# ВСТУП

Правильне ведення бухгалтерського обліку - це гарантія фінансової стабільності, тому що бухгалтерія не обмежується тільки перевіркою правильності заповнення звітів та відомості витрат і доходів. Бухгалтерія - це своєрідне дослідження фінансового стану, прямий контроль за всіма пунктами і статтями грошових надходження і збитків. Це початковий етап для проведення оптимізації фінансового становища.

Домашня бухгалтерія представляє собою точний контроль за всіма надходженнями до сімейного бюджету і облік всіх статей витрат. Завдання на перший погляд просте, але практика показує, що багато людей пускають на самоплив домашню бухгалтерію. У подібному випадку рахунки і квитанції накопичуються в якійсь шухляді комоду, витрати ведуться бездумно, нічого не фіксується. Чому в такому разі дивуватися, коли в сімейному бюджеті раптово з'являється «діра». Для того, щоб уникнути подібних ситуацій необхідно вести мінімальний бухгалтерський облік, прогнозувати можливі статті витрат, планувати вигідні вклади вільних коштів. Для цього й необхідний додаток для зручнішої роботи зберігання та моніторингу грошових потоків.

В роботі розглядається питання розробки нового інструменту для роботи з домашньою бухгалтерією, що допоможе користувачам отримувати якісну та вичерпну інформацію щодо їх доходів та витрат.

# ЕТАПИ РОЗРОБКИ ВЕБ-САЙТУ

Веб-розробка — процес створення веб-сайтів або веб-додатки. Основними етапами процесу є складання технічного завдання, розробка веб-дизайн, програмування для веб на стороні клієнта і сервера, а також конфігурування веб-сервера.

**1.1 Створення технічного завдання**

Складанням технічного завдання для фахівців займається менеджер проекту. Робота з замовником починається із заповнення брифу, в якому замовник викладає свої побажання щодо візуального представлення і структури сайту, вказує на помилки в старій версії сайту, наводить приклади сайтів конкурентів. Виходячи з брифа, менеджер складає технічне завдання, враховуючи можливості програмних і дизайнерських засобів. Етап закінчується після затвердження технічного завдання замовником. Важливо відразу зазначити, що етапи проектування веб-сайтів залежать від багатьох факторів, таких як обсяг сайту, функціональність, завдання, які повинен виконувати майбутній ресурс і багато іншого. Однак, є кілька етапів, які в обов'язковому порядку присутні в плануванні будь-якого проекту

**1.2 Програмування**

Проаналізовано існуючі рішення додатків, котрі візуалізують грошові потоки. Визначені їх основні недоліки, до яких можна віднести застарілий дизайн, складність у внесенні даних користувачем, відсутність можливості синхронізувати декілька аккаунтів, неможливість автоматизованого внесення даних по надходженням та витратам з кредитної карти, відсутність інтеграції з соціальними мережами, відсутність можливості представлення інформації у вигляді графіків, неможливість зручно користуватися додатком з мобільних пристроїв.

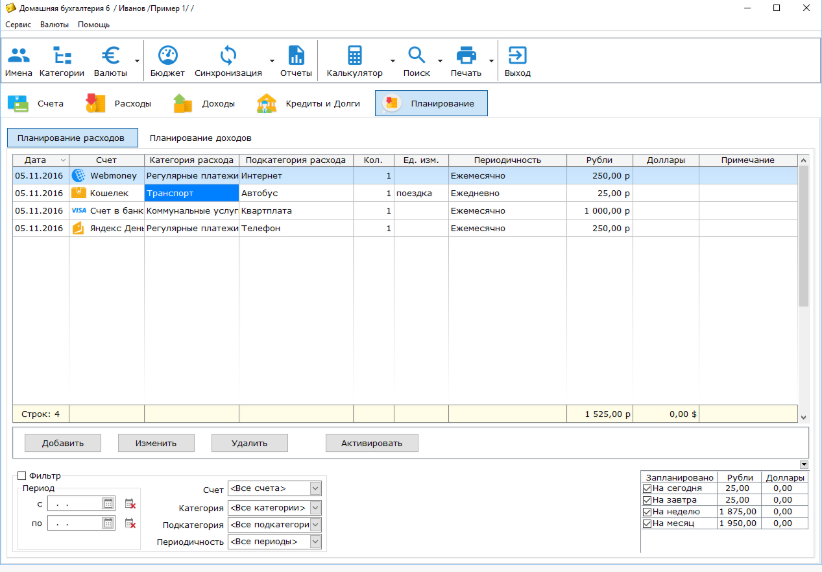
## Аналоги

«Домашня бухгалтерія» компанії KeepSoft - доступно для комп'ютера і на мобільних пристроях Android і iPhone / iPad.

Ціна від 500 грн до 800 грн.

Інтерфейс додатку наведено на рисунку 1.1

Рисунок 1.1 - Інтерфейс додатку «Домашня бухгалтерія» компанії KeepSoft

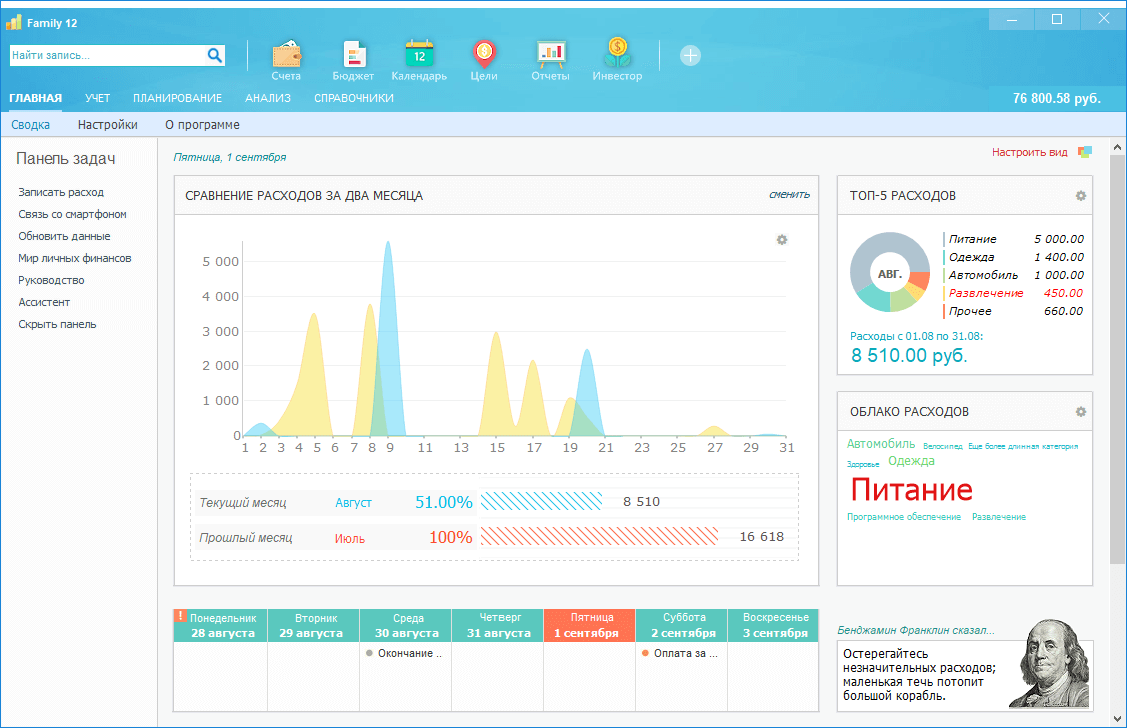


**«family12»** - Домашняя бухгалтерия для Windows.

Має лише версію для однієї операційної системи. Не має мобільної версії.  
Працює з витратами, генерує графіки.

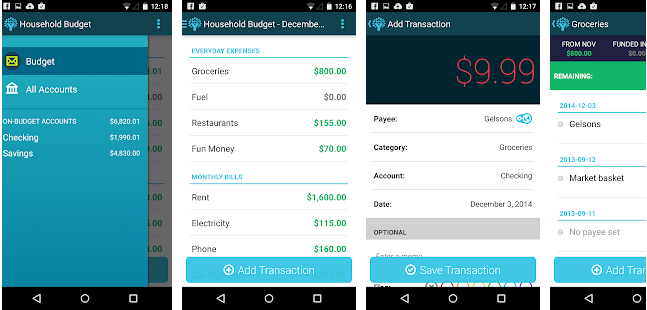
Цена от 125грн до 675грн

Інтерфейс додатку наведено на рисунку 1.2

Рисунок 1.2 - Інтерфейс додатку **«family12»** 

«YNAB» - Програма працює тільки з однією валютою і не русифікована. YNAB. Додаток для платформи Android та IOS враховує доходи і витрати. Це повноцінна система, яка дозволяє планувати бюджет на короткострокову та довгострокову перспективу. Вона визначає недорасхода і перевитрата і коректно працює з боргами

Інтерфейс додатку наведено на рисунку 1.3

Рисунок 1.3 - Інтерфейс додатку «YNAB»

Інструменти, котрі використані для розробки додатку

Визначені основні вимоги до веб-додатку ведення домашньої бухгалтерії та спроектована його узагальнена архітектура. В якості архітектури сервісу обрана трирівнева архітектура «клієнт-сервер застосунків – сервер бази даних».

HTML  — стандартна [мова розмітки](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D1%80%D0%BE%D0%B7%D0%BC%D1%96%D1%82%D0%BA%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) [веб-сторінок](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0) в [Інтернеті](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82). Більшість [веб-сторінок](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0) створюються за допомогою мови HTML (або [XHTML](https://uk.wikipedia.org/wiki/XHTML)). Документ HTML оброблюється [браузером](https://uk.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80) та відтворюється на екрані у звичному для людини вигляді. HTML є похідною мовою від [SGML](https://uk.wikipedia.org/wiki/SGML), успадкувавши від неї [визначення типу документа](https://uk.wikipedia.org/wiki/Document_Type_Definition) та ідеологію структурної розмітки тексту. Попри те, що HTML — штучна комп'ютерна мова, вона не є [мовою програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F). HTML разом із [каскадними таблицями стилів](https://uk.wikipedia.org/wiki/%D0%9A%D0%B0%D1%81%D0%BA%D0%B0%D0%B4%D0%BD%D1%96_%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D1%96_%D1%81%D1%82%D0%B8%D0%BB%D1%96%D0%B2) та [вбудованими скриптами](https://uk.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B2%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) — це три основні технології побудови [веб-сторінок](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0).

CSS — спеціальна [мова](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), що використовується для опису зовнішнього вигляду сторінок, написаних [мовами розмітки даних](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D1%80%D0%BE%D0%B7%D0%BC%D1%96%D1%82%D0%BA%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85).

Найчастіше CSS використовують для візуальної презентації сторінок, написаних [HTML](https://uk.wikipedia.org/wiki/HTML) та [XHTML](https://uk.wikipedia.org/wiki/XHTML), але формат CSS може застосовуватися до інших видів [XML](https://uk.wikipedia.org/wiki/XML)-документів.

Специфікації CSS були створені та розвиваються [Консорціумом Всесвітньої мережі](https://uk.wikipedia.org/wiki/W3C).

CSS має різні *рівні* та *профілі*. Наступний рівень CSS створюється на основі попередніх, додаючи нову функціональність або розширюючи вже наявні функції. Рівні позначаються як CSS1, CSS2 та CSS3. Профілі — сукупність правил CSS одного або більше рівнів, створені для окремих типів пристроїв або [інтерфейсів](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81). Наприклад, існують профілі CSS для [принтерів](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BD%D1%82%D0%B5%D1%80), мобільних пристроїв тощо.

PHP — [скриптова](https://uk.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B2%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [мова програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), була створена для генерації [HTML](https://uk.wikipedia.org/wiki/HTML)-сторінок на стороні [веб-сервера](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80). PHP є однією з найпоширеніших мов, що використовуються у сфері веб-розробок (разом із [Java](https://uk.wikipedia.org/wiki/Java), [.NET](https://uk.wikipedia.org/wiki/.NET), [Perl](https://uk.wikipedia.org/wiki/Perl), [Python](https://uk.wikipedia.org/wiki/Python), [Ruby](https://uk.wikipedia.org/wiki/Ruby)). PHP підтримується переважною більшістю [хостинг](https://uk.wikipedia.org/wiki/%D0%A5%D0%BE%D1%81%D1%82%D0%B8%D0%BD%D0%B3)-[провайдерів](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B2%D0%B0%D0%B9%D0%B4%D0%B5%D1%80). PHP — проект [відкритого програмного забезпечення](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%BA%D1%80%D0%B8%D1%82%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F).

PHP інтерпретується веб-сервером у HTML-код, який передається на сторону клієнта. На відміну від скриптової мови [JavaScript](https://uk.wikipedia.org/wiki/JavaScript), користувач не бачить PHP-коду, бо браузер отримує готовий html-код. Це є перевага з точки зору безпеки, але погіршує інтерактивність сторінок. Але ніхто не забороняє використовувати PHP для генерування [JavaScript](https://uk.wikipedia.org/wiki/JavaScript)-кодів, які виконуються вже на стороні клієнта.

JavaScript (**JS**) — динамічна, [об'єктно-орієнтована](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [прототипна](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D1%82%D0%B8%D0%BF%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [мова програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F). Реалізація стандарту [ECMAScript](https://uk.wikipedia.org/wiki/ECMAScript). Найчастіше використовується для створення сценаріїв [веб-сторінок](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0), що надає можливість на стороні [клієнта](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80%D0%BD%D0%B0_%D0%B0%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0) (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, [асинхронно](https://uk.wikipedia.org/wiki/AJAX) обмінюватися даними з [сервером](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80%D0%BD%D0%B0_%D0%B0%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0), змінювати [структуру](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%B0_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0) та [зовнішній вигляд](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B4%D0%B8%D0%B7%D0%B0%D0%B9%D0%BD) [веб-сторінки](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0).

JavaScript класифікують як [прототипну](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D1%82%D0%B8%D0%BF%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) (підмножина [об'єктно-орієнтованої](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)), [скриптову мову](https://uk.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B2%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) програмування з динамічною типізацією. Окрім прототипної, JavaScript також частково підтримує інші парадигми програмування ([імперативну](https://uk.wikipedia.org/wiki/%D0%86%D0%BC%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) та частково [функціональну](https://uk.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D1%96%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) і деякі відповідні архітектурні властивості, зокрема: [динамічна](https://uk.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D1%96%D1%87%D0%BD%D0%B0_%D1%82%D0%B8%D0%BF%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F) та [слабка](https://uk.wikipedia.org/w/index.php?title=%D0%A1%D0%BB%D0%B0%D0%B1%D0%BA%D0%B0_%D1%82%D0%B8%D0%BF%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F&action=edit&redlink=1) [типізація](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%82%D0%B8%D0%BF%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%97), автоматичне керування пам'яттю, прототипне [наслідування](https://uk.wikipedia.org/wiki/%D0%9D%D0%B0%D1%81%D0%BB%D1%96%D0%B4%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)), [функції](https://uk.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D1%96%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) як [об'єкти першого класу](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82_%D0%BF%D0%B5%D1%80%D1%88%D0%BE%D0%B3%D0%BE_%D0%BA%D0%BB%D0%B0%D1%81%D1%83).

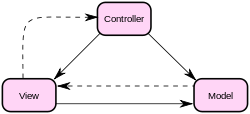
У комп'ютерних технологіях трирівнева архітектура, синонім триланкова архітектура передбачає наявність наступних компонентів програми: [клієнтський](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) [застосунок](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) , підключений до [сервера застосунків](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80_%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BA%D1%96%D0%B2), який в свою чергу підключений до [серверу бази даних](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85).  
[Клієнт](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) — це [інтерфейсний](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0) ([графічний](https://uk.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D1%96%D1%87%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81)) компонент, який представляє перший рівень, власне [застосунок](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) для кінцевого користувача. Перший рівень не має прямих зв'язків з [базою даних](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) (за вимогами безпеки), не навантажений основною бізнес-логікою (за вимогами масштабованості) і зберігає стан програми (за вимогами надійності). На перший рівень винесена найпростіша бізнес-логіка: інтерфейс авторизації, алгоритми шифрування, перевірка значень, що вводяться, на допустимість і відповідність формату, нескладні операції (сортування, групування, підрахунок значень) з даними, вже завантаженими на термінал.

* [Сервер застосунків](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80_%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BA%D1%96%D0%B2) розташовується на другому рівні. На другому рівні зосереджена більша частина бізнес-логіки. Поза ним залишаються фрагменти, що експортуються на термінали , а також розміщені в третьому рівні збережені процедури і тригери.
* Сервер бази даних забезпечує зберігання даних і виноситься на третій рівень. Це [об'єктно-орієнтована](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B0_%D0%B1%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) [СУБД](https://uk.wikipedia.org/wiki/%D0%A1%D0%A3%D0%91%D0%94). Коли третій рівень являє собою базу даних разом з [збереженими процедурами](https://uk.wikipedia.org/wiki/%D0%97%D0%B1%D0%B5%D1%80%D0%B5%D0%B6%D0%B5%D0%BD%D0%B0_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D0%B4%D1%83%D1%80%D0%B0), [тригерами](https://uk.wikipedia.org/wiki/%D0%A2%D1%80%D0%B8%D0%B3%D0%B5%D1%80_(%D0%B1%D0%B0%D0%B7%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85)) і схемою, яка описує застосунок в термінах реляційної моделі, то другий рівень будується як програмний інтерфейс, що зв'язує клієнтські компоненти з прикладною логікою бази даних.
* Також можна виділити четвертий шар - шар домену. Шар домену містить бізнес-логіку, що керує сайтом. Цей рівень включає в себе реалізацію сервісного контракту для функцій електронної комерції (також легко виставляється як веб-інтерфейси), моделі доменів, що містять бізнес-логіку, і насиченість бази даних. Моделі (з шаблону MVC) підтримують бізнес-логіку. Моделі ресурсів несуть відповідальність за взаємодію з базами даних, наприклад, для зберігання даних продукту чи даних клієнта.

Програмна реалізація шару уявлень (клієнтської частини веб-сервісу) здійснена за допомогою адаптивного CSS-фреймворку TwitterBootstrap. Для візуалізації даних використана бібліотека jQuery UI, amCharts. Серверна частина веб-сервісу реалізована за допомогою фреймворку yii2. В якості системи управління базою даних використана MуSQL.

TwitterBootstrap — це клієнтський [фреймворк](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B8%D0%B9_%D0%BA%D0%B0%D1%80%D0%BA%D0%B0%D1%81), тобто інтерфейс для користувача, на відміну від коду серверної сторони, який знаходиться на [сервері](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80). [Репозиторій](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BF%D0%BE%D0%B7%D0%B8%D1%82%D0%BE%D1%80%D1%96%D0%B9_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) з даним фреймворком є одним з найбільш популярних на [GitHub](https://uk.wikipedia.org/wiki/GitHub).  
jQuery UI — [бібліотека JavaScript](https://uk.wikipedia.org/wiki/%D0%91%D1%96%D0%B1%D0%BB%D1%96%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_JavaScript) з відкритим вихідним кодом для створення [насиченого інтерфейсу користувача](https://uk.wikipedia.org/wiki/%D0%9D%D0%B0%D1%81%D0%B8%D1%87%D0%B5%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) в [веб-застосунках](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA), частина проекту [jQuery](https://uk.wikipedia.org/wiki/JQuery). Побудована поверх основної бібліотеки [jQuery](https://uk.wikipedia.org/wiki/JQuery) та надає розробникові спрощений доступ до її функцій взаємодії, анімації та ефектів, а також набір [віджетів](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%B6%D0%B5%D1%82).   
«AmCharts» - велика онлайн-бібліотека, яка пропонує графіки з візуалізацією різних даних.  
Yii2-  це високопродуктивний веб-[фреймворк](https://uk.wikipedia.org/wiki/%D0%A4%D1%80%D0%B5%D0%B9%D0%BC%D0%B2%D0%BE%D1%80%D0%BA), написаний на [PHP](https://uk.wikipedia.org/wiki/PHP), реалізує парадигму [модель-вид-контролер](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C-%D0%B2%D0%B8%D0%B4-%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D0%B5%D1%80). Схематично зображено на рисунку 1.4

Рисунок 1.4 - Діаграма взаємодії між компонентами шаблону

[](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB:ModelViewControllerDiagram2.svg)

Моде́ль–вигляд–контро́лер (або Модель–представлення–контролер, [англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) *Model-view-controller*, MVC) — [архітектурний шаблон](https://uk.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%BD%D1%96_%D1%88%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD%D0%B8_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F), який використовується під час проектування та розробки [програмного забезпечення](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F).

Цей шаблон передбачає поділ системи на три взаємопов'язані частини: [модель даних](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85), вигляд ([інтерфейс користувача](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0)) та модуль керування. Застосовується для відокремлення даних (моделі) від інтерфейсу користувача (вигляду) так, щоб зміни інтерфейсу користувача мінімально впливали на роботу з даними, а зміни в моделі даних могли здійснюватися без змін інтерфейсу користувача.

Мета шаблону — гнучкий дизайн програмного забезпечення, який повинен полегшувати подальші зміни чи розширення програм, а також надавати можливість повторного використання окремих компонентів програми. Крім того використання цього шаблону у великих системах сприяє впорядкованості їхньої структури і робить їх більш зрозумілими за рахунок зменшення складності.

MySQL — [вільна](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%BB%D1%8C%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) [система керування реляційними базами даних](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D1%80%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B8%D0%BC%D0%B8_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85).

MySQL — компактний [багатопотоковий](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B3%D0%B0%D1%82%D0%BE%D0%BF%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%B2%D1%96%D1%81%D1%82%D1%8C) [сервер баз даних](https://uk.wikipedia.org/w/index.php?title=%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80_%D0%B1%D0%B0%D0%B7_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85&action=edit&redlink=1). Характеризується високою швидкістю, стійкістю і простотою використання.

MySQL вважається гарним рішенням для малих і середніх застосувань. [Сирцеві коди](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%80%D1%86%D0%B5%D0%B2%D1%96_%D0%BA%D0%BE%D0%B4%D0%B8) [сервера](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80) компілюються на багатьох платформах. Найповніше можливості сервера виявляються в [UNIX](https://uk.wikipedia.org/wiki/UNIX)-системах, де є підтримка [багатопоточності](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B3%D0%B0%D1%82%D0%BE%D0%BF%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%B2%D1%96%D1%81%D1%82%D1%8C), що підвищує продуктивність системи в цілому.

Можливості сервера MySQL:

* простота у встановленні та використанні;
* підтримується необмежена кількість користувачів, що одночасно працюють із БД;
* кількість рядків у таблицях може досягати 50 млн;
* висока швидкість виконання команд;
* наявність простої і ефективної системи безпеки.

# СТВОРЕННЯ, НАЛАШТУВАННЯ ТА ПРОСУВАННЯ САЙТУ

## **2.1 Вибір засобів проектування web-сайтів**

**2.1.1 Інтегроване середовище розробки PHPStorm**

На сьогоднішній момент існує багато потужних засобів створення програмної продукції. Для розроби цього додатку був обраний PHPStorm, він має дуже зручний інтерфейс, а також він однаково гарно підтримує синтаксис як PHP так і JS.

JetBrains PhpStorm — комерційне крос-платформове [інтегроване середовище розробки](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D1%81%D0%B5%D1%80%D0%B5%D0%B4%D0%BE%D0%B2%D0%B8%D1%89%D0%B5_%D1%80%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B8) для [PHP](https://uk.wikipedia.org/wiki/PHP),яке розробляється компанією [JetBrains](https://uk.wikipedia.org/wiki/JetBrains) на основі платформи [IntelliJ IDEA](https://uk.wikipedia.org/wiki/IntelliJ_IDEA).

PhpStorm являє собою інтелектуальний редактор для [PHP](https://uk.wikipedia.org/wiki/PHP), [HTML](https://uk.wikipedia.org/wiki/HTML) і [JavaScript](https://uk.wikipedia.org/wiki/JavaScript) з можливостями аналізу коду на льоту, запобігання помилок у [сирцевому коді](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%80%D1%86%D0%B5%D0%B2%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4) і автоматизованими засобами [рефакторинга](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%BD%D0%B3) для PHP і JavaScript. Автодоповнення коду в PhpStorm підтримує специфікацію PHP 5.3/5.4/5.5/5.6/7.0/7.1 (сучасні і традиційні проекти), включаючи генератори, співпрограми, простори імен, замикання, типажі і синтаксис коротких масивів. Присутній повноцінний [SQL](https://uk.wikipedia.org/wiki/SQL)-редактор з можливістю редагування отриманих результатів запитів.

PhpStorm розроблений на основі платформи IntelliJ IDEA, написаної на [Java](https://uk.wikipedia.org/wiki/Java). Користувачі можуть розширити функціональність середовища розробки за рахунок установки [плаґінів](https://uk.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D2%91%D1%96%D0%BD), розроблених для платформи IntelliJ, або написавши власні плаґіни.

Вся функціональність [WebStorm](https://uk.wikipedia.org/wiki/WebStorm) включена в PhpStorm.

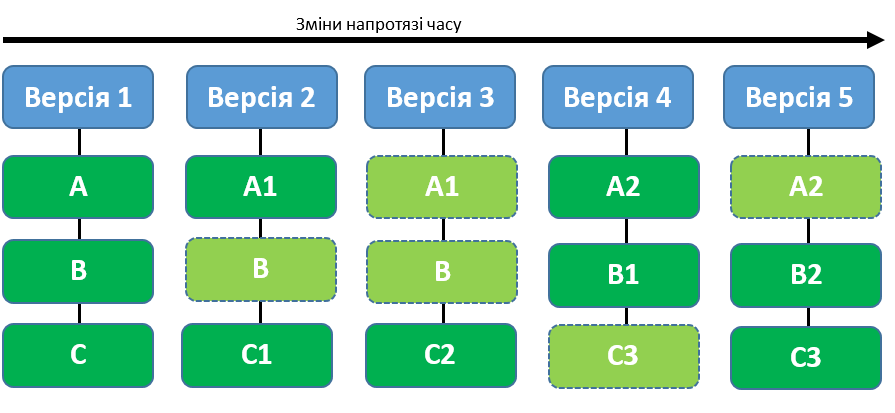
2.1.2 Система керування версіями та місце зберігання коду

Git — [розподілена система керування версіями](https://uk.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B7%D0%BF%D0%BE%D0%B4%D1%96%D0%BB%D0%B5%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8) [файлів](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB) та спільної роботи. Git є однією з найефективніших, надійних і високопродуктивних [систем керування версіями](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8), що надає гнучкі засоби нелінійної розробки, що базуються на відгалуженні і злитті [гілок](https://uk.wikipedia.org/wiki/%D0%93%D1%96%D0%BB%D0%BA%D0%B0_(%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8)). Для забезпечення цілісності історії та стійкості до змін заднім числом використовуються криптографічні методи, також можлива прив'язка [цифрових підписів](https://uk.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B8%D0%B9_%D0%BF%D1%96%D0%B4%D0%BF%D0%B8%D1%81) розробників до тегів і комітів.

Віддалений доступ до [репозиторіїв](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BF%D0%BE%D0%B7%D0%B8%D1%82%D0%BE%D1%80%D1%96%D0%B9) Git забезпечується git-[демоном](https://uk.wikipedia.org/wiki/%D0%94%D0%B5%D0%BC%D0%BE%D0%BD_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%B0)), [SSH](https://uk.wikipedia.org/wiki/SSH) або [HTTP](https://uk.wikipedia.org/wiki/HTTP) [сервером](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80). [TCP](https://uk.wikipedia.org/wiki/TCP)-сервіс git-daemon входить у дистрибутив Git і є разом з SSH найпоширенішим і надійним методом доступу. Метод доступу HTTP, незважаючи на ряд обмежень, дуже популярний в контрольованих мережах, тому що дозволяє використання існуючих конфігурацій мережевих фільтрів.  
В своїй базі Git зберігає все по хешам файлів. Як хешуюча функція використовується [SHA-1](https://uk.wikipedia.org/wiki/SHA-1). Перед кожним збереженням файлів Git обчислює [SHA-1](https://uk.wikipedia.org/wiki/SHA-1) [хеш](https://uk.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88) файлу і отриманий хеш стає індексом файлу в Git. Використовуючи [хеш](https://uk.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88) Git легко відслідковує зміни в файлах.

Схематично роботу з системою контролю версій зображено на рисунку 1.5

Рисунок 1.5 - Дані як зліпки стану проекту в часі



Для зберігання проекту був вибраний веб-сервіс Bitbucket.  
Bitbucket — веб-сервіс для [хостингу](https://uk.wikipedia.org/wiki/%D0%A5%D0%BE%D1%81%D1%82%D0%B8%D0%BD%D0%B3) проектів на базі [систем керування версіями](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8): [Mercurial](https://uk.wikipedia.org/wiki/Mercurial) та [Git](https://uk.wikipedia.org/wiki/Git) . Bitbucket надає як безкоштовні так і платні послуги. Bitbucket є аналогом [GitHub](https://uk.wikipedia.org/wiki/GitHub), проте на відміну від [GitHub](https://uk.wikipedia.org/wiki/GitHub), у якого при безкоштовному профілі файли зберігаються лише у відкритому доступі, Bitbucket дозволяє безкоштовно створювати приватні репозиторії з можливостю спільної роботи з файлами до 5-ти користувачів.

Bitbucket Pipelines - це інтегрована служба CI / CD, вбудована в Bitbucket. Це дозволяє автоматично створювати, перевіряти і навіть розгорнути кодекс, на основі файлу конфігурації у вашому сховищі. По суті, ми створюємо контейнери в хмарі для вас. Всередині цих контейнерів ви можете запускати команди (як ви можете на локальній машині), але з усіма перевагами свіжої системи, настроєної на ваш смак.

Composer — [менеджер пакетів прикладного рівня](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BF%D0%B0%D0%BA%D1%83%D0%BD%D0%BA%D0%B0%D0%BC%D0%B8) для [мови програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [PHP](https://uk.wikipedia.org/wiki/PHP) що забезпечує стандартний формат для управління залежностями у програмному забезпеченні та необхідними [бібліотеками](https://uk.wikipedia.org/wiki/%D0%91%D1%96%D0%B1%D0%BB%D1%96%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC)

Composer працює з [командного рядка](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D0%BD%D0%BE%D0%B3%D0%BE_%D1%80%D1%8F%D0%B4%D0%BA%D0%B0) і встановлює залежності (наприклад, бібліотек) для застосунку. Він також дозволяє користувачам встановлювати PHP пакети, доступні на «Packagist», який є його основним сховищем, яке містить доступні пакети. Він також реалізує [автозавантажувач](https://uk.wikipedia.org/w/index.php?title=%D0%90%D0%B2%D1%82%D0%BE%D0%B7%D0%B0%D0%B2%D0%B0%D0%BD%D1%82%D0%B0%D0%B6%D1%83%D0%B2%D0%B0%D1%87&action=edit&redlink=1) класів, для встановлених бібліотек і це полегшує використання коду від сторонніх розробників.

Composer використовується як складова частина декількох популярних PHP проектів з [відкритим вихідним кодом](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%BA%D1%80%D0%B8%D1%82%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F), наприклад: [Laravel](https://uk.wikipedia.org/wiki/Laravel), [Symfony](https://uk.wikipedia.org/wiki/Symfony).

## Розпочати роботу з додатком

Yii 2 Основний шаблон проекту - це скелет [Yii 2] (http://www.yiiframework.com/), що найкраще підходить для

Створюючи невеликі проекти.  
Шаблон містить основні функції, включаючи вхід користувача / вихід із системи та сторінку контактів. Вона включає в себе всі загальнодоступні конфігурації, які дозволяють зосередитись на додаванні нових особливості програми.

[![Остання стабільна версія](https://poser.pugx.org/yiisoft/yii2-app-basic/v/stable.png)](https://packagist.org/packages/yiisoft/yii2-app-basic)

[![Загальне завантаження](https://poser.pugx.org/yiisoft/yii2-app-basic/downloads.png)](https://packagist.org/packages/yiisoft/yii2-app-basic)

[![Статус збірки](https://travis-ci.org/yiisoft/yii2-app-basic.svg?branch=master)](https://travis-ci.org/yiisoft/yii2-app-basic)

## СТРУКТУРА КАТАЛОГУ

assets/ містить визначення активів

commands/ містить консольні команди (контролери)

config містить конфігурації програми

controllers/ містить класи веб-контролера

mail/ містить файли перегляду для електронних повідомлень

models/ містить моделі класів

runtime/ містить файли, створені під час виконання

tests/ містить різні тести для базової програми

vendor/ містить залежні сторонні пакети

views/ містить файли перегляду для веб-програм

web/ містить сценарій входу та веб-ресурси

ВИМОГИ  
Мінімальна вимога цього шаблону проекту, що ваш веб-сервер підтримує PHP 7.0.

ВСТАНОВЛЕННЯ  
Встановити через Composer — [менеджер пакетів прикладного рівня](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BF%D0%B0%D0%BA%D1%83%D0%BD%D0%BA%D0%B0%D0%BC%D0%B8) для [мови програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [PHP](https://uk.wikipedia.org/wiki/PHP) що забезпечує стандартний формат для управління залежностями у програмному забезпеченні та необхідними [бібліотеками](https://uk.wikipedia.org/wiki/%D0%91%D1%96%D0%B1%D0%BB%D1%96%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC).  
Якщо у вас немає [Composer] (http://getcomposer.org/), ви можете встановити його, дотримуючись інструкцій.

[getcomposer.org] (http://getcomposer.org/doc/00-intro.md#installation-nix).

Потім ви можете встановити цей шаблон проекту за допомогою наступної команди:

php composer.phar create-project --prefer-dist --stability=dev yiisoft/yii2-app-basic basic

Тепер ви маєте можливість отримати доступ до програми за допомогою такої URL-адреси, вважаючи, що `basic` це каталог

безпосередньо під веб-кореневим каталогом.

http://localhost/basic/web/

Встановити з архіву  
Витяг архівного файлу, завантаженого з [yiiframework.com] (http://www.yiiframework.com/download/) на каталог з ім'ям `basic`, який знаходиться безпосередньо під веб-кореневим каталогом.

Встановити файл перевірки cookie у файлі `config / web.php` до якогось випадкового секретного рядку

php

'request' => [

// !!! вставте секретний ключ у наведений нижче приклад(якщо він порожній) - це потрібно для перевірки файлів cookie

'cookieValidationKey' => '< Секретний випадковий рядок >',

],

Потім ви можете отримати доступ до програми за допомогою такої URL-адреси:

http://localhost/basic/web/

## КОНФІГУРАЦІЯ

База даних

Відредагуйте файл `config / db.php` з реальними даними, наприклад:

php

return [

'class' => 'yii\db\Connection',

'dsn' => 'mysql:host=localhost;dbname=yii2basic',

'username' => 'root',

'password' => '1234',

'charset' => 'utf8',

];

ПРИМІТКИ  
- Yii не створює базу даних, це потрібно зробити вручну, перш ніж отримувати доступ до неї.

- Необхідно перевірити та відредагувати інші файли в каталозі `config /`, щоб налаштувати програму, як потрібно.

## ТЕСТУВАННЯ

Тести містяться в каталозі `tests`. Вони розроблені з [Codeception PHP Testing Framework] (http://codeception.com/).

За умовчанням існують 3 тестові набори:

- `unit` (Одиничні тести)

- `functional` (Функціональні тести)

- `acceptance` (Тести прийому)

Тести можуть бути виконані запуском

vendor/bin/codecept run

Наведена вище команда виконуватиме одиничні та функціональні тести. Одиничні тести - тестування компонентів системи, в той час як функціональні тести для тестування взаємодії з користувачем. Тести прийому за замовчуванням відключені, оскільки вони потребують додаткового налаштування вони виконують тестування в реальному браузері.

Запуск приймальних випробувань

Для виконання приймальних випробувань виконайте такі дії

1. Перейменуйте `tests / acceptance.suite.yml.example` на` tests / acceptance.suite.yml`, щоб дозволити налаштування пакету

2. Замініть пакет `codeception / base` в` composer.json` за допомогою `codeception / codeception` для встановлення повнофункціональної   версії Codeception

3. Оновити залежності з Composer

     composer update

4. Завантажте [Selenium Server] (http://www.seleniumhq.org/download/) і запустіть його:    java-jar ~ / selenium-server-standalone-x.xx.x.jar

    У разі використання Selenium Server 3.0 з браузером Firefox від v48 або Google Chrome з V53 необхідно завантажити [GeckoDriver] (https://github.com/mozilla/geckodriver/releases) або [ChromeDriver] (https://sites.google .com / a / chromium.org / chromedriver / downloads) і запустити Selenium з нею:

для Firefox: java -jar -Dwebdriver.gecko.driver=~/geckodriver ~/selenium-server-standalone-3.xx.x.jar

для Google Chrome: java -jar -Dwebdriver.chrome.driver=~/chromedriver ~/selenium-server-standalone-3.xx.x.jar   
Як альтернативний спосіб можливе використання вже налаштованного контейнер Docker зі старими версіями Selenium та Firefox

docker run --net=host selenium/standalone-firefox:2.53.0

5. (Необов'язково) Створіти базу даних `yii2\_basic\_tests` та оновіти її, застосуючи міграції, якщо у вас є :     tests/bin/yii migrate

Конфігурацію бази даних можна знайти за адресою `config / test\_db.php`.

6. Запустити веб-сервер:  tests/bin/yii serve

7. Тепер ви маємо можливість запустити всі доступні тести

   - виконувати всі доступні тести: vendor/bin/codecept run

   -запустити прийомні тести: vendor/bin/codecept run acceptance

   - запускати лише одиничні та функціональні тести: vendor/bin/codecept run unit,functional

Підтримка коду покриття коду  
За умовчанням кодове покриття вимикається у файлі конфігурації `codeception.yml ', щоб розпізнати необхідні рядки, щоб збирати покриття коду. Необхідно запустити свої тести та збирати покриття за допомогою такої команди :

збирати охоплення для всіх тестів: vendor/bin/codecept run -- --coverage-html --coverage-xml

поширювати покриття тільки для одиничного тестування: vendor/bin/codecept run unit -- --coverage-html --coverage-xml

збирати охоплення для одиничних та функціональних випробувань: vendor/bin/codecept run functional,unit -- --coverage-html --coverage-xml

Вихідний код можна побачити в каталозі `tests / \_output '

## Основні можливості додатку

### Авторизація та регистрація Додаток має відкриту регестрацію , що дозволяє кожному бажаючому створити свій аккаунт для користування. Для регестрації необхідно ввести свою пошту та вигадати та ввести пароль. Після регестрації користувача автоматично буде перенаправлено до системи. Для авторизації є можливість введення логіну та паролю, котрий був створений при регестрації , або використати функціонал регестрації через соціальні мережі. Далі на рисунку 1.6 та 1.7 наведені макети сторінки регестрації та авторизації

### Рисунок 1.6 - Макет сторінки регестрації

### Рисунок 1.7 – Макет сторінки авторизації При використанні авторизації за допомогою соціальних мереж попередня регестрація у додатку не потрібно.

Для входу за допомогою соціальних мереж використано **OAuth**  2.0

**OAuth** — це [відкритий стандарт](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%BA%D1%80%D0%B8%D1%82%D0%B8%D0%B9_%D1%81%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82) [авторизації](https://uk.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D1%96%D1%8F), який дозволяє користувачам відкривати доступ до своїх приватних даних (фотографії, відео, списки контактів), що зберігаються на одному сайті, іншому сайту, без необхідності вводу імені користувача та паролю.

OAuth дозволяє користувачам роздавати сайтам маркери доступу, до даних що розміщуються на сайтах-сервісах. Кожен маркер доступу надає доступ конкретному сайту до конкретних та на визначений термін (наприклад, на наступні 2 години). Це дозволяє користувачам надавати доступ третім сайтам до їх інформації, що зберігається на інших сайтах — постачальниках послуг, не передаючи повною мірою самих даних та без застосування імені/паролю.

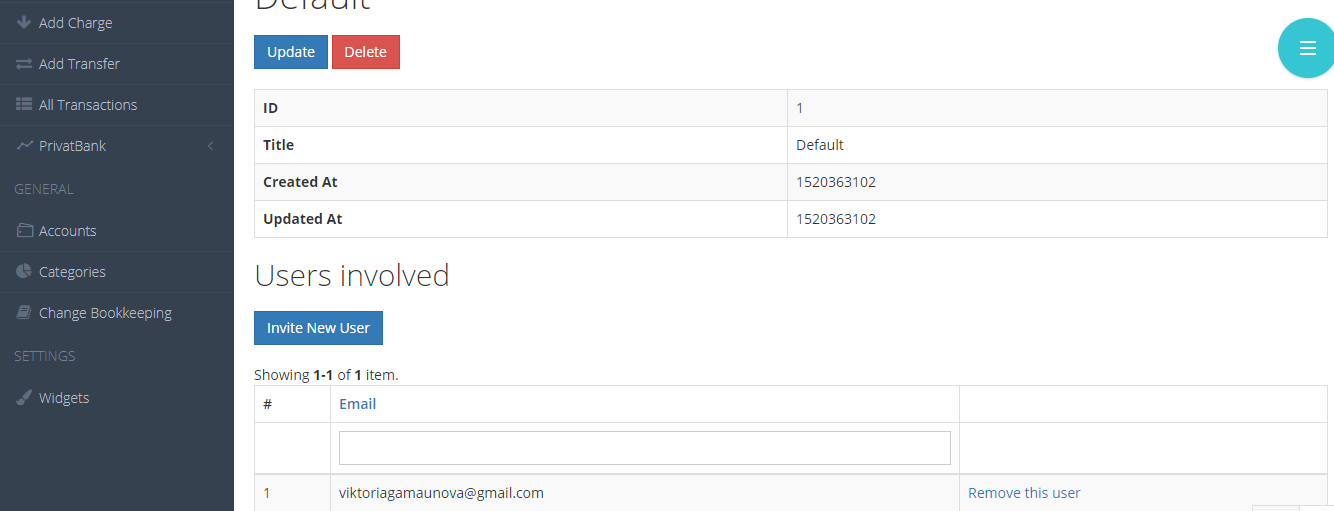
OAuth 2.0 — наступне покоління протоколу OAuth, зворотно не сумісне з OAuth 1.0. OAuth 2.0 фокусується на простоті розробки клієнтської частини, забезпечуючи спеціальні потоки дозволу для [веб-застосунків](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA), настільних [застосунків](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA), [мобільних телефонів](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B1%D1%96%D0%BB%D1%8C%D0%BD%D0%B8%D0%B9_%D1%82%D0%B5%D0%BB%D0%B5%D1%84%D0%BE%D0%BD). Специфікація розробляється в рамках робочої групи [IETF](https://uk.wikipedia.org/wiki/IETF) OAuth.

Нове Graph API від [Facebook](https://uk.wikipedia.org/wiki/Facebook) підтримує тільки OAuth 2.0 і є найбільшою реалізацією нового стандарту. З 2011 року Google додав експериментальну підтримку OAuth 2.0 в своє API.

### Можливість керування транзакціями між аккаунтами

Додаток позиціоную себе як засіб для збереження моніторину саме домашньої бухгалтерії та розраховано, що користуватися ним будуть сім’ї, тому додатку має функціонал котрий дозволяє, управляти транзакціями з різних аккаунтів.  
для цього необхідно кожному з членів сім’ї зареєструватися в додатку, з одного з аккаунтів відправити запрошення на інші аккаунти, використовуючи пошту . Далі усі користувачі, котрі отримали запрошення мають можливість, редагувати, створювати та переглядати транзакції за сімейною бухгалтерією.

Далі на рисунку 1.8 зображено макет сторінки для вибору, перегляду, створення, видалення бухгалтерій.

Рисунок 1.8 –Макет сторінки бухгалтерії

### Багатовалютність

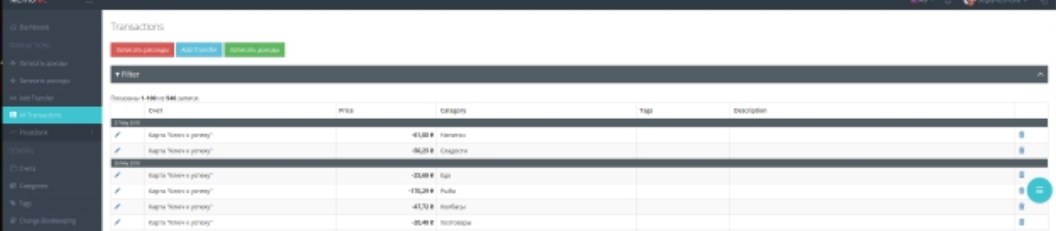
Додаток підтримую багато валют. Завдяки інтеграції API Приват Банку, можливе використання усіх валют, котрі використовує банк, якщо нам недостатньо валют, є можливість запрограмувати додаток на потрібну нам валюту. Конвертація валют виконується по курсу на даний проміжок часу.

### Теги та категорії

Теги – це ідентифікатор для категоризації, опису, пошуку даних і завдання внутрішньої структури;  
Категорії – це таксоном

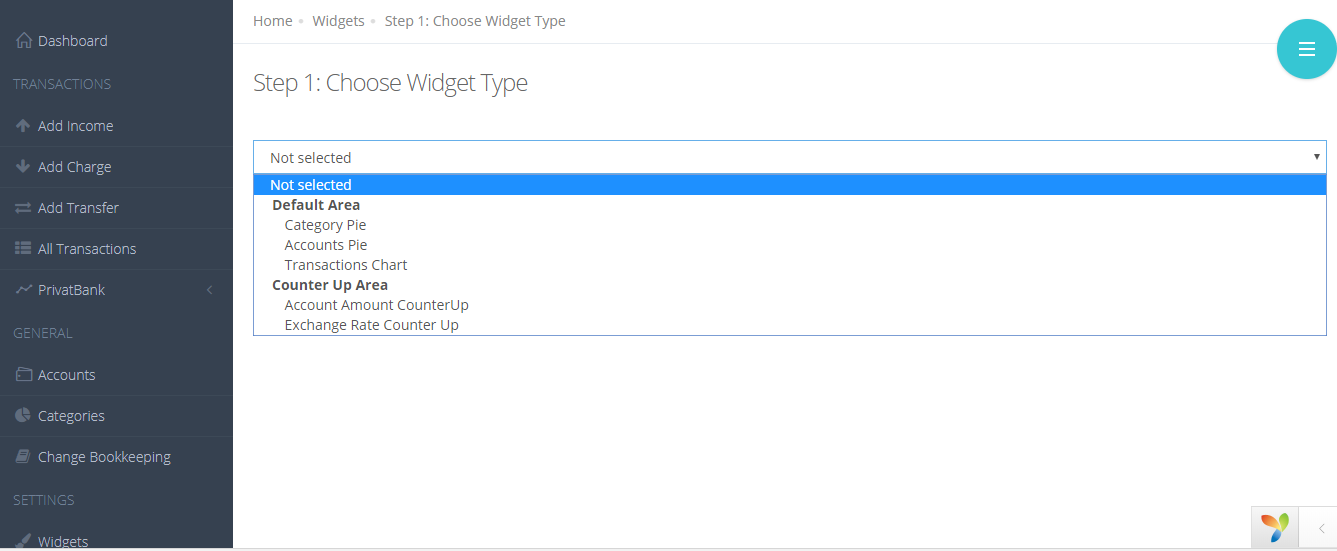
Категорії та теги створюються самим користувачем.

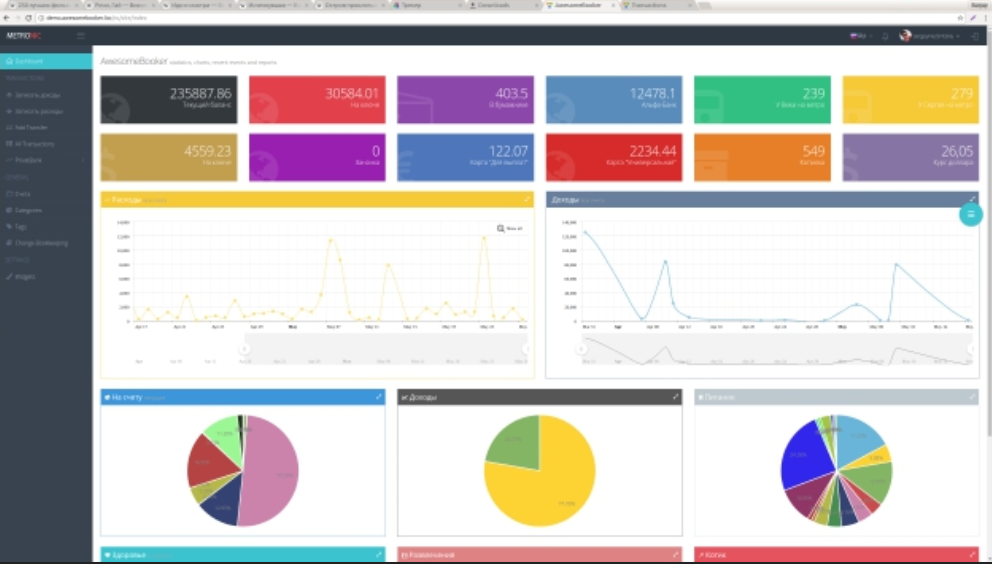
У транзакції може бути лише одна категорія збражено на малюнку 1.9. Категорії мають деревоподібну структуру, тобто кожна категорія може мати під категорію, може бути багато рівнів. У транзакції може бути багато тегів. Теги мають зв'язок багато до багатьох.

Рисунок 1.9 – Макет сторінки транзакцій

### Віджети

Додаток дозволяє додавати віджети у вигляді різноманітних графіків доходів та витрат, групувати за категоріями та тегами, а також візуалізувати залишки на рахунках на рисунку 1.11 відображено зовнішній вигляд віджетів . Перелік віджетів наведено нижче на рисунку 1.10.

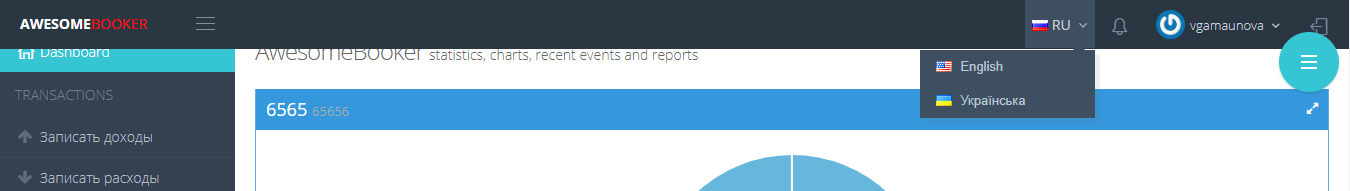
Рисунок 1.10 – Макет першого кроку створення віджету

Рисунок 1.11 – Макет головної сторінки на котрій зображені віджети

Інтеграція с ПриватБанком  
Інтеграція дозволяє отримати данні о виписках з карти мерчанта, та можливість подальшої категорізації цих даних.  
Можливість отримати конвертацію курсу валют для не гривневих транзакцій.

### Багатомовність

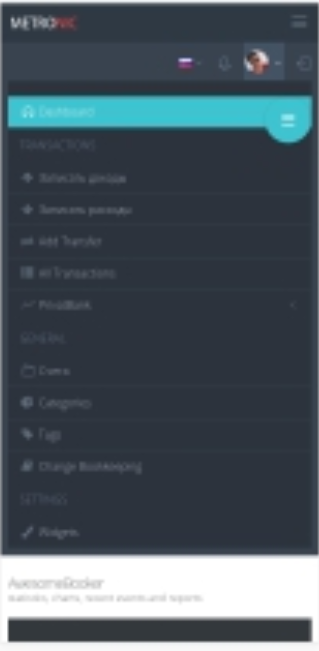
Додаток підтримує українську, російську то англійську мову. Можливе швидке та легке розширення мов рисунок 1.12.

Рисунок 1.12 – Макет шапки додатку.

### Адаптація додатку під мобільні пристрої

Додаток адаптовано під усі мобільні пристрої завдяки фреймворку TwitterBootstrap, та використовуючи «резинову» верстку вдалося отримати даний результат.

Рисунок 1.13 - Планшетна орієнтація

 Рисунок 1.14 - Мобільна орієнтція

### 

### Запити, котрі приймає додаток

\* - поля, що обов’язково мають буди передані.   
GET / або GET /site/index - головна сторінка сайту- повертає віджети, що містять різноманітні статестичні дані стосовно використання додатку.

/site/contact - Вертає контактну форму

/site/contact - Приймає запити для зв'язку з адміністратором сайту. Отримані дані надсилаються у вигляді листа на адмінстраторський поштову адресу. Для запиту нема необхідності бути авторизованим.  
Параметри, котрі приймаються:

\*ContactForm[name] – контактне імя

\*ContactForm[email] – контактна поштова адреса

\*ContactForm[subject] – предмет запиту

\*ContactForm[body] – текст запиту

\*ContactForm[[verifyCode] – каптча

Аутентифікація  
GET /user/login – відображає сторінку входу

POST /user/login – приймае параметри після відправки форми

Параметри, котрі приймаються:

\*login-form[login] – Логин або email

\*login-form[password] – пароль

login-form[rememberMe] – запам’ятати на сайті та не розлогінювати

GET/user/forgot – відображає форму відновлення паролю

POST/user/forgot – приймає значення

\* recovery-form[email] з форми  
GET /user/register- відображає сторінку регестрації

POST /user/register – приймає параметри:

\* register-form[username]

\* register-form[email]

\* register-form[password]

GET /user/auth - Логин/регистрация через социальные сети, используя технологию Auth2.0. Прийма параметри:

\* authclient – ім’я користувача, через котре буде проходити авторизація

POST /user/auth - приймає : \* authclient – ім’я користувача, через котре буде проходити авторизація.

GET /user/confirm/{userId}/{registrationHash}- підтверження регестрації

POST /user/recover/{user-id}/{recovery-password-token} – приймає параметр:

\* recovery-form[password]

GET /user/settings/profile – Повертає форму для редагування профілю користувача

POST /user/settings/profile – отримує :

Profile[name] - ім’я

Profile[email] – публічна пошта

Profile[website] - вебсайт

Profile[location] – адреса/місцезнаходження

Profile[timezone] – часова зона

Profile[gravatar] – email для сервісу gravatar.ru

Profile[about] – коротка інформація про себе

GET /user/settings/account - повертає форму для редагування налаштунків акаунту користувача

POST /user/settings/account – приймає параметр:

\* Account[email] – email

\* Account[usename] – логин

Account[password] - пароль

Account[password-repeat] - пароль(для порівняння)

GET /user/settings/networks –відоражая список соціальних мереж, що прикріплені до строінки.  
GET /user/logout – вихід з додатку

Транзакції

GET /transaction /index- сторінка всіх транзакцій- відображає таблицю транзакцій.

Приймає парметри :

TransactionSearch[account\_id]

TransactionSearch[category\_id]

TransactionSearch[tag\_id]

TransactionSearch[date]

TransactionSearch[date\_from]

TransactionSearch[date\_to]

Page

GET /transaction/update – відображає форму редагування. Приймає обов’язковий параметр id-ідентифікатор транзакції.

POST /transaction/update – відновлює дані.

\* Transaction[id]

\* Transaction[price]

\* Transaction[base\_price]

\* Transaction[category\_id]

\* Transaction[tag\_ids][]

\* Transaction[description]

GET /transaction/income - сторінка доходів- відображає форму для запису дходу  
POST /transaction/income – для збереження даних з форми.

Приймає дані:

\* Income[price]

\* Income[account\_id]

Income[category\_id]

Income[tag\_ids][]

Income[description]

GET / transaction/charge- сторінка витрат- відображає форму для запису витрат  
POST / transaction/charge– для збереження даних з форми.

Приймає дані:

\* Charge[price]

\* Charge[account\_id]

Charge[category\_id]

Charge[tag\_ids][]

Charge[description]

GET /transaction/transfer- сторінка переказів- відображає форму для запису переказів  
POST /transaction/transfer– для збереження даних з форми.

Приймає дані:

\* Transfer[account\_id\_src] – Рахунок з котрого приходить транзакція

\* Transfer[account\_id\_dst] – Рахунок на котрий приходить транзакція

\* Transfer[price\_src] – сума транзакції в валюті рахунку з якого списуються кошти.

Transfer[prrice\_dst] – сума транзакцій в валюті рахунку, на котрий зараховуються кошти. (у разі міжвалютної транзакції), не потрібна якщо в одній валюті.

POST /transaction/delete – для видалення транзакцій передаємо ідентифікатор

GET /privatbank/transactions/index- відображає транзакції, що прийшли з Приват Банку

GET /privatbank/credit-cards/index- відображає зареєстровані користувачем карти.

GET /privatbank/credit-cards/create- відображає форму для реєстрації картки.

POST /privatbank/credit-cards/create- для збереження даних

\* CreditCard[number]

\* CreditCard[title]

\* CreditCard[account\_id]

CreditCard[is\_active]

CreditCard[merchant\_id]

CreditCard[merchant\_secret]

GET /privatbank/credit-cards/update- для відображення форми редагування даних карти.

Приймає параметри:

\* id

POST /privatbank/credit-cards/update – дані, що приймає роут.

\* id

\* CreditCard[number]

\* CreditCard[title]

CreditCard[merchant\_id]

CreditCard[merchant\_secret]

\* CreditCard[account\_id]

CreditCard[is\_active]

GET /privatbank/credit-cards/view- для перегляду карти,

Приймає параметри:

\* id

POST /privatbank/credit-cards/delete- для видаенння карти

Приймає параметри:

\* id

POST /privatbank/credit-cards/import – імпорт транзакцій з api Приват Банку в додаток.

Приймає параметри:

\* id

GET /privatbank/transactions або privatbank/transactions/index – відображає список транзакцій імпортованих з Приват Банку.  
POST /privatbank/transactions/import- імпорт транзакцій.  
приймає параметри:

TransactionImportForm[date]

TransactionImportForm[date\_from]

TransactionImportForm[date\_to]

GET /privatbank/transactions/convert-all – перетворює усі імпортовані транзакції у транзакції в додатку.  
GET /privatbank/transactions-map/\* - налаштування карти для коректного перетворення імпортованих транзакцій в транзакції додатку.  
Використовує стандартний CRUD .  
Також за стандартним CRUD виконані роути –

/account/\*- для рахунків  
/category/\*- для категорій за котрими групують транзакції  
/tag/\*- для тегів, за котрими групують транзакції

/bookkeping/\* - для зв’язаних аккаунтів

кожний з них можна переглянути, редагувати, створити, та видалити.   
POST /bookkeping/remove – приймає параметр   
\*id -ідентифікатор користувача, которого необхідно виділити.

POST /invite/create – відправляє запрошення користувачеві для сумісного ведення бухгалтерії.

Приймає параметри:

\* bookkeeping\_id

\* Invite[invitee\_id] – ідентифікатор користувача, котрого запрошують для ведення сумісної бухгалтерії

GET /invite/accept – для прийняття запрошення на ведення бухгалтерії  
приймає параметр:

\* token

### Консольні команди

privat-bank/import-transactions

language/init

Налаштування додатку на сервісі Амазані

Крок 1. Відкриття панелі управління EC2

Відкрити консоль управління AWS в новому вікні браузера. При цьому покрокове навчальний посібник залишиться відкритим. Знайдіть варіант EC2 під заголовком Compute і клацніть по ньому, щоб відкрити консоль Amazon EC2.

Крок 2. Створення та налагодження віртуальної машини

Відкрився екран консолі Amazon EC2. Натисніть Launch Instance.  
За допомогою Amazon EC2 можна вказати програмне забезпечення та параметри інстанси, який потрібно використовувати. На цьому екрані представлені варіанти вибору способу машини Amazon (AMI), що представляє собою шаблон з необхідною для запуску інстанси конфігурацією програмного забезпечення.

Тепер необхідно вибрати тип інстанси. Типи інстанси включають різні комбінації таких компонентів, як ЦПУ, пам'ять, сховище і мережеві можливості, і дозволяють вибрати відповідний набір ресурсів для додатків.

епер можна переглянути параметри, вибрані для інстанси: «AMI Details», «Instance Type», «Security Groups», «Instance Details», «Storage» і «Tags». Тут можна залишити значення за замовчуванням. Потім натисніть Launch в нижній частині сторінки

Крок 3. Створення пари ключів і запуск інстанси

Для підключення до віртуальної машини необхідна пара ключів. Вона використовується для входу в інстанси

Крок 4. Підключення до інстанси

Після запуску інстанси слід отримати пароль адміністратора і підключитися до інстанси з використанням протоколу віддаленого робочого столу (RDP)

Тестування юніт та функціональні

Для забезпечення якості та стабільності коду необхідно приділяти увагу написанню юніт-тестів, які ще на етапі розробки можуть виявити помилки та запобігти подальшим проблемам.

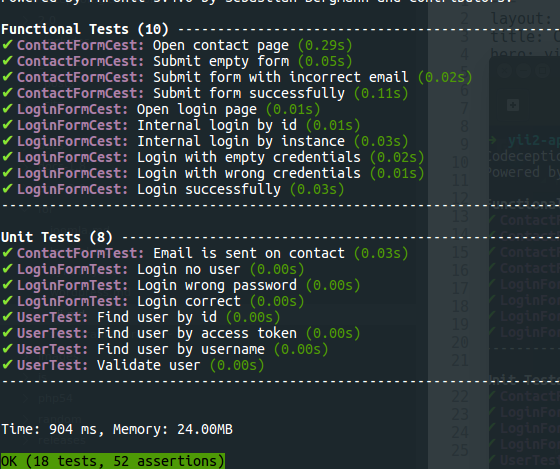
Yii Framework забезпечує базові та розширені шаблони застосунків. Обидва включають зразок Codeception тестів, таким чином, починаючи з Codeception, необхідно запустити новий проект Yii

Після створення проекту з базового шаблону бачимо папки та файл конфігурації codeception.yml. Там з самого початку є функціональні та модульні тестові апартаменти

Щоб запустити їх, необхідно в терміналі виконати команду:

./vendor/bin/codecept run

Отримані результаті можна побачити на рисунку 1.15

Рисунок 1.15 – Результати тестів  


Одиничні тести

Одиничні тести знаходяться в каталозі тестів / підрозділів і передбачають, що вони містять всі види підрозділів та тестування інтеграції.

Кожен тестовий випадок розширює клас Codeception \ Test \ Unit, який є стандартним форматом Codeception для тестування пристроїв. У Yii досить складно розробляти повністю ізольовані модульні тести, тому програма завантажується перед кожним тестом. Тести налаштовуються в тестах / unit.suite.yml з включеним модулем Yii2:

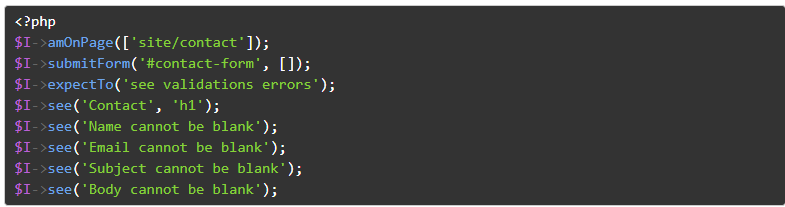
Цей модуль запускає програму Yii для тестового випадку та надає додаткові допоміжні методи для спрощення тестування. Він має лише частину orm та електронної пошти, щоб виключити методи, необхідні лише для функціонального тестування.

Доступ до класу $ this-> tester у тестовому випадку можна використовувати методами модуля Yii2.

Функціональні тести

Коли мова йде про перевірку справжніх особливостей веб-додатків, то лише одиничних тестів замало . Необхідно перевірити, як програма обробляє запити, які відповіді вона дає, які дані зберігаються в базі даних і так далі. Для тестування застосунків у навколишньому середовищі користувача, але без запуску реального веб-сервера або браузера, використовуються функціональні тести. Вони набагато простіші, ніж одиничні тести. Вони описують сценарії взаємодії в простій DSL, тому не потрібно звертатися безпосередньо до програми, а потрібно описати дії з точки зору користувача:

На рисунку 1.16 зображені дії з точки зору користувача  
Рисунок 1.16 – Код імулючий дії користувача



Таким чином, перевіряється не тільки форма на сайті, а й фактично виводиться програма, яку бачить користувач. Codeception пропонує стандартний набір дій, таких як amOnPage(перейти до сторінки), submitForm(відправити форму). Для тестування. Модуль Yii2 надає спеціальні методи, такі як amLoggedInAs (для швидкої автентифікації Функціональні тести написані всередині Cest-файлів, які є сценарним тестом формату Codeception

Щоб створити новій тест необхідно ввести

./vendor/bin/codecept g:cest functional MyNewScenarioCest

[PHPUnit](https://phpunit.de/manual/current/en/).  
Для юніт-тестування  зазвичай використовую фреймворк [PHPUnit](https://phpunit.de/manual/current/en/). Це загальноприйнятий стандарт, який повністю покриває всю зону відповідальності. Багато інструментів заточені на роботу саме з ним.

Щоб встановити PHPUnit та підключити його, використовуємо команду:

$ composer require --dev phpunit/phpunit

Ініціалізуємо базову структуру тесту

Перш за все, створюємо клас з суфіксом Test і успадковуємо його від \PHPUnit\_Framework\_TestCase класу з пакету PHPUnit:

class MyServiceTest extends \PHPUnit\_Framework\_TestCase

{

// ... тести на сервіс

}

Для ініціалізування базової структури тесту використовуємо:

* метод setUp(), в якому оголошуємо базові речі для ініціалізації класу, який тестуємо. Тут можна замокати вхідні параметри конструктора класу та створити як реальний об’єкт класу, який тестуємо, так і partial (частковий) мок;
* метод tearDown(), в якому бажано очищати пам’ять, використану поточним тестом;

Визначаємо тест-кейси

Далі вирішуємо, скільки тест-кейсів буде і як їх правильно розбити.

Кожний тест-кейс повинен покривати конкретну зону, проте він не має враховувати всі можливі кейси.

Викликаємо метод з необхідними тестовими даними

Після того як ми вибрали конкретний кейс, який хочемо протестувати, необхідно прописати виклик методу, який тестуємо, з необхідними тестовими даними (параметрами):

Визначаємо очікування

Уся суть юніт-тестування — перевірка поведінки методу залежно від вхідних даних. А отже, нам потрібно прописати те, що ми очікуємо від методу, який тестуємо, якщо викличемо його з певним набором параметрів, оголошених на попередньому кроці.

Використання Bitbucket Pipelines  
Bitbucket Pipelines - в першу чергу це сервіс, добре інтегрований з сховищем репозиторіїв Bitbucket.

Як зараз прийнято у всіх модних CI системах, ми додаємо в корінь сховища спеціальний файл, в даному випадку bitbucket-pipelines.yml, і далі на кожен push відправлений в Bitbucket запускаються завдання, описані в цьому файлі. Запуск відбувається цілком і повністю на серверної інфраструктури Bitbucket всередині Docker контейнерів.

Файл bitbucket-pipelines.yml виглядає досить типово і звичайно: сутність самого верхнього рівня - це pipeline. У нас може бути кілька різних Пайплайн прив'язаних до різних гілок або тегами вихідного коду. Причому можна вказувати маски glob patterns для імен гілок і тегів.

Далі всередині Пайплайн йде набір кроків. Кроки виконуються послідовно, якщо якийсь крок завершився з помилкою, то наступні не виконуються. Крок можна визначити, як запускаємий виключно вручну - зручно для реалізації деплоя по кнопці. Кожен крок починається з чистого аркуша - це новий Docker контейнер в якому автоматично доступні вихідні коди нашого застосування. Є можливість передавати артефакти між кроками.

Усередині кроків ми описуємо задачі. Задачі це по суті консольна команда - наприклад, спочатку робемо composer install, потім codecept run.

Щоб не качати composer пакети з інтернету кожен раз, сервіс надає механізм кешування. З коробки є заготовки для composer, для npm, але можна описати і свої правила, яку саме директорію кешувати.

А якщо для запуску тестів потрібна база даних (типу MySQL або PostrgreSQL) - в термінах Bitbucket Pipelines це «сервіси». Разом з запуском основного контейнера з тестами ми можемо запустити до п'яти додаткових сервісів, які також є запущеними Docker контейнерами доступними по мережі за їхніми іменами.

Об'єм оперативної пам'яті обмежений 4 Гб на все - основний контейнер + усі додаткові сервіси. Є можливість запускати контейнери з подвоєним об'ємом пам'яті, вказавши спеціальну опцію в yml файлі size: 2x  
Секретні дані (ключі, паролі) можна передати всередину оточення, в якому запускаються наші завдання в вигляді змінних оточення, попередньо додавши їх через веб-інтерфейс Bitbucket.

Код файлу bitbucket-pipelines.yml, котрий знаходиться в корні проекту.

**image:** magecom/php:dev7  
**pipelines:  
 default:** - **step:  
 caches:** - vendor  
 **script:** - composer install  
 - export CACHE\_DRIVER=file  
 - export SESSION\_DRIVER=file  
 - export DB\_CONNECTION=mysql  
 - cp ./init/common-local.php ./config/common-local.php  
 - cp ./init/console-local.php ./config/console-local.php  
 - cp ./init/test-local.php ./config/test-local.php  
 - cp ./init/web-local.php ./config/web-local.php  
 - php ./tests/bin/yii migrate --interactive=0 --migrationPath=@vendor/dektrium/yii2-user/migrations  
 - php ./tests/bin/yii migrate --interactive=0  
 - php ./tests/bin/yii migrate --interactive=0 --migrationPath=@app/modules/privatbank/migrations  
 - php ./tests/bin/yii language/init  
 - vendor/bin/codecept build  
 - vendor/bin/codecept run  
 **services:** - mysql  
**definitions:  
 services:  
 mysql:  
 image:** mysql  
 **environment:  
 MYSQL\_DATABASE: 'awesomebooker\_demo'  
 MYSQL\_RANDOM\_ROOT\_PASSWORD: 'yes'  
 MYSQL\_USER: 'admin'  
 MYSQL\_PASSWORD: 'admin'  
 caches:  
 vendor:** vendor

База знаходиться в третій нормальній формі, вона приведена до другої нормальної форми і кожен не ключовий стовпець незалежний один від одного.  
Щоб привести базу до третьої нормальної формі, було виконано дії:

1. Визначили, в яких полях яких таблиць є взаємозалежність

2. Створено відповідні таблиці

3. Створено первинні ключі. Кожна таблиця має первинний ключ.

4. Створено необхідні зовнішні ключі, які утворюють одну з відносин

В побудові бази даних активно використовувалися mysql індекси.  
В якості движка для таблиць використовувався innodb, що дає можливість використовувати зовнішні ключі і транзакції

В силу обмежень mysql зовнішні ключі не використовувалися для зберігання зв'язків батько – дитина

Реалізація такого зв'язку даних вимагає зберігання у записи parent\_id, в якій повинен бути ідентифікатор батьківського записи з тієї ж таблиці. Це означає що потрібно створити зовнішній ключ з одного поля таблиці на інше поле тієї ж таблиці, але Mysql такий тип зв'язків не підтримує

основні таблиці:

-transaction

-account

-category

-tag

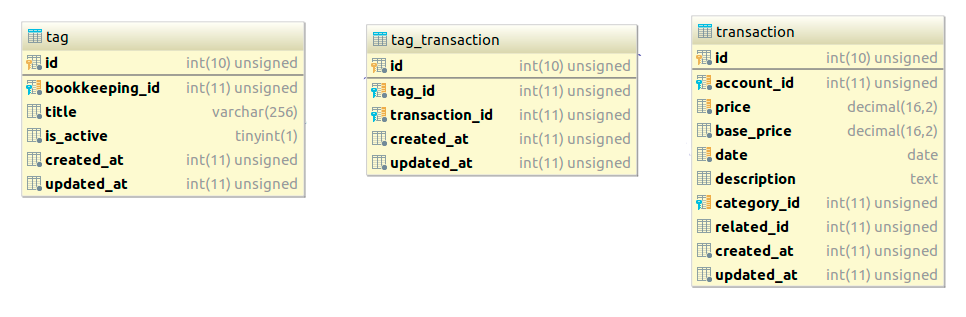
зв'язки:

transaction - account - Один до багатьох

transaction - category - один до багатьох

transaction - tag - багато до багатьох

Зв'язок багато-до-багатьох реалізована через проміжну таблицю  
також використовуються зв'язку типу один-до-одного, але їх реалізацію неможливо забезпечити на рівні mysql. Це забезпечується створенням унікального зовнішнього ключа в одній з таблиць. Приклад використання таблиці user – profile зображено на рисунку 1.17

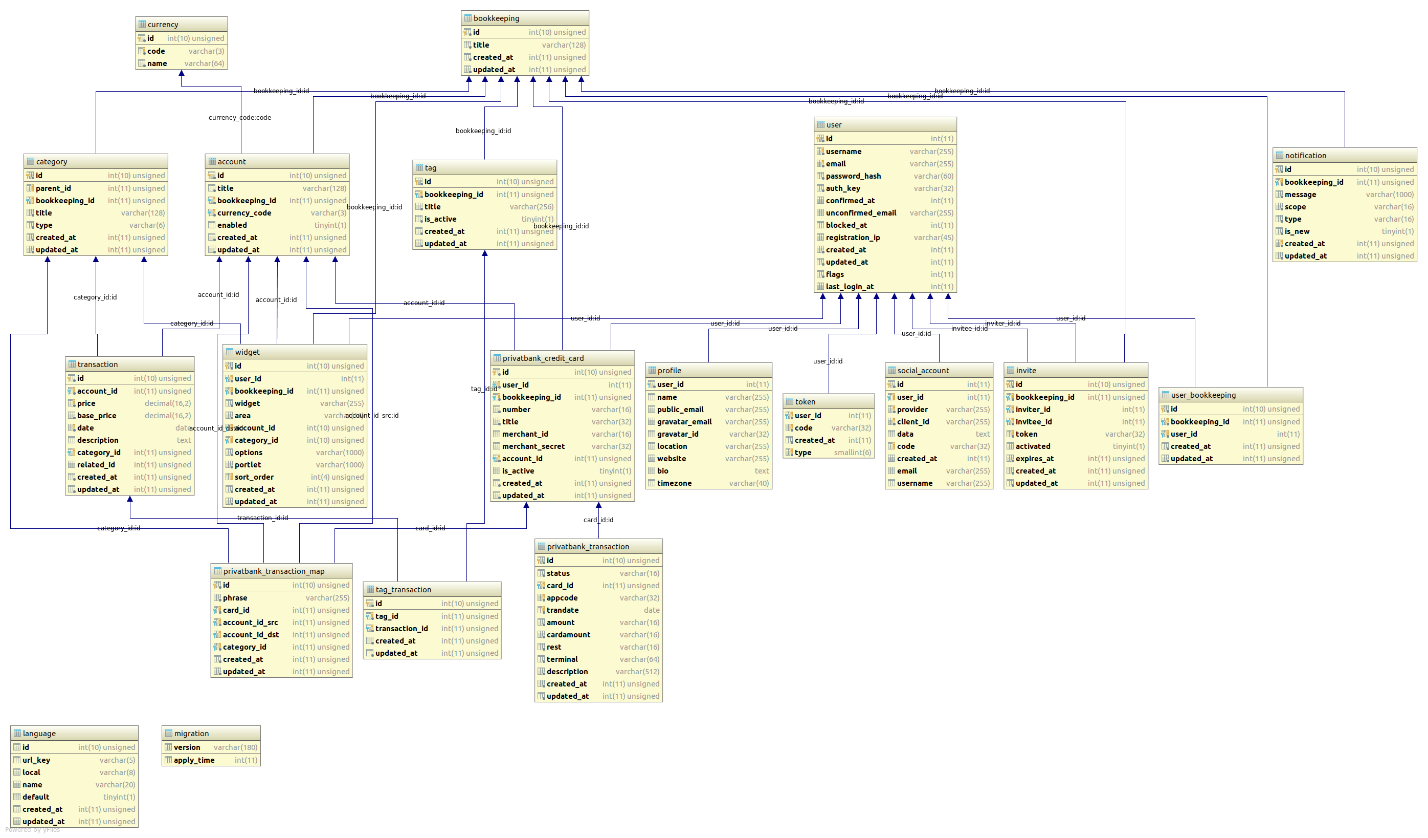
Рисунок 1.17 – Зв'язок transaction - tag - багато до багатьох

Повна структура бази даних наведена у додатку 1

ВИСНОВКИ

Переглянувши певну кількість подібних застосунків для ведення домашньої бухгалтерії були знайдені певні позитивні та негативні їх сторони. До позитивних можливо віднести те, що більшість застосунків має графічне відображення транзакцій та коректну роботу. До мінусів можна приписати те, що данні застосунки платні, занесення витрат можливе лише вручну. А також застосунки мають або лише версію для роботи з версією для персонального компютера, або лише мобільну версію, що не дає можливості використання застосунку в зручний для користувача час.

Дивлячись на усі позитивні та негативні сторони існуючих застосунків при виконанні роботи з них були узяті усі найкращі сторони та було додано певну кількість рішень для більш зручного користування.   
Для розробки застосунку використані сучасні технології.  
Для того щоб забезпечити постійний доступ до застосунку було виготовлено саме Веб-застосунок. Для автоматизації внесення витрат була зроблена синхронізація з api приватБанку. Для візуалізації грошових потоків до застосунку додано графіки різних видів. Для сумісного використання одного бухгалтерського обліку декількома людьми є можливість синхронізування аккаунтів.   
Застосунок можна легко розширювати, додаючи нові можливості, вдосконалювати його. За рахунок покриття коду тестами подальша розробка буде більш правильним, зменшиться кількість можливих помилок.   
Виконано мануальне тестування під час знаходження коду у стадії розробки та після завершення розробки на локальній версії, та все у бойовій версії після перенесення застосунку на сервіс amazon. Автоматизоване тестування використовувалось при виготовленні додатку кожного разу після залиття коду до сервісу github.



**4 ТЕХНІКО-ЕКОНОМІЧНЕ ОБҐРУНТУВАННЯ РОЗРОБКИ**

**4.1 Мета і призначення**

Тема: «РОЗРОБКА WEB-ДОДАТКУ ДЛЯ ВЕДЕННЯ ДОМАШНЬОЇ БУХГАТЕРІЇ З МОЖЛИВІСТЮ СИНХРОНИЗАЦІЇ АККАУНТІВ»

Мета роботи: розробити веб-додаток, завдяки якому користувачі зможуть вносити у додаток данні про витрати та надходження, генерувати графіки грошових потоків, синхронізувати профілі користувачів. Головними функціями додатку є наступні: вивід графіку транзакцій, синхронізувати кредитну картку з додатком для отримання виписок з рахунку, додавати та редагувати транзакції.

**4.2 Розробка бізнес-плану**

Бізнес-план – документ, котрий дозволяє обґрунтувати доцільність збуту продукції і розробки виробництва в умовах конкуренції.

Формування бізнес-плану включає розрахунки по визначенню місткості ринку, його сегментації по параметрах , конкурентах, споживачах, ціновому фактору. Складання кошторису витрат на розробку продукту, розрахунок договірної ціни й прибутку, його тиражування й збут.

На основі аналізу можна прогнозувати перспективи розвитку продукту та стійкості його позицій на ринку. За потреби – скорегувати розробку. Для кращого розуміння слід зробити порівняння відносно конкурентів та визначитися з якісними показниками продукту.

Необхідно зауважити, що бізнес-план є також важливим інструментом для залучення зовнішніх інвестицій. Тому що саме бізнес-план допомагає зрозуміти інвестору, на які ризики він піде і зіставити їх з можливою вигодою.

**4.2.1 Структура і загальні вимоги до змісту бізнес-плану**

Аналіз і підготовка вихідних даних.

Вихідними даними для оцінки доцільності розробки програмного продукту служать матеріали спеціальної частини дипломного проекту, а також дані, що містяться в планових і звітних документах підприємств, періодичній печаті, результатах аукціонів, біржових новинах, виставок і іншої комерційній інформації. Ці дані дозволяють порівняти функціональні характеристики розроблювального програмного продукту з характеристиками аналогів продуктів конкурентів.

**4.2.2 Опис характеристик продукту**

Призначення продукту – зберігати та аналізувати витрати та надходження користувача.

Додаток є багатовалютним, що дозволяє зберігати данні в різних валютах. Для підрахунку загальної суми використовуються дані з поточного курсу валютної пари.

Продукт представляє собою веб-додаток, доступ до якого можливий через мережу Інтернет. Для використання продукту необхідно створити обліковий запис на сайті продукту. Після створення облікового запису відкривається доступ до функцій продукту.

Основна область використання продукту – зберігання даних щодо грошових потоків. Невеликі підприємства та фізичні особи можуть використовувати додаток для домашньої бухгалтерії.

Основною характеристикою продукту є новизна сервісу. На поточний час існує багато способів та методів для зберігання даних, які включають в себе як аналітику, так і наукові моделі. Частіше всього такі моделі реалізовані у статистичних та математичних програмах. Сервіс придатний до використання для користувачів, котрі не мають досвіду в роботі з бухгалтерією.

**4.3 Дослідження й аналіз ринку збуту**

Ціль цього дослідження – визначення ємності сегмента і сегментація ринку.

Ємність товарного ринку – показник, що характеризує принципово можливий обсяг збуту товару. Місткість ринку визначається обсягом у фізичних одиницях або у вартісному вираженні й складається з її сегментів.

Сегмент ринку – виділена частина ринку, група споживачів і підприємства, що володіють певними загальними ознаками.

Цей вид дослідження дозволяє визначити можливий обсяг продажів продукту, виявити потенційних споживачів продукту, оцінити їх чисельність.

**4.3.1 Сегментація ринку по споживачам**

Сегментація ринку – один з найважливіших інструментів маркетингу. Від правильності вибору сегмента ринку багато в чому залежить успіх підприємства в конкурентній боротьбі.

Сегментація ринку по категоріях користувачів представлена нижче в таблиці 5.1.

Групи споживачів за сегментами:

І –фізичні особи;

ІІ – представники малого бізнесу;

Таблиця 4.1 – Сегментація ринку по споживачах

|  |  |  |  |
| --- | --- | --- | --- |
| Області використання (сегменти) | Код сегмента | Споживачі | |
|  | I | II |
| Фізичні особи | А | Х | Х |
| Малі підприємства | Б |  | X |

Після вибору основних груп споживачів необхідно обрати сегмент, на який орієнтований даний програмний продукт (таблиця 4.2).

Таблиця 4.2 – Аналіз ємності сегментів ринку збуту продукту

|  |  |  |  |
| --- | --- | --- | --- |
| Об'єкти використання продукту | Кількість об'єктів, що використо-вують виріб | Передбачуване число продажів одному об’єкту, шт | Передбачувана ємність сегментів |
| Фізичні особи | 200 | 1 | 200 |
| Малі підприємства | 10 | 1 | 10 |
| Разом |  |  | 210 |

Згідно з отриманих розрахунків ємність ринку становить 210.

**4.3.2 Параметрична сегментація ринку**

У цьому розділі необхідно встановити, які властивості програмного продукту найбільше приваблюють споживача. Особливо важливий даний вид сегментації для нових програмних продуктів.

Таблиця 4.3 Параметрична сегментація ринку

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Параметри продукту | Оцінка споживачів | | Підсумкова  оцінка | Питома вага, % |
| А | Б |
| 1. Ціна | 5 | 4 | 9 | 17.32 |
| 2. Надійність | 5 | 5 | 10 | 19.23 |
| 3. Швидкість роботи | 4 | 4 | 8 | 15.38 |
| 4. Сервісна підтримка | 2 | 4 | 6 | 11.54 |
| 5. Простота використання | 5 | 2 | 7 | 13.46 |
| 6. Зовнішній вигляд | 5 | 3 | 8 | 15.38 |
| 7. Використання сучасних наукових методів | 2 | 2 | 4 | 7.69 |
| Всього | | | | 100 |

Виходячи з отриманих розрахунків найбільш важливими параметрами на ринку є надійність та кількість вірних прогнозувань.

**4.3.3 Сегментація ринку по основним конкурентам**

Конкурентоспроможність - це здатність продукції відповідати вимогам конкурентного ринку, запитам покупців порівняно з іншими аналогічними товарами, представленими на ринку. Конкурентоспроможність продукту визначається не його абсолютною цінністю, а порівняльною корисністю його характеристик з аналогами по економічним та технічним параметрам.

Для оцінки конкурентоспроможності необхідно проаналізувати ринок та вибрати найбільш близькі по характеристикам вироби, які працюють на тому ж сегменту ринку.

Проаналізовано існуючі рішення додатків, котрі візуалізують грошові потоки. Визначені їх основні недоліки, до яких можна віднести застарілий дизайн, складність у внесенні даних користувачем, відсутність можливості синхронизації декількох акакунтів, неможливість автоматизованного внесення даних по надходженням та витратам з кредитної карти, відсутність інтеграції з соціальними мережами, відсутність можливості представлення інформації у вигляді графіків, неможливість зручно користуватися додатком з мобільних пристроїв . Що реалізовано у даному додатку завдяки чому додаток є більш конкуренто спроможним.

**4.4 Розрахунок собівартості та ціни програмного продукту**

Собівартість являє собою виражені в грошовій формі поточні витрати на виробництво й реалізацію програмного продукту. У ході виробничо-господарської діяльності ці витрати повинні відшкодовуватися за рахунок реалізації програмного продукту.

Використання показників собівартості в практиці, у всіх випадках вимагає забезпечення однаковості витрат, що враховують у її складі. Для забезпечення такої однаковості конкретний склад витрат, віднесений на собівартість регламентується типовим положенням з планування, обліку й калькулювання собівартості продукції (робіт, послуг) у промисловості (Постанова КМ 19.01.2007р. № 27/4248).

Метою обліку собівартості продукції є повне й достовірне визначення фактичних витрат, пов'язаних з розробкою, виробництвом і збутом продукції.

Витрати, що включають у собівартість продукції групуються по елементах:

1) витрати на оплату праці;

2) матеріальні витрати;

3) загальновиробничі витрати;

4) відрахування на соціальні заходи.

**4.4.1 Матеріальні витрати**

У склад елемента «матеріальні витрати» включають в себе витрати: на матеріали та сировину у виробничій діяльності підприємства. Під час виготовлення продукції (робіт, послуг) чи для господарських потреб, технічних цілей та сумісні дії у виробничому процесі.

Розрахунок виконуємо за формулою

, (5.1)

де Нpi –норма розходу i-го матеріалу на одиницю продукції;

Yi – ціна одиниці i-го матеріалу;

m – кількість видів матеріалів.

Розрахунки наведені у таблиці 4.4.

Таблиця 4.4 – Витрати на матеріали

|  |  |  |  |
| --- | --- | --- | --- |
| Перелік матеріалів, послуг і покупних виробів | Кількість | Ціна одиниці, грн. | Вартість, грн. |
| 1. Принтер | 1 | 1500,00 | 1500,00 |
| 2. Заправка картриджа для принтера | 1 | 150,00 | 150,00 |
| 3. Папір А4 | 1 уп. | 100,00 | 100,00 |
| Разом: |  |  | 1800,00 |

Аналогічно розраховуються витрати на закупівлю ліцензійного програмного забезпечення. Розрахунок вартості проводиться на підставі відомості про закупівлю ліцензійного програмного забезпечення.

Розрахунок вартості ліцензійного програмного забезпечення наведений у таблиці 4.6.

Таблиця 4.6 – Розрахунок вартості програмного забезпечення

|  |  |  |  |
| --- | --- | --- | --- |
| Найменування  комплектування | Кількість, шт. | Ціна за од., грн. | Сума, грн. |
| WebStorm | 1 | 2781 | 2871 |
| Домен | 1 | 345 | 345 |
| Хостинг | 1 | 600 | 600 |
| Разом | | | 3816 |

**4.4.2 Витрати на оплату праці**

До витрат на оплату праці ставляться основна й додаткова заробітна плата персоналу, зайнятого безпосередньо на виконанні конкретної теми: науково-технічний, науковці, науково-допоміжний персонал і виробничі робітники.

Розрахунок витрат на основну заробітну плату наведений у таблиці 4.7.

Таблиця 4.7 - Розрахунок витрат на основну заробітну плату

|  |  |  |  |
| --- | --- | --- | --- |
| Посада | Оклад, грн. | Кількість місяців | Сума, грн. |
| Керівник з розробки | 8000,00 | 2 | 16000,0 |
| Інженер - програміст | 7000,00 | 2 | 14000,0 |
| Разом | | | 30000,0 |

Додаткова заробітна плата

Вона включає доплати, надбавки, гарантійні й компенсаційні виплати, передбачені законодавством.

Додаткову заробітну плату приймаємо 10 % від Зосн.

Здоп = 0,1 ⋅ 30000,0 = 3000,0 грн.

Відрахування на соціальні заходи

До елемента «Відрахування на соціальні заходи» включаються:

1. відрахування на державне (обов'язкове) соціальне страхування, включаючи відрахування на обов'язкове медичне страхування, що становить 2,5 % від Зосн + Здоп;

0,025 ⋅ (Зосн + Здоп) = 0,025 ⋅ (30000,0 + 3000,0) = 825,0 грн.

1. відрахування на державне (обов'язкове) пенсійне страхування (у Пенсійний фонд) – 33,2 % від Зосн + Здоп;

0,332 ⋅ (Зосн + Здоп) = 0,332 ⋅ (30000,0 + 3000,0) = 10956,0 грн.

1. відрахування у Фонд сприяння зайнятості населення, що становить 2,5 % від Зосн + Здоп;

0,025 ⋅ (Зосн + Здоп) = 0,025 ⋅ (30000,0 + 3000,0) = 825,0 грн.

1. відрахування до Військового фонду 1,5 % від Зосн + Здоп.

0,015 ⋅ (Зосн + Здоп) = 0,015 ⋅ (30000,0 + 3000,0) = 495 грн.

**4.4.3 Загальновиробничі витрати**

До статті калькуляцій «Загальновиробничі витрати» ставляться витрати на повне відновлення й капітальний ремонт основного фонду (амортизаційні відрахування), витрати на обслуговування виробничого процесу, орендна плата, витрати на паливо, енергію й т.д.

У дипломній роботі загальновиробничі витрати приймаємо в розмірі 40 % від Зосн

Ззаг вир = Зосн ⋅ 0,4 = 30000,0 ⋅ 0,4 = 12000 грн.

**4.4.4 Розрахунок вартості машинного часу**

Вартість машинного часу визначається з розрахунку – 1 грн. за 1 годину. Всі роботи по створенню проекту велися 2 місяці, по 8 годин на день. Робочих днів у місяці - 22. Число виконавців - 2.

Вмч = 1 ⋅ 8 ⋅ 22 ⋅ 2 = 352,00 грн.

**4.4.5 Адміністративні витрати**

До статті «Адміністративні витрати» ставляться витрати; пов'язані із придбанням сировини, витрати на пожежну й сторожову охорону, матеріалів, забезпечення правил техніки безпеки праці й т.д.

У дипломній роботі адміністративні витрати приймаємо в розмірі – 10 % від Зосн

Задм = Зосн ⋅ 0,1 = 30000,0 ⋅ 0,1 = 3000,0 грн.

**4.4.6 Калькуляція собівартості**

Ціна розробки програмного продукту розраховується на основі калькуляції собівартості. Калькуляція собівартості пристрою представлена в таблиці 4.8.

Таблиця 4.8 - Калькуляція собівартості розробки продукту

|  |  |
| --- | --- |
| Найменування статей калькуляції | Сума, грн. |
| 1. Витрати на матеріали | 1800,0 |
| 2. Покупні комплектуючі вироби | 3816 |
| 3. Основна заробітна плата | 30000,0 |
| 4. Додаткова заробітна плата | 3000,0 |
| 5. Відрахування на соціальні заходи |  |
| Соціальне медичне страхування | 825,0 |
| Державне страхування (пенсійний фонд) | 10956,0 |
| Фонд зайнятості | 825,0 |
| Відрахування до Військового фонду | 495,0 |
| 8. Загальновиробничі витрати | 12000,0 |
| 9. Вартість машинного часу | 352,0 |
| 10. Виробнича собівартість | 64069,0 |
| 11. Адміністративні витрати | 3000,0 |
| 12. Витрати на збут (5 %) | 1500,0 |
| 13. Собівартість | 68569,0 |
| 14. Прибуток 25 % | 17142,25 |
| 15. Ціна розробки програмного продукту | 85711,25 |
| 16. ПДВ (20%) | 17142,18 |
| 17. Ціна продажу | 102853,5 |

Ціна розробки програмного продукту склала 85711,25грн. Прогнозована ціна продажу становить 102853,5грн.

**4.5 Оцінка ризику і страхування**

Метою аналізу проектних ризиків є надання оцінки всім видам ризиків проекту, а також визначення:

* можливих шляхів зниження ризиків;
* ступеня доцільності реалізації проекту за наявного рівня ризику та способів його зниження.

Аналіз проектних ризиків передбачає вирішення таких завдань:

* виявлення ризиків проекту;
* оцінка ризику проекту;
* визначення чинників, що впливають на ризик;
* пошук шляхів скорочення ризику;
* врахування ризику при оцінці доцільності реалізації проекту;
* врахування ризику при оцінці доцільності реалізації способу фінансування проекту.

При оцінці проектів передбачається, що всі вихідні величини, зокрема величини грошових потоків, відомі або можуть бути точно визначені. У реальній ситуації такого практично не буває. Параметри, які визначають величину грошових потоків, можуть набувати значень, які неабияк відхиляються від очікуваних.

Для розробленого продукту, можливі наступні види ризику:

* несанкціоноване копіювання як програмного забезпечення так і вихідного коду з метою подальшого використання;
* несанкціоноване втручання у зміст програмного продукту, з метою дискредитації, що приведе до падіння або повної відсутності продажів;
* несанкціоноване копіювання з метою продажів.

Зменшити ступінь ризику можна двома способами: самострахування і страхування за допомогою страхових компаній.

Самострахування передбачає забезпечення контролю за виробництвом і системою захисту інформації. Способами захисту інформації в даному випадку є:

* контроль за доступом до документації та ЕОМ;
* використання шифрування даних для захисту даних;
* заборона використання переносних носіїв інформації
* контроль за доступом до системи виробництва.

Для самострахування буде використано шифрування даних за допомогою безкоштовних програм. Також на самострахування буде використано 1% вартості розробки програмного продукту, тобто – 1028,5 грн. на місяць.

**4.6 Стимулювання продажу**

Стимулювання збуту – цілеспрямована діяльність підприємства по сприянню потенційним споживачам у виборі і придбанні продукції, що випускається ним або по створенню позитивної думки про неї.

Комплекс маркетингового стимулювання вміщує до себе крім рекламної діяльності наступні методи впливу на процес збуту виробів і послуг:

– персональний продаж;

– формування позитивної громадської думки про продукцію підприємства;

– економічне стимулювання збуту.

У випадку розроблюваного продукту найбільш підходять перший і останній методи.

Персональне (особисте) рекламування передбачає безпосередній контакт представника підприємства-розробника з споживачем продукції. У ході персонального продажу представник виробника допомагає споживачеві краще засвоїти переваги і вигоди виробу, що пропонується і переконує останнього придбати його або підтримати торгову марку виробника при спілкуванні з іншими споживачами.

Методами економічного стимулювання, що є доречними у випадку розроблюваного продукту є наступні:

* повна технічна підтримка протягом року;
* наявність знижок;
* можливість придбання ліцензії по оптовим цінам (для компаній).

**4.8 Фінансовій план**

Будь-яка сучасна компанія, яка веде економічну діяльність в тій чи іншій області бізнесу, займається плануванням. Планування в бізнесі грає якщо не провідну, то, по принаймні, важливу роль в питаннях економічної ефективності та направлено на максимізацію ефективності, яку здатний показати бізнес.

Фінансовий план підприємства - це підвид групи управлінських, взаємопов'язаних документів, який складається і ведеться для перспективного планування і операційного управління доступними фірмі ресурсами в грошовій формі. Простіше кажучи, завдяки фінансовим планам забезпечується баланс між плановими і фактичними надходженнями виручки, і з іншого боку - плановими і фактичними витратами на діяльність компанії.

Збалансованість фінансово-економічного стану компанії, яке досягається завдяки якісному фінансовому плануванню, мабуть, є головним прибутком використання такого інструменту менеджменту, як фінансовий план підприємства.

Табл. Карта руху готівки за 1-й рік реалізації

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Прибутки та витрати | | Сума, тис. грн. | | | | | |
|  | | І | ІІ | ІІІ | ІV | V | VI |
| Об’єм продажу | |  | 2 | 2 |  |  |  |
| Прибуток від реалізації | |  | 100 | 100 |  |  |  |
| постійні | Податки | 1,71 | 1,71 | 1,71 | 1,71 | 1,71 | 1,71 |
| Страхові внески | 1,02 | 1,02 | 1,02 | 1,02 | 1,02 | 1,02 |
| Витрати на рекламу | 0,6 |  |  |  |  | 0,6 |
| Усього | 3,33 | 2,73 | 2,73 | 2,73 | 2,73 | 3,33 |
| змінні | Податки |  | 32,9 | 32,9 |  |  |  |
| Витрати на адаптацію |  | 6,93 | 6,93 |  |  |  |
| Усього |  | 39,83 | 39,83 |  |  |  |
|  |  | -3,33 | 57,44 | 57,44 | -3,33 | -3,33 | -3,33 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Прибутки та витрати | | Сума, тис. грн. | | | | | | Усього |
|  | | VII | VIII | ІX | X | XI | XII |
| Об’єм продажу | | 2 | 2 |  | 2 |  |  | 10 |
| Прибуток від реалізації | | 100 | 100 |  | 100 |  |  | 500 |
| постійні | Податки | 1,71 | 1,71 | 1,71 | 1,71 | 1,71 | 1,71 |  |
| Страхові внески | 1,02 | 1,02 | 1,02 | 1,02 | 1,02 | 1,02 | 12,24 |
| Витрати на рекламу |  |  |  |  | 0,6 |  |  |
| Усього | 2,73 | 2,73 | 2,73 | 2,73 | 3,33 | 2,73 | 34,56 |
| змінні | Податки | 32,9 | 32,9 |  |  |  |  |  |
| Витрати на адаптацію | 6,93 | 6,93 |  |  |  |  |  |
| Усього | 39,83 | 39,83 |  |  |  |  |  |
|  |  | 57,44 | 57,44 | -3,33 | -3,33 | -3,33 | -3,33 | 79,66 |

Табл. Карта руху готівки за 2-й рік реалізації

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Прибутки та витрати | | Сума, тис. грн. | | | | Усього |
|  | | І кв. | ІІ кв. | ІІІ кв. | ІV кв. |
| Об’єм продажу | | 4 | 3 | 2 | 1 | 10 |
| Прибуток від реалізації | | 200 | 150 | 100 | 50 | 500 |
| постійні | Податки | 5,13 | 5,13 | 5,13 | 5,13 | 20,51 |
| Страхові внески | 3,06 | 3,06 | 3,06 | 3,06 | 12,24 |
| Витрати на рекламу | 1,8 |  | 1,8 |  | 3,6 |
| Усього | 9,99 | 8,19 | 9,99 | 8,19 | 36,36 |
| змінні | Податки | 65,8 | 49,35 | 32,9 | 16,45 | 180,95 |
| Витрати на адаптацію | 13,89 | 10,38 | 6,93 | 3,46 | 34,7 |
| Усього | 79,69 | 59,73 | 39,83 | 19,91 | 199,16 |
|  |  | 110,32 | 82,08 | 50,18 | 21,9 | 264,48 |

Табл. Карта руху готівки за 3-й рік реалізації

|  |  |  |
| --- | --- | --- |
| Прибутки та витрати | | Усього |
|  | |
| Об’єм продажу | | 8 |
| Прибуток від реалізації | | 400 |
| постійні | Податки | 20,51 |
| Страхові внески | 12,24 |
| Витрати на рекламу | 3,6 |
| Усього | 36,36 |
| змінні | Податки | 180,95 |
| Витрати на адаптацію | 34,7 |
| Усього | 199,16 |
|  |  | 264,48 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Прибутки та витрати | | Сума, тис. грн. | | | | Усього |
|  | | Початкові | 1рік. | 2рік. | 3 рік |
| Об’єм продажу | |  | 10 | 10 | 10 | 30 |
| Прибуток від реалізації | | 0 | 500 | 500 | 400 | 1400 |
| постійні | Зп виробника | 14 |  |  |  | 14 |
| Ціна матеріалів | 1,8 |  |  |  | 1,8 |
| Ціна машинного часу | 8,36 |  |  |  | 8,36 |
| Відрахування на соц страхування | 9,92 |  |  |  | 9,92 |
| Відрахування у фонд зайнятості | 1,6 |  |  |  | 1,6 |
| Накладні витрати | 18,78 |  |  |  | 18,78 |
| З\п реалізації та податки |  | 26,4 | 26,4 | 26,4 | 79,2 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| назва | | Сума, тис. грн. | | | | Усього |
|  | | Початкові значення | 1рік | 2 рік | 3рік |
| змінні | Страхові внески |  | 12,24 | 12,24 | 12,24 | 36,72 |
| Витрати на рекламу |  | 3,6 | 3,6 | 3,6 | 10,8 |
| Податки |  | 180,9 | 180,95 | 180,95 | 542,85 |
| Витрати на адаптацію |  | 34,7 | 34,7 | 34,7 | 104,1 |
| Усього | 54,46 | 257,84 | 257,84 | 257,84 | 854,38 |
|  |  | -54,46 | 242,16 | 242,16 | 242,16 | 672,02 |

**4.9 Висновки**

Проведений аналіз дозволяє зробити висновок про доцільність розробки та продажу «Програмного продукту для домашньої бухгалтерії».

Була розрахована ємність сегментів ринку виходячи з якої загальний передбачуваний обсяг продажів носить прогнозний характер і може бути розширений за рахунок модернізації та просування товару на нові сегменти  
ринку.

Згідно з проведеного аналізу конкурентоспроможності даний програмний продукт є вигідним для розробки, оскільки присутня новизна продукту.

Ціна розробки «Програмного продукту для домашньої бухгалтерії» склала 82296,25 грн. Прогнозована ціна продажу становить 100401,18 грн

6 ОХОРОНА ПРАЦІ ТА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1 Загальні положення з охорони праці

Охорона праці – це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров’я і працездатності людини у процесі трудової діяльності [1].

Завдання охорони праці – забезпечення безпечних, нешкідливих і сприятливих умов праці.

Охорона праці містить три основних складових частини: правові норми трудового законодавства, виробничу санітарію, гігієну та техніку безпеки, а також протипожежний захист і електробезпеку .  
 Мета охорони праці **-** забезпечення безпечних, нешкідливих і сприятливих умов праці через вирішення багатьох складних завдань.

## 6.2 Характеристика робочого приміщення

Дана дипломна робота розроблялася у робочому приміщенні, яке містить одне робоче місце та має розміри: довжина – 5 м, ширина – 2 м, висота – 3 м, загальна площа - 10 м² та об’єм - 30м³, що відповідає нормам НПАОП 0.00-1.28-10 [2]. На одне робоче місце, обладнане комп’ютером, відводиться S = 6 м2 , V = 20 м3. Кімната з робочим місцем знаходиться на п’ятому поверсі п’ятиповерхового будинку.

6.3 Виробнича санітарія

6.3.1 Аналіз небезпечних та шкідливих чинників

При роботі за комп’ютером на людину впливає низка небезпечних і шкідливих факторів згідно з ДСН 3.3.6.037-99 [3], які суттєво погіршують умови праці, знижують фізичну активність оператора та дуже сильно впливають на загальний стан здоров’я людини. Найбільшу небезпеку для людини, що працює за комп’ютером становлять фактори, які наведено в таблиці 6.1.

Таблиця 6.1 – Перелік небезпечних та шкідливі факторів.

|  |  |  |  |
| --- | --- | --- | --- |
| Найменування чинників | Джерела виникнення | Нормоване значення | Нормативні документі |
| 1 | 2 | 3 | 4 |
| Фізичні | | | |
| 1. Підвищений рівень шуму на робочому місці | Принтери, сканери, системні блоки | LA≤50 дБ(А) | ДСН 3.3.6.037-99. [3] |
| 2. Підвищена пульсація світлового потоку | Газорозрядні лампи | Кп= 5% | ДСанПіН 3.3.2-007-98. [8] |
| 3. Підвищена яскравість світла | Екран монітора | В=200кд/м2 | ДБН В. 2.5-28-2006 [15] |
| 4. Недостатня освітленість робочої зони | Недоліки системи освітлення | Еmin = 300 лк | ДБН В. 2.5-28-2006 [15] |
| 5. Пряме та відбите відблискування | Екран монітора | Р = 40 кд/м2 | ДСанПіН 3.3.2-007-98. [8] |
| 6. Недоліки природного освітлення | Невірне розташування монітора, віконних отворів, ламп денного освітлення | e = 1,08% | ДСанПіН 3.3.2-007-98. [8] |
| 7. Перевищення значення напруги в електричній мережі, замикання якої може трапитись крізь людське тіло | Мережа живлення | І = 0.3 мА  Uдог = 2 В | ДСанПіН 3.3.2-007-98. [8] |
| 8. Підвищений рівень іонізуючих випромінювань у робочій зоні | Рентгенівське випромінювання монітора | Кількість пози-тивних іонів  Ф=1500÷3000, негативних іонів  Ф=3000÷5000 | ДСанПіН 2.2.4-1294-03 [7] |
| 9. Підвищенна або знижена температура повітря робочої зони | Нераціональна організація системи вентиляції та опалення | Температура в холодний період 22÷24,а в теплий 23÷25 | ДСН 3.3.6. 042-99 [7] |
| 10. Підвищена або знижена рухомість повітря | Нераціональна організація системи вентиляції та опалення | Швидкість руху повітря  υ ≤ 0,1 м/с | ДСН 3.3.6. 042-99 [7] |
| 11. Підвищена або знижена вологість повітря | Нераціональна організація системи вентиляції та опалення | Відносна вологість 40÷60% | ДСН 3.3.6. 042-99 [7] |
| 12.Підвищений рівеньелектромагнітного випромінювання | ЕПТ монітора, електроапаратура |  | НПАОП 0.00–1.28–10. [2] |
| 13. Підвищена напруженість електростатичного поля | Поверхня обладнання, речовин та джерела живлення | Е ≤ 20 кВ/м | ДСанПіН 3.3.2-007-98. [8]  ДБН В.2.5-67:2013. [18] |
| Хімічні | | | |
| 14. Виробничий пил | Статична електрика, накопичена на поверхні комп'ютера. Нерегулярне прибирання | ГДК= 4мг/м3 | ДСН 3.3.6.042-99 [16] |
| Психофізіологічні | | | |
| 15. Нервово-психічні перенавантаження: монотонність праці | Введення даних з клавіатури | 40 тис. знаків | НПАОП 0.00–1.28–10. [2] |
| 16. Розумове перенапруження | Обробка великої кількості інформації | Зниження витривалості до вихідного | НПАОП 0.00–1.28–10. [2] |
| 17. Перенапруження аналізаторів: зір | Робота з дисплеєм | Зниження реакції користувача на звук та світло на 40÷50% | НПАОП 0.00–1.28–10. [2] |
| 18. Фізичні перенавантаження статичної дії | Робоча поза | Зниження статичної витривалості на 10% | НПАОП 0.00–1.28–10. [2] |

5.3.2 Мікроклімат робочої зони

Робота дослідника відноситься до категорії Iа легка та не потребує фізичних напружень, але оскільки робота відноситься до категорії нервово-напруженої праці, то умови мікроклімату у приміщенні повинні відповідати оптимальним значенням згідно з ДСН 3.3.6.042-99 [8]

Таблиця 6.2 – Оптимальні параметри мікроклімату

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Категорія робіт | Пора року | Температура  t, °С | Відносна вологість ϕ, % | Швидкість руху повітря υ, м/с |
| Легка Iа | Холодна | 22÷24 | 40÷60 | ≤0,1 |
| Легка Iа | Тепла | 23÷25 | 40÷60 | ≤0,1 |

Для забезпечення комфортності у зимовий період року діє загальна система опалення , що забезпечує достатнє постійне й рівномірне нагрівання повітря в приміщенні в холодний період року, а у літній період – вентиляція та кондиціонування згідно ДБН В.2.5-67:2013 [9].

5.3.3 Виробниче освітлення

Працездатність оператора у більшості залежить від освітлення. Незадовільне освітлення кількісно або якісно втомлює не тільки зір, але і викликає втому організму в цілому, оказує вплив на продуктивність праці оператора.

Для забезпечення нормального освітлення застосовується природне та штучне освітлення, а також змішане, які нормуються санітарними нормами і правилами ДБН В.2.5-28-2006 [5].

Природне освітлення в приміщеннях може здійснюватися через вікна, орієнтовані в північно-східному напрямку (м. Харків знаходиться у III світло-кліматичному районі), і забезпечувати коефіцієнт природного освітлення згідно ДБН В.2.5-28-2006 [5].

Природне освітлення нормується коефіцієнтом природної освітленості (КПО). Нормовані значення КПО для будівель визначаються за такою формулою:

*= eн \**  (5.1)

де : eн = 1,2% - коефіцієнт природної освітленості при боковому освітленні для III розряду зорової роботи (згідно з таблицями 1,2 ДБН В.2.5-28-2006 [5]),

= 0,9 - коефіцієнт світлового клімату

N – номер групи забезпеченості природним світлом приміщення (орієнтація світлових прорізів на північний захід)

Таким чином нормоване значення КПО в даному випадку буде складати:

=1,2\*0,9 = 1,08% (5.2)

На вікнах повинні бути передбачені сонцезахисні пристрої (жалюзі або штори) для захисту від прямих сонячних променів. При роботі по ПЕОМ використовують бокове одностороннє природне освітлення.

Штучне освітлення приміщень може бути оснащено системою взаємного рівномірного освітлення. На робочих місцях застосовується загальне штучне освітлення, в денний час – комбіноване.

Для створення комфортних умов зорової роботи середньої точності необхідні дані по нормам освітлення, які приведені в таблиці 6.3

Таблиця 6.3– Характеристика виробничого освітлення

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Характеристика зорової праці | Мінімальний розмір  об'єкта розрізнення | Фон | Контраст | Розряд, підрозділ зорової праці | Нормоване значення характеристик освітлення | | |
| Суміщене освітлення | Штучне освітлення Еmin, лк | |
| *eN*, % | комбіноване | загальне |
| Висока точність | від 0,3 до 0,5 мм | світлий | середній | ІІІ,г | 1,08 | 400 | 200 |

6.3.4 Шум і вібрація

Вібрація - це механічний коливальний рух системи з пружними зв’язками. Дотримання нормованих значень вібрації ДСН 3.3.6.042-99 [8] та шуму досягається завдяки використанню якісного обладнання, що виготовлене з використанням новітніх звукопоглинаючих та віброізолюючих матеріалів.

Шум – це набір звуків різної інтенсивності і частоти, що знаходяться в хаотичному, безладному поєднанні. Шум та вібрацію в машинному залі створюють пристрої для друку та вентиляція системних блоків ПЕОМ, кондиціонер.

Швидкість поширення звукових хвиль при нормальному атмосферному тиску і температурі 200С складає: у повітрі – 344 м/с; у воді – 1500 м/с; в тканинах тіла людини – 1500-1600 м/с.

У робочому приміщенні причиною шуму є апарати, прилади, й обладнання (друкувальний пристрій, комп’ютери, кондиціонер і т.д.).

Рівень тиску звуку в приміщенні, де працює обслуговуючий персонал, не повинен перевищувати 50 дБ(А), згідно з ДСН 3.3.6.037-99\*[3].

Рівень шуму в кабінеті становить 40 дБ(А), що не перевищує норму.

6.4 Допустимий рівень іонізації повітря приміщення

Іонний склад повітря в машинному залі може значно змінюватись під дією випромінювання від моніторів. Для підтримки необхідної концентрації позитивних та негативних іонів у повітрі робочих зон застосовують наступні заходи та обладнання: зволожувачі кондиціонери, примусову вентиляцію (провітрювання, системи загально обмінної припливно-витяжної вентиляції, пристрої місцевої вентиляції), захисні екрани, що заземлені.

Рівні іонізації повітря приміщення згідно з ДСН 3.3.6.042-99. [7] наведе в таблиці 6.4.

Таблиця 6.4 – Оптимальні рівні іонізації повітря в приміщені з ПЕОМ

|  |  |  |
| --- | --- | --- |
| Рівні | Кількість іонів в 1 см3 повітря | |
|  |  |
| Оптимальні | 1500÷3000 | 3000÷5000 |

6.5 Допустимі параметри електромагнітного неіонізуючого випромінювання та електростатичного поля

Електромагнітні поля високої інтенсивності призводять до перегріву тканин та впливають на органи зору. Електромагнітні поля помірної інтенсивності викликають порушення діяльності центральної нервової та серцево-судинної систем, порушення біологічних процесів у тканинах та клітинах. Електромагнітні поля малої інтенсивності – підвищення стомлюваності, головні болі, випадіння волосся.

Гранична напруга електростатичного поля становить 10кВ/м, що відповідає вимогам ДСанПіН 3.3.2-007-98. [8] та не перевищує значения Е < 20 кВ/м.

Для захисту від впливу електромагнітних полів застосовуються наступні заходи: захист часом (обмеження часу перебування в зоні джерела електромагнітного поля); захист збільшенням відстані (60-80 см віл екрана); метод екранування робочого місця або джерела випромінювання електромагнітного поля. Допустимі рівні неіонізуючого та іонізуючого випромінювання електромагнітного та електростатичного полів на робочих місцях згідно з ДСанПіН 3.3.2-007-98 [8] наведені в таблиці 6.5.

Таблиця 6.5 – Допустимі параметри електромагнітних неіонізуючих випромінювань та електричного поля

|  |  |  |  |
| --- | --- | --- | --- |
| Види поля | Допустимі параметри поля | | Припустима поверхнева щільність потоку енергії (інтенсивність) Вт/ |
| Електрична складова Е, В/м | Магнітна складова  Н, А/м |
| 1 | 2 | 3 | 4 |
| Напруженість ЕМП  60 кГц ÷ 3 МГц | 50 | 5 |  |
| 3 МГц ÷30 МГц | 20 | - |  |
| 30 МГц ÷ 50Мгц | 10 | 0,3 |  |
| 30 МГц ÷ 300 МГц | 5 | - |  |
| 300 МГц ÷ Гц | - | - | 10 |
| Електромагнітне поле оптичного діапазону в ультрафіолетовому спектрі  УФ - С (220 ÷ 280 нм)  ВФ - В (280÷320 нм)  УФ - А (320 ÷ 400 нм) |  |  | 0,001  0,01  10,0 |
| У видимій частині спектру:  400-700 мм |  |  | 10,0 |
| В інфрачервоній частині спектру: 0,76 – 10,0 мкм |  |  | 35,0 – 70,0 |
| Напруженість електричного поля | 20кВ/м |  |  |

6.6 Електробезпека

Джерелом живлення комп’ютерів є однофазна трипроводова мережа: фазний провід ,нульовий робочий провід, нульовий захисний провід, напругою 220 В, частотою 50 Гц, потужністю 3 кВт.

Машинний зал за небезпекою ураження електричним струмом відноситься до приміщень з підвищеною небезпекою, оскільки можливе одночасне доторкання людини до металевих конструкцій споруд, що мають з’єднання з одного боку з землею, а з іншого – з металевим корпусом електрообладнання відповідно до ПУЕ-14 [9].

В приміщенні обчислювального центру прокладена шина повторного заземлення ( провідник, яким заземлюється), яка відповідає вимогам ДБН В.2.5-56- 2014 [19] та з’єднується із заземленою нейтраллю електроустановки. Електробезпека електричних приладів забезпечується комплексом конструктивних, схемно-конструктивних та експлуатаційних засобів та засобів захисту.

Конструктивні заходи електробезпеки запобігають можливості дотику людини до струмопровідних частин. Всі вимірювальні прилади виконані в захисних корпусах. Розкриття кришок корпусів слід проводити тільки після від’єднання приладу від мережі живлення. Згідно з ПУЕ-14 [9] ступінь захисту оболонок та корпусів апаратури прийнятий не нижче ІР-44, де перший знак «4» – захист від твердих тіл розміром більше 1 мм, другий знак «4» – захист від бризок.

Схемна-конструктивні заходи електробезпеки забезпечують безпеку дотику людину до металевих та струмопровідних частин електричних апаратів при випадковому ушкодженні їхньої ізоляції та виникнення електричного потенціалу на них.

Зважаючи на той факт, що напруга менше 3000В, але більше 1000В, то згідно з «Правила улаштування електроустановок» [21] з метою захисту від ураження електричним струмом застосовується заземлення з заземленою нейтраллю.

Згідно з НАПБ Б.03.001-2001 [20] приймаємо І клас захисту від ураження електричним струмом персоналу, тому що комп’ютер має робочу ізоляцію та елементі заземлення, що забезпечують безпеку дотику людини до металевих струмопровідних частин електроприладів при випадковому пробої їхньої ізоляції та виникнення електричного потенціалу на них.

До основних експлуатаційно-технічних заходів електробезпеки відносять: захисне заземлення: занулення; вирівнювання потенціалів; малу напругу; електричне розподілення мереж (за допомогою розділових трансформаторів); захисне вимкнення; ізоляція струмопровідних частин (робоча, додаткова, підсилена. подвійна); компенсація струмів замикання на землю; огороджувальні пристрої; електрична попереджувальна сигналізація; блокіровки. Допоміжними технічними засобами, що використовуються при обслуговуванні та експлуатації ЕУ, вважають плакати – знаки безпеки/небезпеки; засоби індивідуального захисту, запобіжне улаштування, дерев'яні решітки, гумові килими.

## 6.7 Індивідуальне завдання

Індивідуальним завданням є розрахунок захисного заземлення за індивідуальним варіантом.

Таблиця 6.6 – Вхідні дані для розрахунку захисного заземлення

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Трансформаторна підстанція напругою U, кВ | Розміри будівлі | | Розрахунковий опір природного заземлювача, Rе, Ом | Протяжність лінії електропередач | | Параметри вертикаль-ного електрода | | Параметри горизонталь-ного електрода | Питомий опір землі ρ виміряний,  Ом ⋅ м | Кліматична зона |
| L, м | В, м | ***l*К.Л.**,  км | ***l*В.Л.**, км | ***l*В,** м | d, мм | Перетин смуги, мм2 |
| 6 / 0.4 | 18 | 12 | 12 | 99 | 86 | 5 | 14 | 8 х 25 | 160 | I |

Захисне заземлення - навмисне електричне з'єднання з землею або її еквівалентом металевих не струмоведучих частин, які можуть опинитися під напругою внаслідок замикання на корпус і з інших причин (індуктивний вплив сусідніх струмоведучих частин, розряд блискавки тощо). Відповідно до вимог [24] захисне заземлення електроустановки слід виконувати:

* при номінальній напрузі 380В і вище змінного струму і 440В і вище постійного струму в усіх випадках;
* при номінальних напругах від 42В до 380В змінного і 110В до 440В по-стійного струму при роботах в умовах з підвищеною небезпекою, особливо не-безпечних і зовнішніх установках.

Заземлення здійснюється за допомогою спеціальних пристроїв - заземлювачів. Заземлювачі бувають поодинокі і групові. Груповий заземлювач складається з вертикальних стержнів і з'єднує їх горизонтальна смуга. Вертикальні електроди закладають разом з фундаментом будівель на певній відстані один від одного. З метою економії коштів ПУЕ рекомендує використовувати природні заземлювачі.

У нашому випадку заземлювальний пристрій використовується для електроустановки напругою понад 1000В, тому розрахункове значення струму замикання на землю може бути визначено за такою напівемпіричною формулою:

 (5.2)

де *U*л – лінійна напруга мережі (на високій стороні трансформаторної підстанції), кВ;

*l*к, *l*в – довжина електрично зв’язаних відповідно кабельних и повітряних ліній, км.

Таким чином,

 (5.3)

Відповідно отриманому розрахунковому значенню струму замикання на землю, нормативне значення опору заземлюючого пристрою (ЗП) *R*з находимо за формулою:

*R*з *=* 125 / *I*з,(5.4)

*R*з *=* 125 / 60,86 *=* 2,05 Ом*.*

При використанні природних заземлювачів необхідний опір штучного заземлювача *R*и визначається за формулою:

 (5.5) де *R*е – опір розтікання струму природних заземлювачів, Ом;

*R*и – необхідний опір штучного заземлювача, Ом;

*R*з – розрахунковий нормований опір ЗП, Ом;



Оскільки в даному завданні природний заземлювач не використовується, даний розрахунок не виконується.

Визначаємо розрахунковий питомий опір землі за формулою:

*ρ* = *ρ*изм · ψ, (5.6)

де *ρ* – розрахунково питомий опір землі, Ом·м;

*ρ*изм – питомий опір землі, отриманий в результаті вимірів,

Ом·м (наданий у вхідних даних завдання);

ψ – коефіцієнт сезонності, що враховує промерзання або висихання ґрунту. Для кліматичного поясу I для землі із малою вологістю та електродом довжиною 3м Ψ = 1,3. Таким чином,

*ρ* = 160 · 1,3 = 208 Ом⋅м.

Обчислюємо опір розтікання струму одиночного вертикального заземлювача *R*в. У разі стержневого круглого перетину (трубчастого) заземлювача, заглибленого в землю, розрахункова формула має вигляд:



(5.7)

де *ρ*в– розрахунковий питомий опір ґрунту, обчислений за формулою (5), Ом·м,;

*l* –довжина вертикального стержня, м;

*d* –діаметр перетину, мм;

*t* –відстань від поверхні ґрунту до середини довжини вертикального стержня, м;





Розрахуємо наближену кількість вертикальних стержнів:



де *R*в – опір розтікання струму одиночного вертикального заземлювача, обчислений за формулою (6), Ом;

*R*и – необхідний опір штучного заземлювача, обчислений за формулою (4), Ом;



Отримане число стержнів округляємо до найближчого більшого довідкового значення. Відповідно, n = 20.

Визначаємо конфігурацію групового заземлювача (контур) з урахуванням можливості його розміщення на відведеній території та відповідну довжину горизонтальної смуги:

*l*г = 1,05·*а*·*п*, (5.8)

де *а* – відстань між вертикальними стержнями, м;

*п* – кількість вертикальних стержнів;

*а* = *k* · *l*в, (5.9)

де *k* – коефіцієнт кратності, що дорівнює 1, 2, 3;

*l*в – довжина вертикального стержня, м.

Коефіцієнт кратності приймемо рівним 2.

*а* = 2 ·5 = 10м (5.10)

*l*г = 1,05·10·20 = 210 м. (5.11)

Периметр будівлі = 2 · (18 + 12) = 60 м.

Обчислюємо опір розтікання струму горизонтального стержня *R*г. У разі горизонтального смугового заземлювача розрахунок виконується за формулою:

 (5.12)

где *ρ* – розрахунково питомий опір грунту, Ом·м;

*l* – довжина горизонтальної смуги, м;

*b* –ширина смуги, м;

*t* –відстань від поверхні ґрунту до середини ширини горизонтальної смуги, м;





Оберемо коефіцієнти використання вертикальних стержнів (ηв) і горизонтальної смуги (ηг) з урахуванням числа вертикальних стержнів (*п*) і відносини відстані між стержнями (*а*) до їх довжини (*l*в).

; ηг = 0,29; ηв = 0,58.

Розрахуємо еквівалентний опір розтікання струму групового заземлювача:

, (5.13)

де *R*в, *R*г  – відповідно опір вертикального стержня і горизонтальної смуги, обчислені за формулами (6) і (11) відповідно, Ом;

ηв, ηг  – відповідно коефіцієнти використання вертикальних стержнів і горизонтальної смуги, Ом;

n – кількість вертикальних стержнів.

 (5.14)

Отриманий опір розтікання струму групового заземлювача на повинен перевищувати необхідний опір

*R*гр ≤ *R*и (5.15)

3,5 > 2,47

Отже, отриманий опір задовольняє необхідним умовам (5.15).

Отримані параметри наведені у таблиці 7.7.

Таблиця 6.7 – Результати розрахунку захисного заземлення.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *ρ*,  Ом·м | *l*в,  м | *K* | *n*,  шт | *l*г,  м | *η*в | *η*г | *R*в, Ом | *R*г,Ом | *R*гр,Ом | *R*и,Ом |
| 208 | 5 | 2 | 40 | 420 | 0,58 | 0,29 | 46,17 | 12,43 | 3,5 | 2,47 |

6.8 Пожежна безпека

Причинами виникнення пожежі в робочому залі можуть бути: несправність електропроводки, коротке замикання електричних ланцюгів, перегрівання апаратури, блискавка. З точки зору пожежної безпеки відповідно до ПУЕ-14 [9] клас зони приміщення ІІ-ІІа (приміщення, в якому горючі речовини знаходяться у твердому або волокнистому стані без виділення пилу та волокон).

Категорія приміщення згідно і ДСТУ Б В.1.1-36:2016. [11] га вибуховою, пожежно-вибуховою та пожежною небезпекою відноситься до категорії В, оскільки в приміщеннях наявні тверді горючі матеріали, здатні при взаємодії з киснем повітря або один з одним тільки горіти.

Ступінь захисту вогнестійкості будівлі відноситься до II згідно ДБН В.1.1-7-2002. [12], оскільки будівля відноситься до будинків і несучими та огороджувальними конструкціями з цегли та залізобетону.

Для даного класу будівель І місцевості із середньою грозовою діяльністю 10 і більше грозових годин на рік, тобто для умов міста Харкова, встановлено III рівень блискавок захисту відповідно до ДСТУ Б В.2.5. – 38:2008. [13].

Пожежна безпека згідно з ДБН В.1.2-7-2008. [10] забезпечується системами запобігання пожежі, пожежним захистом та організаційно-технічними заходами. У системі пожежного захисту передбачені аварійне виключення і перемикання апаратури, система сповіщення – звукова сигналізація. Пожежна безпека людини забезпечується використанням вуглекислого вогнегасника ВВК-5 ємкістю 5 літрів відповідно і гасіння вогню малої електропровідності. Застосування пінних вогнегасників включено, тому що ПЕОМ може перебувати під напругою. Машинний зал відповідає усім вимогам пожежної безпеки. Приміщення обладнане системою автоматичного сповішення про пожежу.

6.9 Охорона навколишнього природного середовища

Охорона навколишнього середовища регламентується законом України «Про охорону навколишнього природного середовища» [14].

## Усе необхідне для життєдіяльності людина отримує з природи: повітря, воду, сировину для промисловості. Вплив людини на навколишнє середовище є перетворюючим, що змінює її, причому далеко не завжди в кращу сторону, тому збереження природного середовища і розумна охорона природи - одна з най гостріших проблем, що стоять перед людством, особливо в сучасних умовах.

Під навколишнім середовищем розуміють цілісну систему взаємопов’язаних природних і антропогенних об’єктів і явищ, під впливом і при безпосередньому використанні яких відбувається праця, побутова діяльність, відпочинок людей. Складовою частиною навколишнього середовища є природне середовище. Перед сучасним суспільством стоїть завдання не тільки зберегти природу, а й запобігти негативним наслідкам господарської діяльності людини в майбутньому.

Охорона навколишнього середовища являє собою важку комплексну проблему, яка має відношення до всього суспільства в цілому і до кожного окремого громадянина.

Основною ідеєю охорони навколишнього середовища є використання та відтворення природних ресурсів, забезпечення екологічної безпеки, попередження та ліквідація негативного впливу будь-якої діяльності на навколишнє середовище. Розвиток комп’ютерної техніки викликав забруднення природного середовища електромагнітним випромінюванням. Його дія на людину перш за все позначається на нервовій системі.

При виконанні дипломної роботи основними джерелами забруднення в приміщенні є: відпрацьоване канцелярське приладдя, відходи паперу, побутове сміття.

Усі відходи утилізуються відповідним чином.

6. ЦИВІЛЬНИЙ ЗАХИСТ

Цивільний захист – це функція держави, спрямована на захист населення, територій, навколишнього природного середовища та майна від надзвичайних ситуацій шляхом запобігання таким ситуаціям, ліквідації їх наслідків і надання допомоги постраждалим у мирний час та в особливий період [52].

У данному розділі диплому розглядається тема: «Повноваження інших центральних органів виконавчої влади у сфері цивільного захисту. Права та обов’язки громадян у сфері цивільного захисту».

6.1 Повноваження інших центральних органів виконавчої влади у сфері цивільного захисту [52].

До повноважень інших центральних органів виконавчої влади у сфері цивільного захисту належить:

1) забезпечення цивільного захисту у сфері суспільного життя, в якій реалізує державну політику відповідний орган виконавчої влади; 2) здійснення заходів щодо захисту населення і територій під час надзвичайних ситуацій; 3) забезпечення виконання завдань цивільного захисту створеними ними функціональними підсистемами; 4) розроблення та забезпечення реалізації галузевих програм і планів заходів з питань цивільного захисту, зокрема спрямованих на захист населення і територій від надзвичайних ситуацій та запобігання їх виникненню, забезпечення техногенної та пожежної безпеки; 5) розроблення та здійснення заходів, спрямованих на забезпечення сталого функціонування національної економіки в особливий період, зокрема суб’єктів господарювання, що належать до сфери управління центральних органів виконавчої влади; 6) забезпечення реалізації вимог техногенної безпеки на потенційно небезпечних об’єктах, об’єктах підвищеної небезпеки та інших суб’єктах господарювання, що належать до сфери їх управління, які можуть створити загрозу виникнення аварії; 7) керівництво створеними ними аварійно-рятувальними службами, суб’єктами господарювання, основна діяльність яких спрямована або може бути спрямована на виконання завдань із запобігання та ліквідації наслідків надзвичайних ситуацій, спеціалізованими службами цивільного захисту, забезпечення їх діяльності та здійснення контролю за готовністю до дій за призначенням; 8) забезпечення виконання аварійно-рятувальних та інших невідкладних робіт, заходів і робіт з ліквідації наслідків надзвичайних ситуацій; 9) визначення за погодженням з центральним органом виконавчої влади, який забезпечує формування та реалізує державну політику у сфері цивільного захисту, місцевими державними адміністраціями та органами місцевого самоврядування загальної потреби у захисних спорудах цивільного захисту для суб’єктів господарювання, що належать до сфери їх управління; 10) організація здійснення заходів щодо створення, утримання та використання фонду захисних споруд цивільного захисту суб’єктів господарювання, що належать до сфери їх управління; 11) організація обліку фонду захисних споруд цивільного захисту суб’єктів господарювання, що належать до сфери їх управління, та захисних споруд цивільного захисту державної власності, що перебувають на балансі суб’єктів господарювання приватної форми власності; 12) організація проведення технічної інвентаризації фонду захисних споруд цивільного захисту суб’єктів господарювання, що належать до сфери їх управління; 13) виключення за погодженням з центральним органом виконавчої влади, який забезпечує формування та реалізує державну політику у сфері цивільного захисту, сховищ та протирадіаційних укриттів з фонду захисних споруд цивільного захисту; 14) віднесення суб’єктів господарювання, що належать до сфери їх управління, до категорій цивільного захисту відповідно до основних показників та затвердження їх переліку; 15) організація створення автоматизованих систем раннього виявлення загрози виникнення надзвичайних ситуацій та оповіщення населення у разі їх виникнення на об’єктах підвищеної небезпеки, що належать до сфери їх управління; 16) здійснення методичного керівництва суб’єктами господарювання, що належать до сфери їх управління, стосовно виконання ними вимог техногенної та пожежної безпеки, а також контролю за виконанням зазначених вимог; 17) створення і використання матеріальних резервів для запобігання та ліквідації наслідків надзвичайних ситуацій; 18) забезпечення навчання з питань цивільного захисту, техногенної та пожежної безпеки посадових осіб міністерств та інших центральних органів виконавчої влади, а також суб’єктів господарювання, що належать до сфери їх управління; 19) розроблення та здійснення комплексу заходів, спрямованих на поліпшення пожежної безпеки суб’єктів господарювання, що належать до сфери їх управління; 20) створення та ведення Державного реєстру потенційно небезпечних об’єктів; 21) здійснення інших повноважень у сфері цивільного захисту, передбачених цим Кодексом та іншими законодавчими актами.

Організація заходів цивільного захисту у функціональних підсистемах і забезпечення техногенної та пожежної безпеки у сфері суспільного життя, в якій реалізує державну політику міністерство чи інший центральний орган виконавчої влади, здійснюються підрозділами з питань цивільного захисту, які створюються такими органами у складі їх апаратів. Положення про підрозділи з питань цивільного захисту центрального органу виконавчої влади затверджується відповідним центральним органом виконавчої влади. У центральних органах виконавчої влади, які не створюють функціональні підсистеми, призначаються працівники з питань цивільного захисту, техногенної та пожежної безпеки. Основні функції з питань цивільного захисту, що належать до компетенції відповідного центрального органу виконавчої влади, визначаються в положенні про цей орган.

6 .2 Права та обов’язки громадян у сфері цивільного захисту [52].

1. Громадяни України мають право на:

1) отримання інформації про надзвичайні ситуації або небезпечні події, що виникли або можуть виникнути, у тому числі в доступній для осіб з вадами зору та слуху формі;

2) забезпечення засобами колективного та індивідуального захисту та їх використання;

3) звернення до органів державної влади та органів місцевого самоврядування з питань захисту від надзвичайних ситуацій;

4) участь у роботах із запобігання та ліквідації наслідків надзвичайних ситуацій у складі добровільних формувань цивільного захисту;

5) отримання заробітної плати за роботу з ліквідації наслідків надзвичайної ситуації у разі залучення до таких робіт згідно з трудовими договорами;

6) соціальний захист та відшкодування відповідно до законодавства шкоди, заподіяної їхньому життю, здоров’ю та майну внаслідок надзвичайних ситуацій або проведення робіт із запобігання та ліквідації наслідків;

7) медичну допомогу, соціально-психологічну підтримку та медико-психологічну реабілітацію у разі отримання фізичних і психологічних травм.

2. Громадяни України зобов’язані:

1) дотримуватися правил поведінки, безпеки та дій у надзвичайних ситуаціях;

2) дотримуватися заходів безпеки у побуті та повсякденній трудовій діяльності, не допускати порушень виробничої і технологічної дисципліни, вимог екологічної безпеки, охорони праці, що можуть призвести до надзвичайної ситуації;

3) вивчати способи захисту від надзвичайних ситуацій та дій у разі їх виникнення, надання домедичної допомоги постраждалим, правила користування засобами захисту;

4) повідомляти службі екстреної допомоги населенню про виникнення надзвичайних ситуацій;

5) у разі виникнення надзвичайної ситуації до прибуття аварійно-рятувальних підрозділів вживати заходів для рятування населення і майна;

6) дотримуватися протиепідемічного, протиепізоотичного та протиепіфітотичного режимів, режимів радіаційного захисту;

7) виконувати правила пожежної безпеки, забезпечувати будівлі, які їм належать на праві приватної власності, первинними засобами пожежогасіння, навчати дітей обережному поводженню з вогнем.

3. Іноземці та особи без громадянства, які перебувають в Україні на законних підставах, у разі виникнення надзвичайних ситуацій користуються тими самими правами, а також несуть такі самі обов’язки, як і громадяни України, за винятками, встановленими Конституцією, законами чи міжнародними договорами України, згода на обов’язковість яких надана Верховною Радою України.

Список джерел інформаціі

1. Закон України «Про охорону праці» № 2695-XII від 14.10.1992 р. у редакції № 191-VIII від 12.02.2015 р.

2. НПАОП 0.00–1.28–10. Правила охорони праці під час експлуатації електронно-обчислювальних машин. Затверджено наказом Міністерства юстиції України № 65 від 26.03.2010 р.

3. ДСН 3.3.6.037-99. Санітарні норми виробничого шуму, ультразвуку та інфразвуку. Постанова Міністерства охорони здоров’я № 37 від 01.12.1999 р.

4. ДСТУ ГОСТ 12.1.012:2008. Система стандартів безпеки праці. Вібраційна безпека. Загальні вимоги. Від 20.10.2008 р.

5. ДБН В.2.5-28-2006. Державні будівельні норми України. Інженерне обладнання будинків і споруд. Природне і штучне освітлення. Від 01.10.2006 р.

6. ДСТУ ГОСТ 12.1.012:2008. Система стандартів безпеки праці. Вібраційна безпека. Загальні вимоги. Від 20.10.2008 р.

7. ДСН 3.3.6.042-99. Державні санітарні норми. Санітарні норми мікроклімату виробничих приміщень. Постанова Міністерства охорони здоров’я № 42 від 01.12.1999 р.

8. ДСанПіН 3.3.2-007-98. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин. Від 10.12.1998 р.

9. ПУЕ - 2014. Правила улаштування електроустановок. Від 20.11.2014 р.

10. ДБН В.1.2-7-2008. Державні будівельні норми України. Основні вимоги до будівель і споруд. Пожежна безпека. Від 26.01.2008 р.

11. ДСТУ Б В.1.1-36:2016. Державний стандарт України. Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою. Від 15.06.2016 р.

12. ДБН В.1.1-7-2002. Державні будівельні норми України. Пожежна безпека об`єктів будівництва. Від 03.12.2002 р.

13. ДСТУ Б В.2.5. – 38:2008. Державний стандарт України. Інженерне обладнання будинків і споруд. Улаштування блискавкозахисту будівель і споруд. Затверджено наказом Міністерства регіонального розвитку та будівництва України № 269 від 27.06.2008 р., чинний з 01.10.2008 р.

14. Закон України «Про охорону навколишнього природного середовища» від 25.06.1991 р. у редакції № 1264-XII від 01.02.2017 р

15. ДБН В. 2.5-28- 2006 Державні будівельні норми. Інженерне обладнання будівельників і споруд. Природне і штучне освітлення.-К.: Мінбуд України.-2006.

16. ДСН 3.3.6.042 – 99. Санітарні норми мікроклімату виробничих

приміщень. – К.:2000.

. 17. НАПБ Б 03.002-2007 Нормативний акт пожежної безпеки. Норми визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою. – діє з 03.12.2007.

18. ДБН В.2.5-67:2013. Опалення, вентиляція та кондиціонування. Чинний з 01 січня 2013 року.

19. ДБН В.2.5-56- 2014 Система протипожежного захисту. Міністерство

регіонального розвитку, будівництва та житлово-комунального господарства

України. Київ. 2014. Чинний з 01.07.2015р.

20. НАПБ Б.03.001-2001 Типові норми належності вогнегасників. Затв.

Наказом МНС України 02..04.2004 №151 // Офіційний вісник України. 2004,

№19

21. Правила улаштування електроустановок. – Чинний з 20.11.2014. Затв.

Наказом Міністра енергетики та вугільної промисловості України від 20червня 2014 р. №469.

22.Кодекс цивільного захисту України – ВРУ № 5403 – VІ, від 2.10.2012 р.