить в режим очікування наступних інструкцій або дій. Це може включати очікування нових запитів від користувачів, очікування даних від сервера або очікування інших подій, які можуть вимагати втручання веб-сайту.

2.4 Реалізація функціоналу додатку

2.4.1 Реалізація основної архітектури додатку

Додаток складається з клієнтської та серверної частини. Для початку буде розглянуто те, що стосується саме клієнтської частини.

Файл *app.js* включає в себе фрагменти JavaScript, які виконують різноманітні функції на клієнтській стороні веб-додатку. Основні елементи, які стосуються архітектури проекту, можна розподілити на такі категорії:

Scroll Event Listeners:

Функція `$(window).scroll()` відстежує події прокрутки сторінки. Коли користувач прокручує сторінку на певну відстань, виконуються певні дії, такі як додавання або видалення класу `nav-sticky` для фіксованого меню (`$(".sticky").addClass("nav-sticky")`) [19].

Smooth Scrolling (Плавна прокрутка):

Функція `$('.nav-item a, .mouse-down a').on('click')` відповідає за плавну прокрутку до відповідного елемента на сторінці при натисканні на пункт меню або кнопку "Scroll Down". Вона використовує анімацію для плавного переміщення до вказаної позиції (`$('html, body').stop().animate()`) [20].

Scrollspy (Автоматичне виділення активних пунктів меню під час прокрутки):

Ініціалізація функції `$(".navbar-nav").scrollspy()` дозволяє визначити, який пункт меню повинен бути активним на панелі навігації, коли користувач прокручує сторінку [21].

Contact Form Validation (Валідація форми зв'язку):

Функція `validateForm()` перевіряє правильність заповнення полів форми зв'язку перед відправленням. Вона перевіряє, чи всі поля заповнені, і виводить повідомлення про помилку, якщо якась інформація відсутня [22].

Відображення Повідомлення про Помилку (Error Message Display):

Функція `fadeIn()` здійснює анімацію зміни прозорості для відображення повідомлень про помилки. Вона викликається при виникненні помилок валідації форми зв'язку.

Фільтрація Таблиці (Table Filtering):

Обробник події `$('body').on('change', '#filter-table-years-payment')` відповідає за фільтрацію таблиці за вказаним роком. При зміні значення вибору фільтра відображаються тільки ті рядки, які відповідають вибраному року.

Ці функції відповідають за різні аспекти веб-додатку і спільно створюють користувацький досвід, що відповідає вимогам проекту.

Файл *main.css*, що стосується клієнтської частини, можна виділити кілька аспектів :

Розмітка та стилізація контейнера:

- Визначення глобальних змінних для ширини контейнера та відступів.
- Використання класу `.container` для центрування та обмеження максимальної ширини контенту на сторінці.

- Застосування класу `.container-right` для вирівнювання контенту вправо відносно центрального контейнера.

Загальні стилі та стилізація елементів:

- Визначення загальних стилів для заголовків, тексту та інших елементів.
- Використання класу `.visually-hidden` для приховання тексту відносно користувача, але збереження його для асистентів зчитування екрану.

Форми та елементи форм:

- Встановлення стилів для текстових полів, радіокнопок та інших елементів форми.
- Використання класів `.param_details` та `.param_limits` для стилізації елементів форми та встановлення меж.

Стилі для слайдерів:

- Встановлення стилів для слайдерів, які використовуються для вибору певних значень.
- Використання стилів, щоб забезпечити відповідний зовнішній вигляд та функціональність слайдерів.
- Ці аспекти допомагають у створенні однорідного та зручного інтерфейсу для користувачів веб-сайту, а також забезпечують зручність використання та привабливий зовнішній вигляд.

controller.js відповідає за функціонал клієнтської частини веб-сайту.

Основні аспекти, що стосуються архітектури клієнтської частини, включають:

Імпорт Модулів:

- Імпорт різних модулів, таких як моделі даних, відображення результатів та додаткові утиліти.

Ініціалізація даних та оновлення виду:

- Отримання початкових даних з моделі та ініціалізація різних елементів веб-сторінки, таких як програми кредитування, вхідні дані, слайдери тощо.
- Надання функціоналу для оновлення вигляду веб-сторінки після зміни даних.

Оновлення форми та слайдерів:

- Функція `updateFormAndSliders(data)` відповідає за оновлення форми та слайдерів, залежно від дій користувача.
- Ця функція використовується для синхронізації введених користувачем даних зі значеннями на слайдерах та навпаки.

Обробка подій:

- Обробка подій, таких як зміна форми або подача заявки, використовуючи методи, які викликаються при відповідних діях користувача.

Відправка даних на сервер:

- Функція `fetchData()` відповідає за відправку даних на сервер за допомогою AJAX-запиту після натискання користувачем кнопки подачі заявки.
- Після успішної відправки заявки або виникнення помилки, виводяться відповідні повідомлення користувачеві.

Оновлення стану інтерфейсу:

- Відображення та приховування форми заявки, зміна тексту кнопок, встановлення атрибутів вхідних полів під час відправки заявки.
- Обробка Введених Даних:
- Обробка введених користувачем даних перед відправкою на сервер, перевірка коректності та валідація.
- Цей код спрощує взаємодію користувача з веб-сайтом, забезпечуючи зручний та інтуїтивно зрозумілий інтерфейс.

Також, що стосується клієнтської частини, то на сайті присутня анімація тексту. Ця частина відповідає за анімацію текстового блоку з класом `.hero-5-title`. Вона використовує бібліотеку `Anime.js` для створення затребуваної анімації. Кожна літера тексту обгортається в окремий тег `<span>`, що дозволяє керувати анімацією кожної літери окремо. Це створює ефект плавного з'явлення тексту на сторінці, залучаючи увагу користувачів до важливих повідомлень або заголовків. Присутня також анімація частинок.

Використовується бібліотека `Particles.js` для створення анімації абстрактного фону на сторінці. Ця анімація полягає у русі частинок з визначеними параметрами, такими як кількість, розмір, кольори та швидкість руху. Вона додає динамічний елемент на сторінку, що привертає увагу користувачів і створює враження живого тла.

Наприклад, файл `formatters.js`, код який в ньому записаний відноситься до серверної частини розробки. використовується для форматування цін на серверній стороні відповідно до вимог локалізації. Він використовує об'єкт `Intl.NumberFormat` для створення двох форматерів цін. Перший форматер `priceFormatter` призначений для відображення цін без десяткових частин з валютою в гривнях. Другий форматер `priceFormatterDecimals` використовується для відображення цін з десятковими частинами з валютою в гривнях. Обидва форматери налаштовані на використання мови та валюти України (ua-UA) та максимальною кількістю десяткових знаків відображення. Цей код допомагає забезпечити коректне форматування цін на серверній стороні перед їх відображенням на веб-сторінках.

Файл `contact.php` також на серверній стороні розробки. Цей PHP-скрипт відповідає за обробку даних, що надходять з форми зв'язку на веб-сайті. Після перевірки наявності вхідних даних, він виконує перевірку коректності введених даних, таких як ім'я, email і коментар. Якщо дані вірні, скрипт формує електронне повідомлення з отриманими даними і намагається надіслати його на вказану адресу електронної пошти. Крім того, він встановлює тему повідомлення та налаштовує заголовки електронної пошти, такі як від кого вона приходить та куди відповідати. У разі успішної відправки скрипт повідомляє користувача про успіх, в іншому випадку він виводить повідомлення про помилку. Цей скрипт використовується для забезпечення функціональності форми зв'язку на веб-сайті, забезпечуючи обробку даних та відправку.

`credit_request.php` встановлює з'єднання з базою даних MySQL, використовуючи отримані змінні оточення для імені користувача, пароля та

					ІПЗ.КР.Б - 121 - 24 - ПЗ	Арк.
						32

назви бази даних. Потім він отримує дані, надіслані клієнтом через HTTP POST-запит, та екранує їх, щоб уникнути SQL-ін'єкцій, використовуючи функцію `mysqli_real_escape_string`. Після цього він формує запит SQL для вставки отриманих даних у таблицю бази даних. Якщо запит виконується успішно, він повертає повідомлення "SUCCESS", інакше - повідомлення про помилку з додатковою інформацією про неуспішність виконання запиту. В кінці він закриває з'єднання з базою даних.

2.4.2 Розробка калькулятора

Калькулятор має чотири програми іпотеки на вибір, кожна з яких має свої параметри для індивідуального налаштування. Є можливість зазначати такі параметри як: вартість нерухомості, початковий внесок, термін кредиту, податок та страховка за місяць, місячний чистий дохід. Після внесення всіх необхідних даних з'являються результати розрахунків: щомісячний платіж, сума кредиту та переплата. Також представлені:

- Діаграма, яка показує співвідношення щомісячного платежу до щомісячного доходу.
- Секторна діаграма, на якій зображено обсяг щомісячного платежу, відсотків, податків та страховки. Щомісячний платіж (синій колір на діаграмі) - це найбільша частина платежу. Це основна сума, яку потрібно сплатити щомісяця. Відсотки (фіолетовий колір на діаграмі) - це частина платежу, яка йде на погашення відсотків за користування кредитом. Ця сума менша, ніж щомісячний платіж. Податок та страховка (зелений колір на діаграмі) - це найменша частина платежу. Це сума, яка йде на покриття податків та страхових внесків.
- Графік платежів за роками. Він відображає динаміку платежів за вказаним терміном кредиту. На графіку представлено три типи платежів: "Загальна заборгованість", "Основний платіж" та "Відсотковий платіж". Загальна заборгованість (сірий колір на графіку) - це загальна сума, яку потрібно сплатити. Вона найвища і поступово зменшується. Основний

платіж (зелений колір на графіку) - це частина платежу, яка йде на погашення основного боргу. Ця сума також зменшується пропорційно з загальною заборгованістю. Відсотковий платеж (червоний колір на графіку) - це частина платежу, яка йде на погашення відсотків за користування кредитом. Ця сума також зменшується пропорційно з загальною заборгованістю.

– Таблиця Розрахунків платежів.

Спочатку буде розглянута HTML-розмітка сторінки для онлайн калькулятора. Почнемо з розмітки форми: Використовується елемент `<form>` для збору введених користувачем даних. Кожен параметр кредиту представлений елементом `<label>` з додатковими елементами для назви параметра, поля для введення значення та слайдера для вибору значення в межах певного діапазону. Для вибору програми кредитування використовуються радіо кнопки, які зберігають значення різних програм ("Приват Банк", "Програма «Оселя»", "Райффайзен Банк Аваль" тощо). Кожен параметр кредиту (вартість нерухомості, початковий внесок, термін кредиту, податок та страховка за місяць, місячний чистий дохід) має відповідний слайдер. Слайдери реалізовані з допомогою JavaScript і бібліотеки для інтерактивних елементів. Є скрипт JavaScript, який відповідає за взаємодію користувача з формою. Наприклад, при зміні значення поля "Вартість нерухомості" або переміщенні слайдера, відбувається оновлення значення відповідного тексту та слайдера. Також можливо передбачити валідацію даних на стороні клієнта та інші функції, які полегшують використання калькулятора.

Сторінка оформлена за допомогою css-стилів. Використовуються правила `@import` для підключення зовнішніх стилів з файлу `reset.css`, що допомагає скинути стилі до базових значень. Також використовуються правила `@import` для підключення шрифтів з Google Fonts. Визначаються змінні `:root`, що дозволяють стандартизувати кольори, розміри та інші параметри по всьому сайту. Наприклад, використання `--primary-color` для визначення основного

кольору сайту. Використовується розмітка у вигляді ґридів з класами `.col-left` та `.col-right` для створення розділів сторінки. Стили для сітки розмітки забезпечують гнучкість і можливість розміщення різних елементів на сторінці. Застосовуються стилі для заголовків з використанням шрифту "Inter" та встановленням різних розмірів та кольорів. Використовуються правила для коректування відображення певних елементів, наприклад, клас `.visually-hidden` використовується для приховання тексту, що відображається лише для сценаріїв читача екрану. Деякі властивості, такі як `max-width` та `margin`, налаштовані на відстеження ширини екрану, забезпечуючи адаптивність сторінки до різних пристроїв та розмірів екранів. Для розміщення елементів використовуються правила Flexbox, що забезпечує просте та гнучке розташування елементів на сторінці. Представлені стилі для елементів форм, таких як радіо кнопки та поля введення, забезпечують зручний та привабливий вигляд для користувачів [23].

Функціонал веб-сайту забезпечує Java Script та jQuery. "Прикріплення" навігаційного меню до верхнього краю вікна після прокрутки на певну відстань (50px). Для цього використовується клас "nav-sticky", який додається до елементу з класом "sticky" при прокрутці. Плавну прокрутку до певної секції сторінки при натисканні на посилання в навігаційному меню або інших посиланнях з класом "mouse-down". Scrollspy - це плагін, який автоматично оновлює активне посилання в навігаційному меню в залежності від того, на якій секції сторінки знаходиться користувач [24].

Також присутня *валідація та відправка форми зворотного зв'язку*. При натисканні на кнопку "Відправити", дані з форми валідуються, і якщо всі поля заповнені, дані відправляються на сервер для подальшої обробки.

- Функція `fadeIn()` використовується для плавного зникнення повідомлення про помилку після відображення його користувачу.
- Код JavaScript, визначає об'єкти `data` та `results` для зберігання даних про програму кредитування та результатів розрахунків відповідно.

- Функція `getData()` повертає копію об'єкта `data`, а функція `getResults()` повертає копію об'єкта `results`. Функція `setData(newData)` призначена для оновлення даних у об'єкті `data` на основі нових даних `newData` та для розрахунку результатів кредитування, які зберігаються у об'єкті `results`.
- В функції `setData()` перевіряються різні варіанти оновлення даних, такі як вибір програми кредитування, введення вартості кредиту, внесення першого внеску, термін кредиту, дохід та податок. Після оновлення даних обчислюються різні параметри кредиту, такі як загальна сума кредиту, щомісячний платіж, переплата та відсоток переплати за кожний місяць. Результати обчислень зберігаються у об'єкті `results`.
- Код також містить логування для відстеження оновлень даних та результатів, що допомагає відладити та розуміти, як працює код під час виконання.
- Нарешті, за допомогою `export`, функції `getData`, `setData` та `getResults` доступні для використання в інших модулях або скриптах.

Далі буде розглянуто як працює код, який відповідає за *ініціалізацію поля введення вартості кредиту* на веб-сторінці і налаштування його взаємодії з моделлю даних.

- **Отримання початкових даних:** код викликає функцію `getData()`, щоб отримати початкові дані з моделі.
- **Ініціалізація поля введення:** спочатку, код знаходить поле введення вартості кредиту за допомогою `document.querySelector()`. Потім, використовуючи бібліотеку `Cleave.js`, воно налаштовує це поле для форматування введених значень (додавання розділювачів тисяч) [25].
- **Визначення обробників подій:** код додає обробник подій на поле введення. При введенні нового значення у поле, воно перевіряється на відповідність межах допустимих значень (`minPrice` і `maxPrice`). Якщо значення виходить за ці межі, відображається клас помилки. При зміні значення поля, воно також перевіряється на межі і оновлюється модель даних через функцію `updateModel`.

- **Оновлення моделі даних:** при кожній зміні значення поля введення, дані оновлюються в моделі за допомогою функції `updateModel`, яка отримує нове значення та тип оновлення. Це дозволяє забезпечити синхронізацію між відображенням на сторінці та даними моделі.

На завершення, функція повертає екземпляр `Cleave.js`, який може використовуватися для подальшого контролю та маніпуляцій з полем введення вартості кредиту.

Ініціалізація та управління повзунком вартості кредиту на веб-сторінці здійснюється наступним чином:

- **Отримання початкових даних:** звертається до функції `getData()`, щоб отримати початкові дані з моделі.
- **Ініціалізація повзунка:** код використовує бібліотеку `noUiSlider` для створення повзунка на сторінці. Вказуються параметри початкового значення (`start`), підключення між точками (`connect`), показ вказівників (`tooltips`), крок (`step`) та діапазон вартості кредиту (`range`). Також задається форматування значення повзунка за допомогою бібліотеки `wNumb`.
- **Обробник подій 'slide':** при кожному зміщенні повзунка, спрацьовує обробник подій 'slide'. Він отримує поточне значення повзунка, зберігає його та оновлює модель даних через функцію `updateModel`. При цьому значення перетворюється на ціле число та відфільтровується від пробілів.
- **Оновлення моделі даних:** кожне зміщення повзунка призводить до оновлення значення вартості кредиту в моделі даних за допомогою функції `updateModel`. Це забезпечує синхронізацію між відображенням на сторінці та даними моделі.
- **Повернення повзунка:** функція повертає екземпляр повзунка для подальшого використання або маніпуляцій на сторінці.

Процес здійснення ініціалізації та управління повзунком для вибору рівня доходу на веб-сторінці. Тобто, створюється повзунок на веб-сторінці, де користувач може вибрати свій рівень доходу. При зміщенні повзунка, значення доходу оновлюється в моделі даних через функцію `updateModel`,

					ІПЗ.КР.Б - 121 - 24 - ПЗ	Арк.
						37

забезпечуючи синхронізацію між відображенням на сторінці та даними моделі. Працює це наступним чином: повзунок створюється за допомогою бібліотеки `noUiSlider`. Вказуються параметри початкового значення (`start`), підключення між точками (`connect`), показ вказівників (`tooltips`), крок (`step`) та діапазон значень доходу (`range`). Також задається форматування значення повзунка за допомогою бібліотеки `wNumb`. При кожному зміщенні повзунка, спрацьовує обробник подій `'slide'`. Він отримує поточне значення повзунка, зберігає його та оновлює модель даних через функцію `updateModel`. При цьому значення перетворюється на ціле число та відфільтровується від пробілів. Кожне зміщення повзунка призводить до оновлення значення доходу в моделі даних за допомогою функції `updateModel`. Це забезпечує синхронізацію між відображенням на сторінці та даними моделі.

Управління полем введення першого внеску. Створюється поле введення на веб-сторінці, де користувач може вказати суму першого внеску при оформленні кредиту. При зміні значення у полі введення, дані оновлюються в моделі даних через функцію `updateModel`. Це дозволяє забезпечити синхронізацію між відображенням на сторінці та даними моделі.

Поле введення першого внеску ініціалізується за допомогою бібліотеки `Cleave.js`, яка дозволяє задати формат введених значень.

Управління полем введення першого внеску на веб-сторінці. Створюється поле введення на веб-сторінці, де користувач може вказати суму першого внеску при оформленні кредиту. При зміні значення у полі введення, дані оновлюються в моделі даних через функцію `updateModel`. Це дозволяє забезпечити синхронізацію між відображенням на сторінці та даними моделі.

Працює таким чином:

- **Ініціалізація поля введення:** поле введення першого внеску ініціалізується за допомогою бібліотеки `Cleave.js`, яка дозволяє задати формат введених значень.
- **Обробка подій введення:** при введенні нового значення у поле, спрацьовує обробник подій `'input'`, який перевіряє введене значення і

оновлює відображення помилки, якщо потрібно. При зміні значення поля, спрацьовує обробник подій 'change', який також перевіряє значення та виконує оновлення моделі даних.

– **Оновлення моделі даних:** змінене значення першого внеску передається функції `updateModel`, яка оновлює відповідний параметр у моделі даних.

Ініціалізація та управління кнопками вибору програми кредитування на веб-сторінці. Створюється група радіокнопок на веб-сторінці, де користувач може вибрати тип програми кредитування. Принцип роботи:

– **Ініціалізація кнопок вибору програми:** значення процентної ставки для кожної програми кредитування встановлюються відповідним атрибутом `value` для кожної радіокнопки.

– **Відображення ставок програм:** процентні ставки кожної програми відображаються на сторінці з використанням форматування відсотків.

– **Обробка подій зміни вибору:** при зміні вибору програми, спрацьовує обробник подій 'change' для кожної радіокнопки. При цьому оновлюється модель даних через функцію `updateModel`, передаючи вибрану програму та ідентифікатор радіокнопки як параметри.

Ініціалізація та управління полем введення для вказання податку. Код створює поле введення на веб-сторінці, де користувач може вказати значення податку. Алгоритм роботи:

– **Ініціалізація поля введення:** задаються налаштування для поля введення за допомогою бібліотеки `Cleave`, такі як форматування числа та використання роздільника для тисяч.

– **Обробка подій введення:** при кожному введенні користувачем значення у поле, спрацьовує обробник подій 'input'. Він перевіряє, чи введене значення знаходиться в допустимому діапазоні, та відображає відповідний клас для стилізації. Потім оновлює модель даних через функцію `updateModel`, передаючи нове значення податку та оновлення `'inputTax'`.

– **Обробка події 'change':** при зміні значення поля введення (наприклад, коли користувач вибирає значення з випадającego списку), спрацьовує обробник подій 'change'. Він перевіряє, чи введене значення знаходиться в допустимому діапазоні, і, якщо потрібно, коригує його. Потім оновлює модель даних за допомогою функції `updateModel`, забезпечуючи синхронізацію між введенням на сторінці та даними моделі.

Обробка події "slide" на слайдері та оновлення моделі даних.
Створюється обробник події "slide", який спрацьовує кожного разу, коли користувач змінює значення слайдера. При кожному зміщенні слайдера, значення слайдера перетворюється в ціле число та оновлюється в моделі даних через функцію `updateModel`, забезпечуючи синхронізацію між відображенням на сторінці та даними моделі. Працює наступним чином:

- **Обробка події "slide":** при кожному зміщенні слайдера, спрацьовує обробник події "slide". Він отримує поточне значення слайдера за допомогою методу `get()` та зберігає його в змінній `sliderValue`. Потім значення перетворюється в ціле число, прибираючи десяткову частину та роздільник пробілів за допомогою методів `split('.')`, `parseInt()`, і `replace()`.
- **Оновлення моделі даних:** за допомогою функції `updateModel`, значення події "slide" оновлюється в моделі даних. Це відбувається шляхом передачі нового значення податку та оновлення 'taxSlider' через об'єкт у другому аргументі функції `updateModel`.
- **Повернення слайдера:** після обробки події "slide", функція повертає об'єкт слайдера, щоб його можна було використовувати в інших частинах програми.

Ініціалізація та обробку подій для поля введення терміну кредиту. Як працює:

- **Ініціалізація та налаштування поля введення:** спочатку створюється новий екземпляр бібліотеки `Cleave.js` для поля введення терміну кредиту з налаштуваннями, які вказують на те, що значення поля буде числом з роздільником тисяч та з підтримкою імовірності.

- **Обробка події "input" та "change":** при кожному введенні чи зміні значення в полі введення, обробники подій перевіряють, чи введене значення знаходиться в межах мінімального та максимального терміну кредиту. В залежності від результату перевірки, вони відображають чи видаляють помилку. Потім вони оновлюють модель даних, передаючи нове значення терміну кредиту через функцію `updateModel`.
- **Повернення екземпляру Cleave.js:** після налаштування та додавання обробників подій, функція повертає створений екземпляр бібліотеки `Cleave.js` для подальшого використання.

Обробка подій для слайдера терміну кредиту. Послідовність дій:

- **Ініціалізація слайдера:** код створює новий екземпляр слайдера за допомогою бібліотеки `noUiSlider` для поля введення терміну кредиту та встановлює початкове значення, яке отримане з даних.
- **Обробка події "slide":** при зміні значення слайдера, спрацьовує обробник події "slide", який отримує поточне значення слайдера, перетворює його у ціле число та оновлює модель даних через функцію `updateModel`. Це дозволяє забезпечити синхронізацію між значенням на сторінці та в моделі.
- **Повернення екземпляру слайдера:** після ініціалізації та додавання обробника події "slide", функція повертає створений екземпляр слайдера для подальшого використання.

Код забезпечує взаємодію користувача зі слайдером та оновлення моделі даних при зміні значення терміну кредиту.

Функція `updateResultsCharts` відповідає за оновлення графіків та таблиці з результатами кредиту. Застосовується для:

- Оновлення графіків:
 - Функція спочатку визначає дані для графіків, включаючи щомісячний платіж, відсотки та податок (якщо він існує).
 - За наявності доходу, вона створює графік, який відображає співвідношення між щомісячним платежем та доходом.

– Функція також використовує Highcharts для створення кругового графіку щомісячного платежу, який відображає розподіл між самим платежем, відсотками та податками (якщо вони існують) [26].

- Оновлення таблиці:

– Функція генерує рядки таблиці з річними платежами для кожного року, включаючи номер місяця, загальну заборгованість, основний платіж, відсоткові платежі та залишок.

– Вона також створює список випадаючого меню для фільтрації таблиці за роками.

– Ця функція дозволяє коректно відображати та оновлювати графіки та таблиці з результатами кредиту в залежності від отриманих даних.

Функція *updateResultsView* відповідає за оновлення інтерфейсу з результатами кредиту на сторінці. Функція призначена для оновлення різних елементів в інтерфейсі відповідно до результатів розрахунків кредиту. Вона оновлює дані про загальну відсоткову ставку, щомісячний платіж, загальну вартість кредиту та загальну суму переоплати.

Як працює:

- Форматування даних:

– Функція спочатку форматує відсоткову ставку у зручний для відображення формат, додаючи символ відсотка та фіксуючи кількість знаків після коми.

– Для відображення чисел, таких як щомісячний платіж, загальна вартість кредиту та загальна сума переоплати, використовується форматування з допомогою функцій *priceFormatterDecimals.format* та *priceFormatter.format* з файлу *formatters.js*.

- Оновлення інтерфейсу:

– Після форматування даних, функція оновлює відповідні елементи інтерфейсу з результатами кредиту за допомогою *querySelector*, які відповідають за відображення загальної відсоткової ставки, щомісячного платежу, загальної вартості кредиту та загальної суми переоплати.

- Ця функція забезпечує коректне відображення та оновлення даних на сторінці результатів кредиту, забезпечуючи їх читабельність та зручність для користувача.

Функція *updateMinPercents* відповідає за оновлення значення мінімального відсотка першого внеску на сторінці. Ця функція призначена для відображення мінімального відсотка першого внеску у форматі відсотків на сторінці, щоб користувачі могли бачити актуальні дані. Принцип роботи:

- Оновлення значення:
 - Функція приймає об'єкт *data*, який містить дані про мінімальний відсоток першого внеску.
 - Вона використовує *querySelector* для вибору елемента з ідентифікатором *#percents-from* на сторінці.
 - Після отримання елемента, функція оновлює його вміст за допомогою властивості *innerText*, встановлюючи значення мінімального відсотка першого внеску у форматі відсотків, яке отримано з об'єкта *data*.

Ця функція гарантує, що мінімальний відсоток першого внеску на сторінці завжди відображається коректно та актуально.

Створення модулю для форматування числових значень відповідно до заданих параметрів. Основні функції модулю включають:

- **Валідація параметрів:** перевіряється правильність вхідних параметрів для форматування. Якщо будь-який з параметрів має невірне значення або тип, генерується помилка.
- **Форматування чисел до рядків:** метод *to* приймає числове значення та форматує його в рядок відповідно до заданих параметрів.
- **Розшифровка рядків в числа:** метод *from* приймає рядок та розшифровує його в числове значення відповідно до заданих параметрів.
- **Параметризація форматування:** параметри форматування включають кількість десяткових знаків, роздільник тисяч, маркер десяткової частини, префікс, суфікс та інші.

- **Можливість передачі власних функцій кодування та декодування:** модуль дозволяє користувачам передавати власні функції для кодування та декодування значень, що дозволяє додатково налаштовувати поведінку форматування.
- **Використання в різних середовищах:** модуль може використовуватися як у браузерному середовищі, так і в середовищі Node.js. Він також сумісний з системами модулів AMD.

Розробка скрипту призначеного для форматування введених користувачем номерів телефонів у визначений формат. Ось як він працює:

- **Вибір елементів вводу:** спочатку скрипт знаходить усі елементи вводу, які мають атрибут data-tel-input.
- **Отримання числового значення:** функція `getInputNumbersValue` видаляє всі нечислові символи з введеного значення.
- **Обробка події вставки тексту:** функція `onPhonePaste` відповідає за обробку події вставки тексту. Вона перевіряє, чи містить вставлений текст нечислові символи, і в разі потреби видаляє їх.
- **Обробка події введення:** функція `onPhoneInput` форматує введений номер телефону у заздалегідь визначений формат. Вона розпізнає перший символ введеного значення, який вказує на код країни або оператора, та автоматично додає відповідний префікс.
- **Очищення поля при натисканні Backspace:** функція `onPhoneKeyDown` очищає поле вводу, якщо воно містить лише один символ і користувач натискає клавішу Backspace.
- **Призначення обробників подій:** для кожного поля вводу призначаються обробники подій для взаємодії з користувачем: `keydown` для очищення поля, `input` для форматування введеного номеру телефону та `paste` для обробки вставленого тексту.

Отже, цей скрипт забезпечує зручний інтерфейс для введення номерів телефонів, автоматично форматуючи їх у заздалегідь визначений стандартний формат.

Всі описані фрагменти коду та функції є основою розробки калькулятору іпотеки.

Всі дані інструменти є важливими ключовими аспектами для коректної роботи калькулятору іпотеки.

Висновки до розділу 2

У розділі 2 «Проектування додатку "Фінансовий Помічник" з калькулятором іпотеки» було викладено вимоги до системи, включаючи бізнес та клієнтські вимоги, характеристику об'єкта комп'ютеризації, функціональні та нефункціональні вимоги, а також обрано СУБД для реалізації. Підхід до розробки веб-додатку описано шляхом визначення послідовності роботи клієнтської частини системи та взаємодії з сервером.

Проект передбачає створення зручного та легкого в управлінні веб-інтерфейсу з калькулятором іпотеки та іншими корисними інструментами для користувачів, що дозволить їм оцінити свої фінансові можливості та спланувати майбутні витрати. Вибір Microsoft SQL Server як системи управління базами даних вказує на намір забезпечити стабільність, масштабованість та високий рівень функціональності для обробки даних про іпотечні кредити.

Наприкінці розділу наведено плани реалізації функціоналу додатку, включаючи основну архітектуру та розробку калькулятору. Ці структуровані підходи до розробки сприятимуть впровадженню ефективної та надійної системи, яка відповідатиме потребам клієнтів та бізнесу.

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ДОДАТКОМ

3.1 Порядок встановлення та налаштування параметрів додатку

Система веб-додатку «Фінансовий помічник» складається з різних компонентів, кожен з яких потребує окремого налаштування та встановлення. Щоб краще зрозуміти структуру цих компонентів та процес їх розгортання, була розроблена діаграма розгортання.

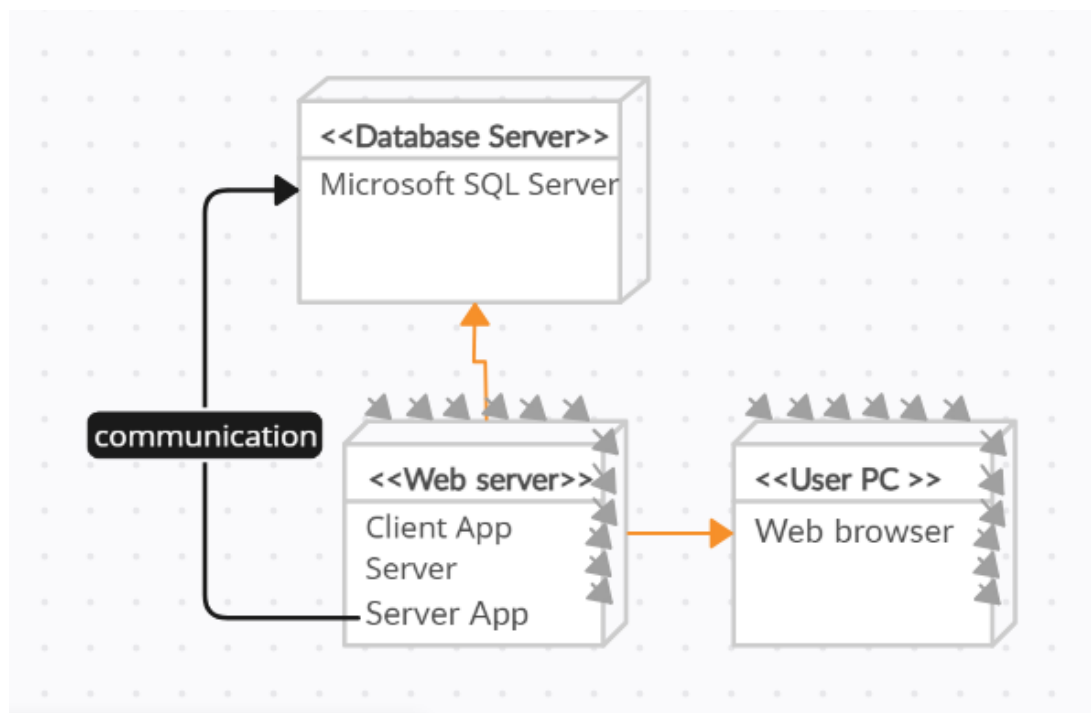


Рис. 3.1. Діаграма розгортання

Діаграма розгортання показує систему, що складається з трьох вузлів: серверу, серверу баз даних та комп'ютера або іншого пристрою користувача. З'єднання між комп'ютером користувача та веб-сервером відбувається через протокол HTTPS, тоді як протокол HTTP використовується для взаємодії між клієнтськими та серверними додатками всередині сервера.

Для налаштування веб-додатка «Фінансовий помічник» необхідно розгорнути й налаштувати всі компоненти системи. На сервері бази даних необхідно встановити сервер Microsoft SQL, створити базу даних та налаштувати параметри для забезпечення з'єднання. Взаємодія між сервером баз даних та веб-сервером відбувається через протокол TCP/IP, який ґрунтується на системі запитів/відповідей та використовує стандарт TCP для

передачі даних без втрат. Клієнтський додаток вимагає лише встановлення веб-браузера на кожному пристрої користувача.

3.2 Структура інтерфейсу веб-додатку та порядок роботи

Демонстрація роботи сайту, іпотечного калькулятора та форми заявки є важливим кроком для забезпечення оптимального функціонування веб-додатку. Вступне тестування та перевірка функціоналу допомагає впевнитися, що всі взаємодії з користувачем працюють належним чином, а результати розрахунків та збору даних є точними і достовірними.

Користувачі можуть увійти в веб-додаток як з комп'ютера, так і з мобільного телефону або інших пристроїв.

Перше, що побачить користувач, перейшовши за посиланням на додаток, це головну сторінку (рис. 3.2.).

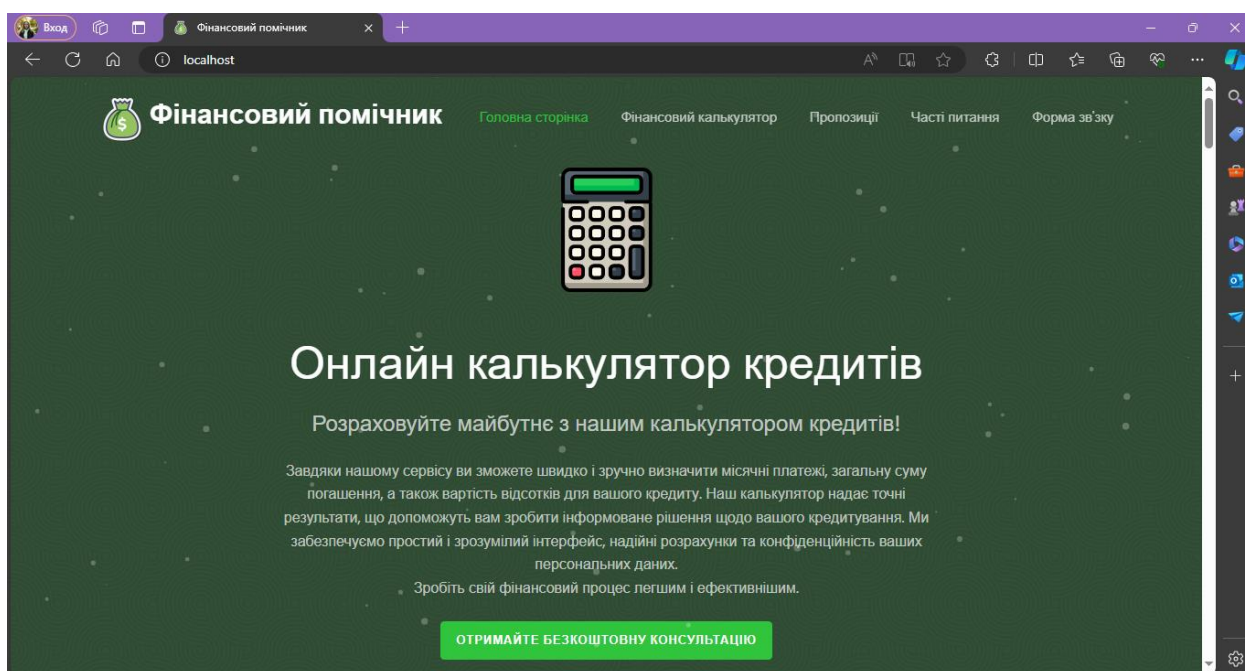


Рис. 3.2. Головна сторінка

На скріншоті представлена головна сторінка сайту, на ній зображена анімація, слова «Онлайн калькулятор кредитів» плавно з'являються та пропадають. Також присутня анімація кульок на фоні. Верхнє меню є перманентним, за його допомогою можна переміщатися на відповідний розділ сторінки, або ж просто скролити донизу.

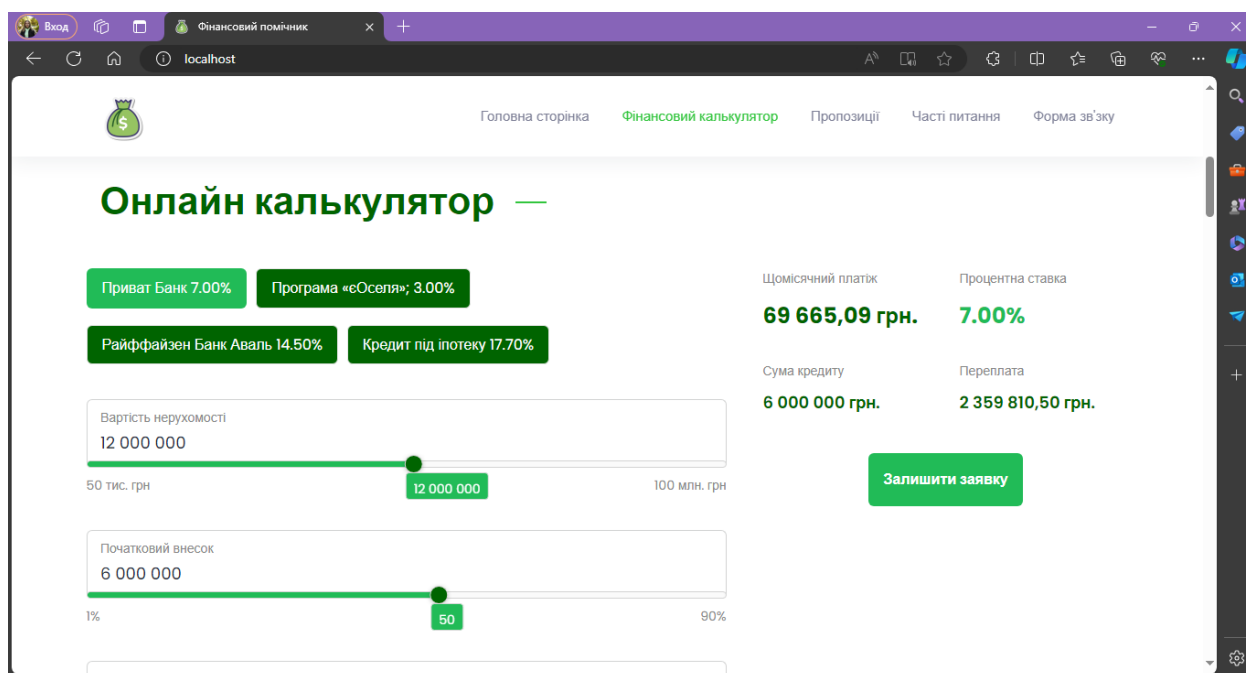


Рис. 3.3. Вигляд розділу «Фінансовий калькулятор»

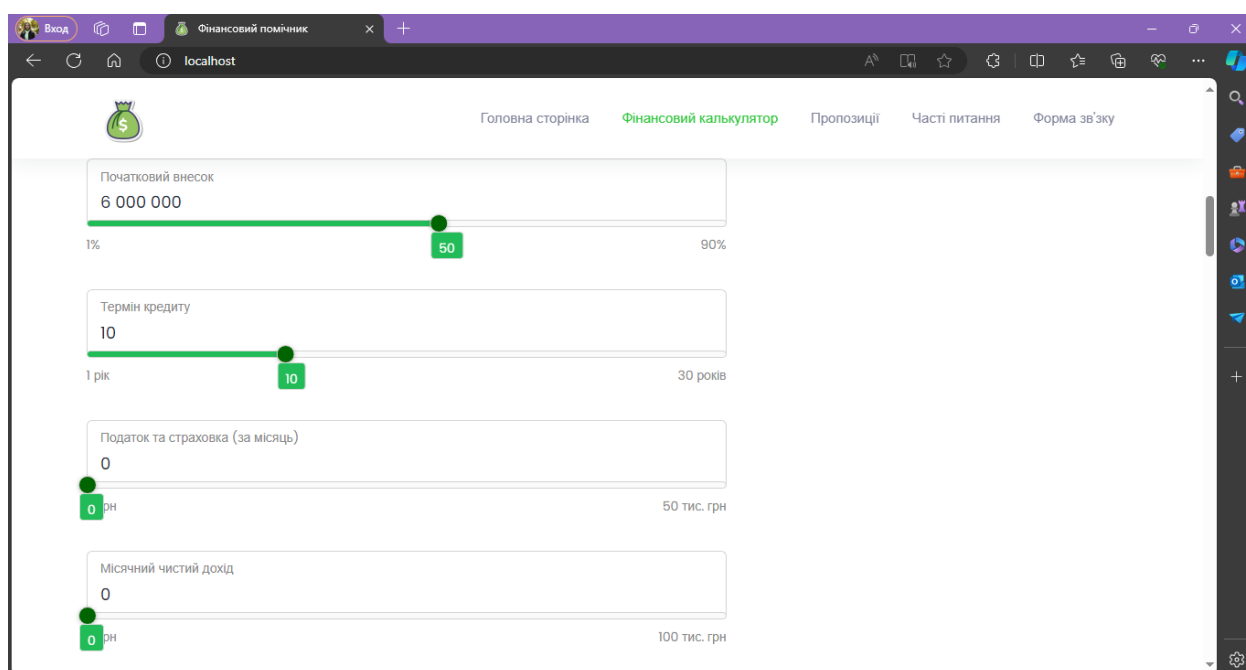


Рис. 3.4. Продовження розділу «Фінансовий калькулятор»

Залежно від обраної програми обираються налаштування і проходять обрахунки. Значення можна встановити за допомогою повзунків, або ж власноруч надрукувати. Також є форма заявки, яка допомагає користувачу вказати точні дані потрібної програми іпотеки, виходячи з цих налаштувань, які були вказані отримати консультацію [27].

Після внесення даних таких як податок та страховка, місячний чистий дохід з'являються позиції: діаграма співвідношення щомісячного платежу до

щомісячного доходу, секторна діаграма щомісячного платежу, графік платежів за роками, таблиця розрахунків платежів за роками [28].

Щомісячний платіж / Щомісячний дохід

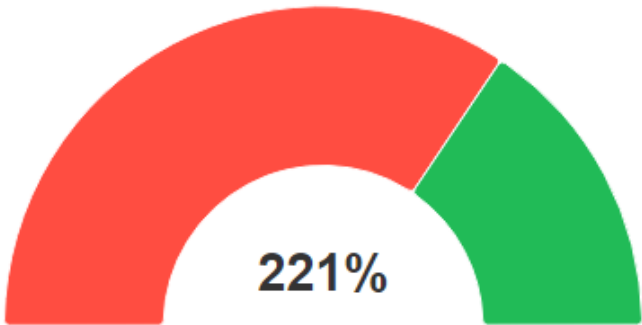


Рис. 3.5. Діаграма співвідношення щомісячного платежу до щомісячного доходу

Щомісячний платіж



Рис. 3.6. Секторна діаграма щомісячного платежу

Платежі за роками

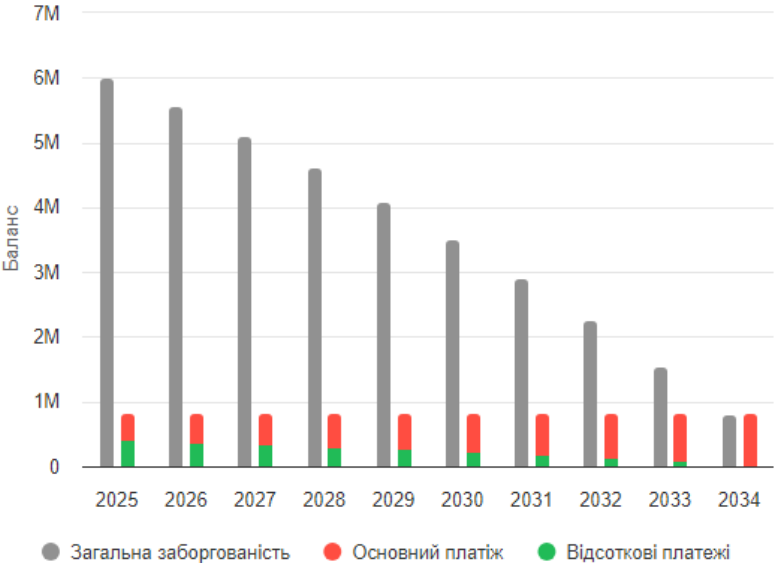


Рис. 3.7. Графік платежів за роками.

Розрахунок платежів 2025 – 2034

Місяць	Загальна заборгованість	Платіж	Основний платіж	Відсотки	Залишок боргу
1	6000000.00	69665.09	34665.09	35000.00	5965334.91
2	5965334.91	69665.09	34867.30	34797.79	5930467.61
3	5930467.61	69665.09	35070.69	34594.39	5895396.92
4	5895396.92	69665.09	35275.27	34389.82	5860121.65
5	5860121.65	69665.09	35481.04	34184.04	5824640.60
6	5824640.60	69665.09	35688.02	33977.07	5788952.58
7	5788952.58	69665.09	35896.20	33768.89	5753056.39
8	5753056.39	69665.09	36105.59	33559.50	5716950.80

Рис. 3.8. Таблиця розрахунків платежів за роками

Далі будуть розглянуті наступні розділи веб-сайту «Фінансовий помічник».

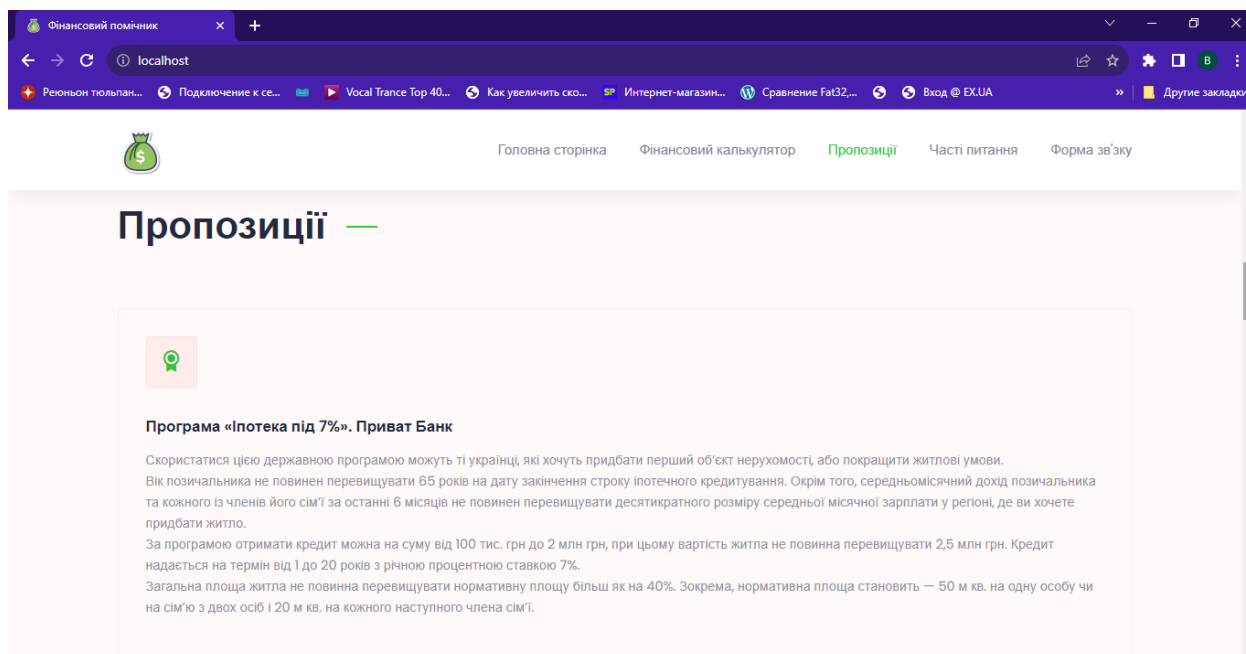


Рис. 3.9. Розділ «Пропозиції»

У цьому розділі предсталені чотири заявлені у калькуляторі програми іпотек, та інформація про них.

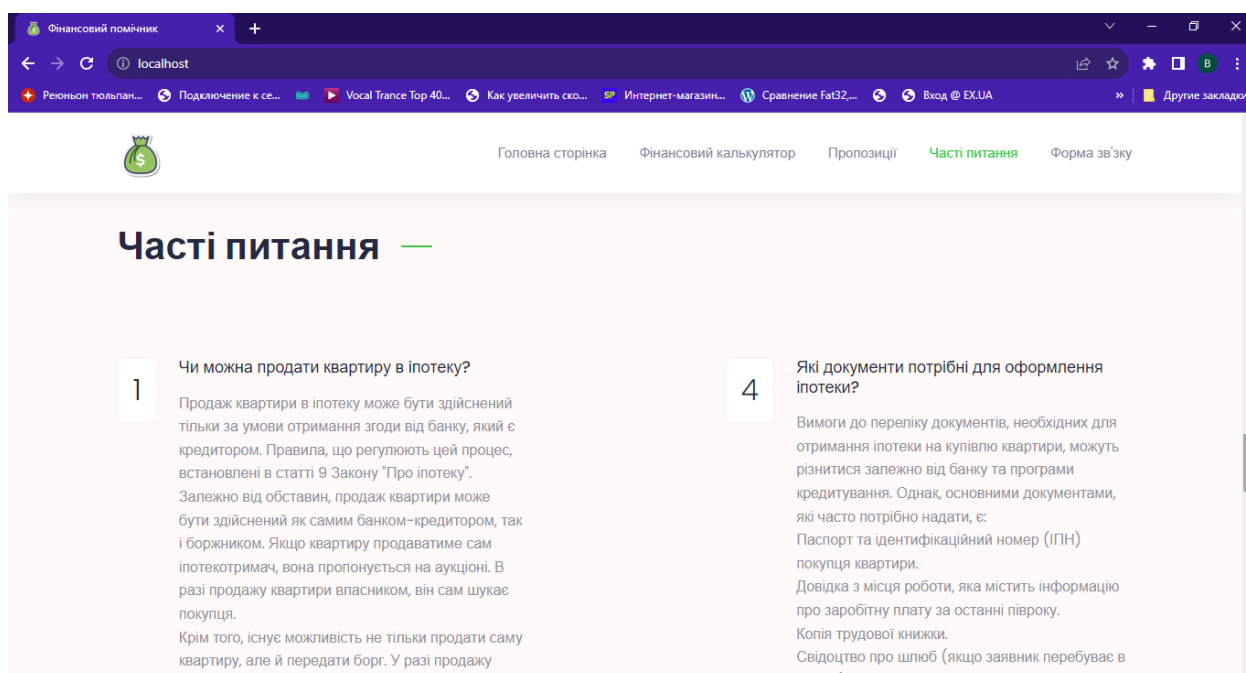


Рис. 3.10. Розділ «Часті питання»

У даному розділі містяться відповіді на самі популярні та актуальні питання серед тих, хто цікавиться оформленням іпотеки.

Рис. 3.11. Розділ «Форма зв'язку»

Ця форма зв'язку створена для того, щоб користувач міг отримати загальну консультацію з приводу іпотеки. Попередня форма має більш конкретизований напрямок програми та цифр, на відміну від цієї.

Рис. 3.12. Футер

Далі буде представлена демонстрація роботи окремих елементів веб-сайту. Спочатку буде розглянуто робота веб-калькулятора. Для цього було використано сторонній онлайн сервіс. В скріншотах нижче бачимо, що всі розрахунки правильні.

Сума покупки: 12000000, Ставка, %: 7, Строк: 10 років, Аванс, %: 15

Кредитні комісії та страхові платежі

РОЗРАХУВАТИ

Результати розрахунку

Початковий внесок	1 800 000,00
Сума кредиту	10 200 000,00
Разові комісійні витрати	0,00
Інші комісійні витрати (щомісячні та річні) за весь строк	0,00

Розрахунок щомісячного платежу і переплати

	Класика ?	Ануїтет ?
Щомісячний платіж	з 143 684,93 до 85 489,04	118 430,65
Процентні витрати за кредитом	3 599 750,00	4 011 677,86
Переплата по кредиту	3 599 750,00	4 011 677,86

Рис. 3.13. Обчислення іпотеки на сторонньому сервісі

Приват Банк 7.00%, Програма «єОселя»; 3.00%, Райффайзен Банк Аваль 14.50%, Кредит під іпотеку 17.70%

Вартість нерухомості: 12 000 000 (50 тис. грн. до 100 млн. грн.)

Початковий внесок: 1 800 000 (1% до 90%)

Термін кредиту: 10 (1 рік до 30 років)

Щомісячний платіж: 118 430,65 грн., Процентна ставка: 7.00%

Сума кредиту: 10 200 000 грн., Переплата: 4 011 677,86 грн.

Залишити заявку

Рис. 3.14. Обчислення за допомогою Фінансового помічника

Як видно результати співпадають, отже все працює коректно.

Далі буде продемонстрована робота форми зв'язку.

Фінансовий помічник

localhost

Головна сторінка **Фінансовий калькулятор** Пропозиції Часті питання Форма зв'язку

Приват Банк 7.00% Програма «єОселя»; 3.00%

Райффайзен Банк Аваль 14.50% Кредит під іпотеку 17.70%

Вартість нерухомості
12 000 000

50 тис. грн 12 000 000 100 млн. грн

Початковий внесок
1 800 000

1% 15 90%

Термін кредиту
10

1 рік 10 30 років

Щомісячний платіж **118 430,65 грн.** Процентна ставка **7.00%**

Сума кредиту **10 200 000 грн.** Переплата **4 011 677,86 грн.**

Ім'я, прізвище

Email

Телефон

Оформити заявку

Рис. 3.15. Форма зв'язку

Фінансовий помічник

localhost

Головна сторінка **Фінансовий калькулятор** Пропозиції Часті питання Форма зв'язку

Приват Банк 7.00% Програма «єОселя»; 3.00%

Райффайзен Банк Аваль 14.50% Кредит під іпотеку 17.70%

Вартість нерухомості
12 000 000

50 тис. грн 12 000 000 100 млн. грн

Початковий внесок
1 800 000

1% 15 90%

Термін кредиту
10

1 рік 10 30 років

Щомісячний платіж **118 430,65 грн.** Процентна ставка **7.00%**

Сума кредиту **10 200 000 грн.** Переплата **4 011 677,86 грн.**

😊👍 **Заявку надіслано успішно!**

Рис. 3.16. Результат відправки форми

Виходячи з цього перейдемо до розгляду роботи бази даних.

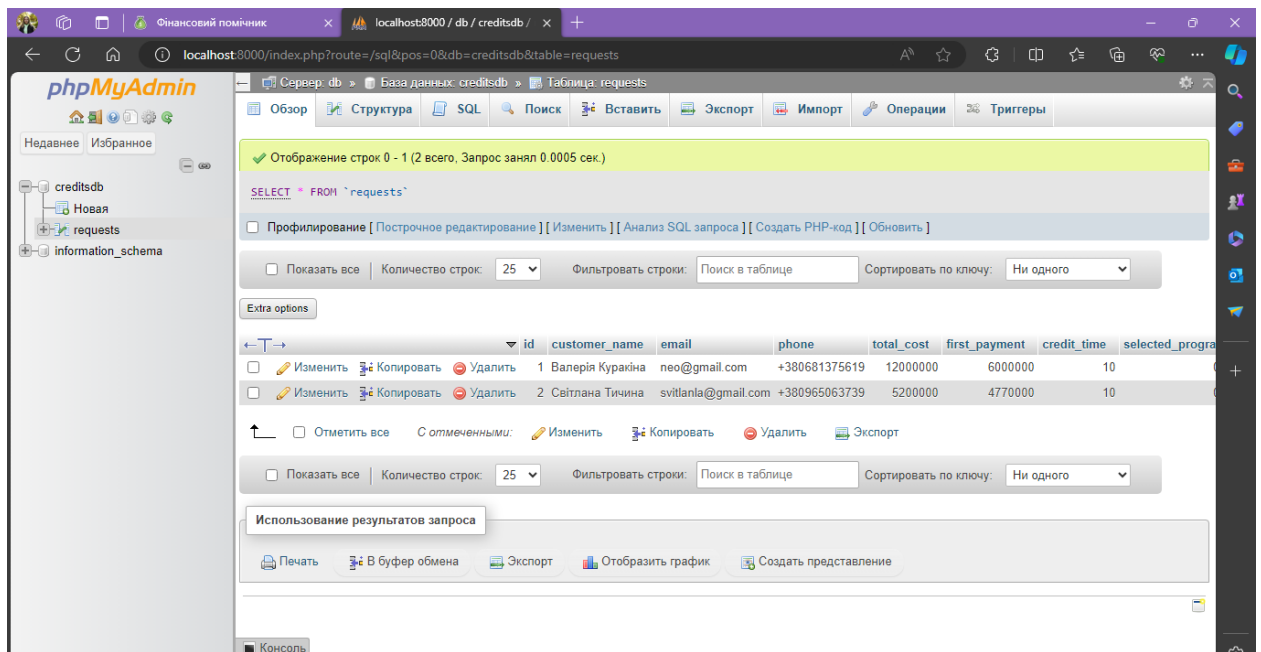


Рис. 3.17. Результат роботи бази даних

База даних коректно зберігає всі дані, які записуються у форму.

Коли відбувається вставка нових записів через виконання SQL-запиту INSERT INTO, база даних забезпечує збереження цих даних у відповідній таблиці. Відповідно до своїх налаштувань та параметрів, забезпечує захист від втрати даних, наприклад, шляхом збереження резервних копій. Під час виконання SQL-запитів, база даних виконує оптимізацію запитів для забезпечення ефективного доступу до даних.

Адміністратор бази даних може відстежувати роботу, використовуючи різні інструменти моніторингу та управління для забезпечення її надійності та продуктивності.

У цілому, коректна робота включає в себе ряд функцій та механізмів, які забезпечують надійне зберігання, ефективний доступ та безпеку даних, що є критичними аспектами для правильної роботи додатків, що використовують бази даних.

3.3 Тестування роботи веб-додатку

Тестування є важливим етапом у забезпеченні якості та стабільності програмного продукту. Його основна мета - виявлення та виправлення помилок, визначення слабких місць та потенційних проблем, а також забезпечення відповідності вимогам та очікуванням користувачів. Проведення тестування на кожному етапі розробки та після завершення проекту допомагає переконатися у високій якості продукту. Для оптимізації процесу можна використовувати прості методи тестування, особливо у випадках, коли програмний продукт не має складних проблем. Основна мета тестування - перевірка правильності функціонування окремих компонентів, модулів або всієї системи, щоб задовольнити потреби користувачів та забезпечити високу якість програмного забезпечення [29].

Передбачення помилок - це метод, за яким тестувальник аналізує можливі проблеми, які можуть виникнути під час розробки, і визначає їхні потенційні причини. Це вимагає від тестувальника інтуїції, глибоких знань і досвіду. Основна мета полягає у тому, щоб визначити умови, при яких ймовірність виникнення помилки найвища, а також перевірити це в процесі тестування [30].

Ось як це працює:

Аналіз граничних значень: цей метод базується на припущенні, що помилки найчастіше виникають на межах діапазону вхідних даних. Тому, якщо програма працює коректно для найменших та найбільших значень, є ймовірність, що вона буде працювати правильно і для інших значень всередині діапазону [31].

Інтуїція та досвід: тестувальник думає над тим, які помилки могли бути допущені в процесі розробки. Наприклад, якщо відомо, що розробник часто помиляється з сортуванням масивів, тестувальник може написати кейси для перевірки порядку розташування елементів.

Застосування на різних етапах розробки: Передбачення помилок може бути корисним на різних етапах розробки, особливо при відсутності або недостатньому обсязі специфікацій.

Проведення тестування програмного забезпечення дозволяє переконатися, що розроблений продукт відповідає необхідним функціональним можливостям та працює ефективно в різних умовах. Ці тести допомагають виявити та виправити будь-які потенційні проблеми або помилки, забезпечуючи стабільну роботу веб-додатку «Фінансовий помічник» [32].

Висновки до розділу 3

У даному розділі описано процес встановлення та налаштування веб-додатку, включаючи роботу з серверною та клієнтською частиною додатку. Проведено детальний огляд елементів інтерфейсу та їх функціоналу, а також тестування для виявлення дефектів та підвищення рівня впевненості в якості додатку. Створено тест-кейси для спрощення пошуку дефектів, їх виявлені дефекти виправлено та знову протестовано [33].

Детально представлена схема розгортання платформи, описано кожен структурний елемент системи та його взаємодію з іншими частинами платформи. Це дозволило зрозуміти архітектуру системи та взаємозв'язки між її компонентами.

Надано опис інтерфейсу розробленого програмного продукту та розглянуті всі функціональні можливості додатку. Це допомагає зорієнтуватись у функціоналі та використовувати додаток відповідно до потреб користувача.

ВИСНОВКИ

В процесі виконання випускної кваліфікаційної роботи бакалавра було реалізовано веб-додаток «Фінансовий помічник».

Зпершу було сформульовано та проаналізовано задачу. Проведено детальний аналіз предметної області та порівняльне дослідження існуючих аналогічних додатків для визначення їхніх переваг та недоліків. Початково було сформульовано та вивчено задачу. Проведено ретельний аналіз предметної області та порівняльне дослідження існуючих аналогічних додатків з метою визначення їх переваг та недоліків. Для втілення поставленої задачі було обрано монолітну архітектуру для серверної частини та модульну архітектуру для клієнтської частини, оскільки ці підходи виявилися найбільш оптимальними. Використання сучасних технологій, таких як HTML5, CSS3, Bootstrap 5, JavaScript та PHP, дозволило створити функціональний та естетично привабливий веб-додаток, який забезпечує зручний інструмент для користувачів у розрахунку іпотечних програм.

Були визначені та ретельно проаналізовані як функціональні, так і нефункціональні вимоги до додатку. Розроблено клієнтську та серверну частину веб-додатку.

Розроблений веб-додаток представляє собою значний крок у напрямку забезпечення користувачів зручним та ефективним інструментом для обчислення параметрів іпотечного кредиту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. React.Quick Start [Електронний ресурс] – Режим доступу до ресурсу: <https://react.dev/learn>.
2. What is HTTPS? [Електронний ресурс] – Режим доступу до ресурсу: <https://www.cloudflare.com/learning/ssl/what-is-https/>.
3. Web Content Accessibility Guidelines (WCAG) 2.1 [Електронний ресурс] – Режим доступу до ресурсу: <https://www.w3.org/TR/WCAG21/>.
4. Finance.ua [Електронний ресурс] – Режим доступу до ресурсу: <https://finance.ua/ua/calc/credit>.
5. Іпотечний кредитний калькулятор [Електронний ресурс] – Режим доступу до ресурсу: <https://bankchart.com.ua/servisy/kalkulyatory/ikk>.
6. Іпотечний калькулятор [Електронний ресурс] – Режим доступу до ресурсу: <https://fin-calc.org.ua/ua/mortgage/calculate/>.
7. HTML5 Tutorial [Електронний ресурс] – Режим доступу до ресурсу: <https://www.tutorialspoint.com/html5/index.htm>.
8. CCS.Accessibility [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.mozilla.org/en-US/docs/Learn/Accessibility>.
9. Bootstrap [Електронний ресурс] – Режим доступу до ресурсу: <https://getbootstrap.com/>.
10. PHP - AJAX and PHP [Електронний ресурс] – Режим доступу до ресурсу: https://www.w3schools.com/php/php_ajax_php.asp.
11. Сучасний підручник з JavaScript [Електронний ресурс] – Режим доступу до ресурсу: <https://uk.javascript.info/>.
12. Модульна архітектура ПЗ [Електронний ресурс] – Режим доступу до ресурсу: <https://javarush.com/ua/quests/lectures/ua.questservlets.level14.lecture05>

13. Мікросервісна архітектура: плюси та мінуси [Електронний ресурс] – Режим доступу до ресурсу: <https://itedu.center/ua/blog/articles/microservices-architecture-advantages-and-disadvantages/>.

14. Що краще моноліт чи мікросервіси? Як обрати архітектуру проєкту? [Електронний ресурс] – Режим доступу до ресурсу: <https://iampm.club/ua/blog/shho-krashhe-monolit-chi-mikroservisi-yak-obrati-arhitekturu-projektu/>.

15. Односторінкові і багатосторінкові веб-додатки. [Електронний ресурс] – Режим доступу до ресурсу: <https://embo.com.ua/uk/blog/odnostorinkovi-spa-i-bagatostorinkovi-pwa/>.

16. SPA у програмуванні: занурення у світ односторінкових додатків [Електронний ресурс] – Режим доступу до ресурсу: <https://foxminded.ua/spa-u-prohramuvanni/>.

17. SPA vs. MPA у Веб Розробці [Електронний ресурс] – Режим доступу до ресурсу: <https://codefinity.com/ua/courses/v2/1dcaf86a-11aa-492e-8e1d-06e055479aa9/9ea7774a-16ac-4122-9a8b-a9a9d105e7c5/9ac655c1-1b09-41bd-9173-07c90afdb5c0>.

18. SQL Server 2019 – провідні можливості для продуктивності й безпеки [Електронний ресурс] – Режим доступу до ресурсу: <https://www.microsoft.com/uk-ua/sql-server/sql-server-2019-features>.

19. How to Implement Smooth Scrolling With CSS & JavaScript [Електронний ресурс] – Режим доступу до ресурсу: <https://webdesign.tutsplus.com/smooth-scrolling-vanilla-javascript--cms-35165t>.

20. Document: scroll event [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Web/API/Document/scroll_event.

21. Scrollspy [Електронний ресурс] – Режим доступу до ресурсу: <https://getbootstrap.com/docs/4.0/components/scrollspy/>.

22. JavaScript Contact Form [Електронний ресурс] – Режим доступу до ресурсу: <https://mailtrap.io/blog/javascript-contact-form/>.
23. A Complete Guide to Flexbox [Електронний ресурс] – Режим доступу до ресурсу: <https://css-tricks.com/snippets/css/a-guide-to-flexbox/>.
24. jQuery Tutorial [Електронний ресурс] – Режим доступу до ресурсу: <https://www.w3schools.com/jquery/default.asp>.
25. Cleave.js [Електронний ресурс] – Режим доступу до ресурсу: <https://nosir.github.io/cleave.js/>.
26. Highcharts [Електронний ресурс] – Режим доступу до ресурсу: <https://www.npmjs.com/package/highcharts-export-server>.
27. «Оселя [Електронний ресурс] – Режим доступу до ресурсу: <https://privatbank.ua/kredyty/e-oselya>.
28. Доступна іпотека: державна допомога у придбанні житла [Електронний ресурс] – Режим доступу до ресурсу: <https://www.me.gov.ua/Documents/Detail?lang=uk-UA&id=20fb406b-1857-418f-85cb-76aed312c20a&title=DostupnaIpoteka-DerzhavnaDopomogaUPridbanniZhitle&isSpecial=true>.
29. Тестування програмного забезпечення: типи, види та застосування [Електронний ресурс] – Режим доступу до ресурсу: <https://foxminded.ua/testuvannia-prohramnoho-zabezpechennia/>.
30. Передбачення помилки як техніка тест-дизайну [Електронний ресурс] – Режим доступу до ресурсу: <https://training.qatestlab.com/blog/technical-articles/predicting-bug-as-a-test-design-technique/>.
31. Теорія тестування від А до Я. [Електронний ресурс] – Режим доступу до ресурсу: <https://ilarionhalushka.github.io/ua/testing-theory>.
32. Все, що потрібно знати про тестування [Електронний ресурс] – Режим доступу до ресурсу: <https://avada-media.ua/ua/services/vse-cho-nuzhno-znatpro-testirovaniye-urovni-tipy-etapy-i-metody-otladki/>.

33. Важливість тестування програмного забезпечення: типи, підготовка, інструменти [Електронний ресурс] – Режим доступу до ресурсу: https://cloud.itstep.org/blog_3/importance-of-software-testing-types-preparation-tools.

ДОДАТКИ

					ІІЗ.КР.Б - 121 - 24 - ІЗ	Арк.
						63

Лістинг App.js

```

$(window).scroll(function() {
    var scroll = $(window).scrollTop();
    if (scroll >= 50) {
        $(".sticky").addClass("nav-sticky");
    } else {
        $(".sticky").removeClass("nav-sticky");
    }
});

$('.nav-item a, .mouse-down a').on('click', function(event) {
    {
        var $anchor = $(this);
        $('html, body').stop().animate({
            scrollTop: $($anchor.attr('href')).offset().top - 0
        }, 1500, 'easeInOutExpo');
        event.preventDefault();
    }
});

$(".navbar-nav").scrollspy({
    offset: 70
});

$(window).scroll(function() {
    var scroll = $(window).scrollTop();
    if (scroll >= 50) {
        $(".nav-btn").addClass("active");
    } else {
        $(".nav-btn").removeClass("active");
    }
});

function validateForm() {
    var name = document.forms["myForm"]["name"].value;
    var email = document.forms["myForm"]["email"].value;
    var subject = document.forms["myForm"]["subject"].value;
    var comments =
document.forms["myForm"]["comments"].value;
    document.getElementById("error-msg").style.opacity = 0;
}

```

```

document.getElementById('error-msg').innerHTML = "";
if (name == "" || name == null) {
    document.getElementById('error-msg').innerHTML =
"<div class='alert alert-warning error_message'*Please
enter a Name*</div>";
    fadeIn();
    return false;
}
if (email == "" || email == null) {
    document.getElementById('error-msg').innerHTML =
"<div class='alert alert-warning error_message'*Please
enter a Email*</div>";
    fadeIn();
    return false;
}
if (subject == "" || subject == null) {
    document.getElementById('error-msg').innerHTML =
"<div class='alert alert-warning error_message'*Please
enter a Subject*</div>";
    fadeIn();
    return false;
}
if (comments == "" || comments == null) {
    document.getElementById('error-msg').innerHTML =
"<div class='alert alert-warning error_message'*Please
enter a Comments*</div>";
    fadeIn();
    return false;
}
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function () {
    if (this.readyState == 4 && this.status == 200) {
        document.getElementById("simple-msg").innerHTML
= this.responseText;
        document.forms["myForm"]["name"].value = "";
        document.forms["myForm"]["email"].value = "";
        document.forms["myForm"]["subject"].value = "";
        document.forms["myForm"]["comments"].value = "";
    }
};
xhttp.open("POST", "php/contact.php", true);
xhttp.setRequestHeader("Content-type", "application/x-
www-form-urlencoded");

```

```

        xhttp.send("name=" + name + "&email=" + email +
"&subject=" + subject + "&comments=" + comments);
        return false;
    }
    function fadeIn() {
        var fade = document.getElementById("error-msg");
        var opacity = 0;
        var intervalID = setInterval(function () {
            if (opacity < 1) {
                opacity = opacity + 0.5
                fade.style.opacity = opacity;
            } else {
                clearInterval(intervalID);
            }
        }, 200);
    }
    $('body').on('change', '#filter-table-years-payment',
function() {
        const selectedValue = $(this).val();
        $('#table-years-payment tbody
tr').removeClass('hidden');
        if(selectedValue!=='all'){

            $('#table-years-payment tbody
tr').addClass('hidden');
            $('#table-years-payment tbody tr.y-
'+selectedValue).removeClass('hidden');
        }
    });
});

```

Додаток В

Лістинг credit_request.php

```

<?php

$mysql_user = getenv("MYSQL_USER");
$mysql_pass = getenv("MYSQL_PASSWORD");
$mysql_db = getenv("MYSQL_DATABASE");

$link = mysqli_connect("db", $mysql_user, $mysql_pass,
$mysql_db, 3306);

if($link === false){

```

```

        die("ERROR: Could not connect. " .
mysqli_connect_error());
    }

    $data = json_decode(file_get_contents("php://input"), true);

    $name = mysqli_real_escape_string($link,
    $data['form']['name']);
    $email = mysqli_real_escape_string($link,
    $data['form']['email']);
    $phone = mysqli_real_escape_string($link,
    $data['form']['phone']);
    $total_cost = $data['data']['cost'];
    $first_payment = $data['data']['payment'];
    $credit_time = $data['data']['time'];
    $selected_program = $data['data']['selectedProgram'];
    $time_of_request = date("Y-m-d H:i:s");

    $sql = "INSERT INTO requests
        (customer_name, email, phone, total_cost, first_payment,
        credit_time, selected_program, time_of_request)
        VALUES ('$name', '$email', '$phone', $total_cost,
    $first_payment, $credit_time, $selected_program,
    '$time_of_request')";

    if(mysqli_query($link, $sql)){
        echo "SUCCESS";
    } else{
        echo "ERROR: Could not able to execute $sql. " .
mysqli_error($link);
    }

    mysqli_close($link);
?>

```