# 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.

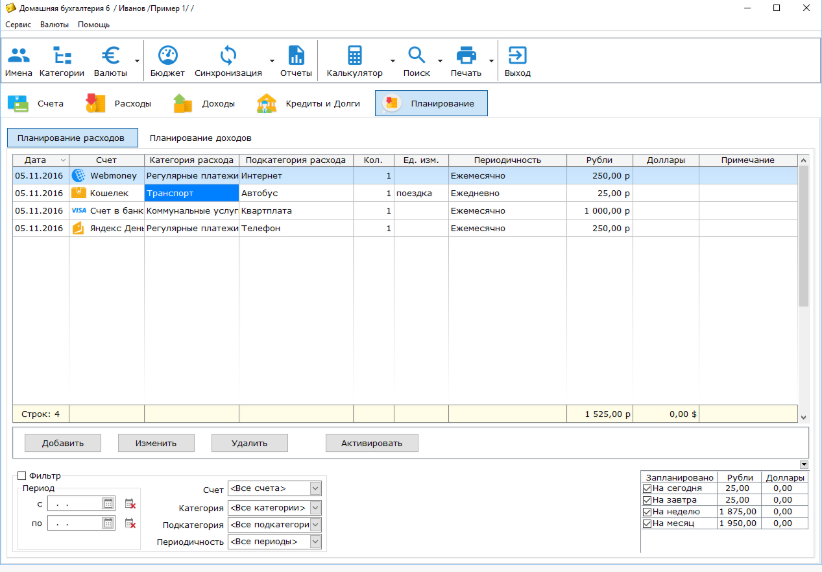
## Аналоги

«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

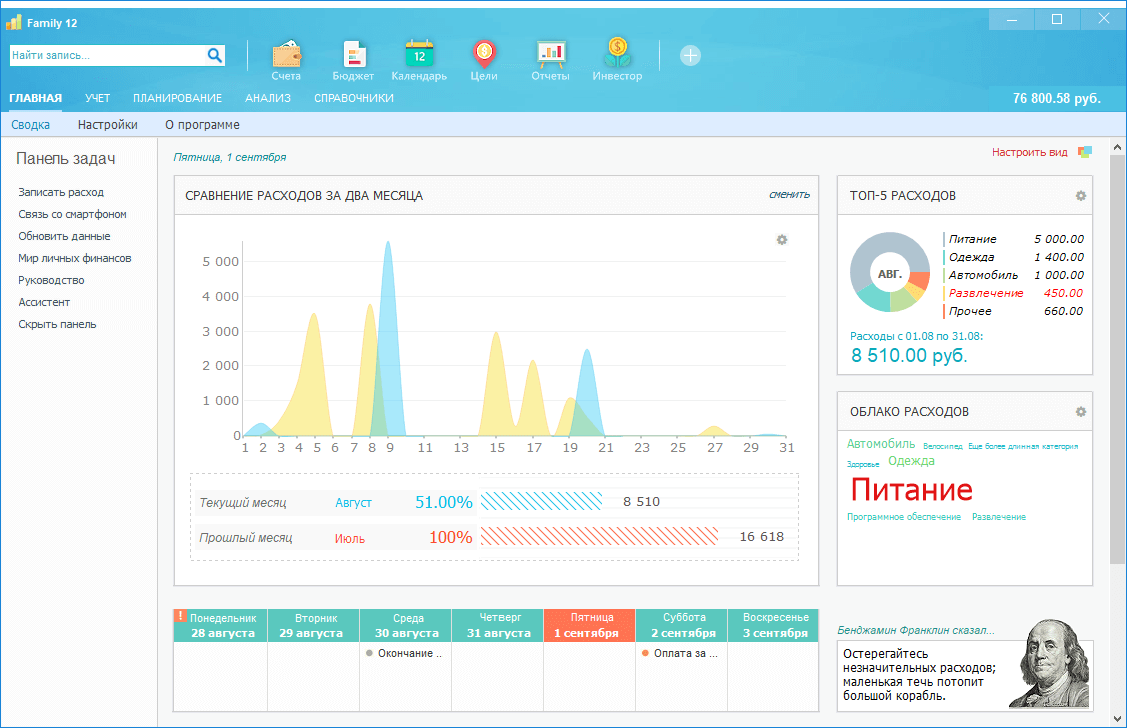


**«family12»** - Домашняя бухгалтерия для Windows.

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

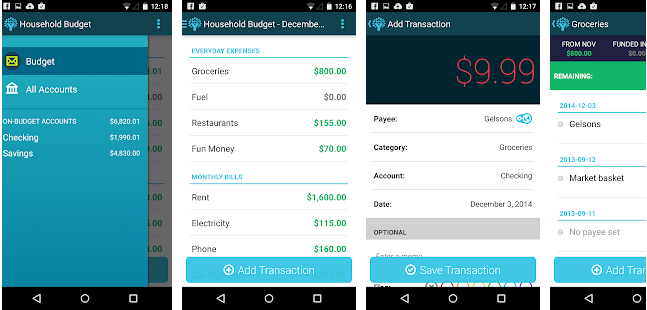
Цена от 125грн до 675грн

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).

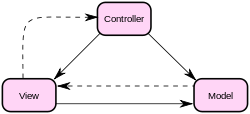
У комп'ютерних технологіях трирівнева архітектура, синонім триланкова архітектура передбачає наявність наступних компонентів програми: [клієнтський](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) [застосунок](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%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.

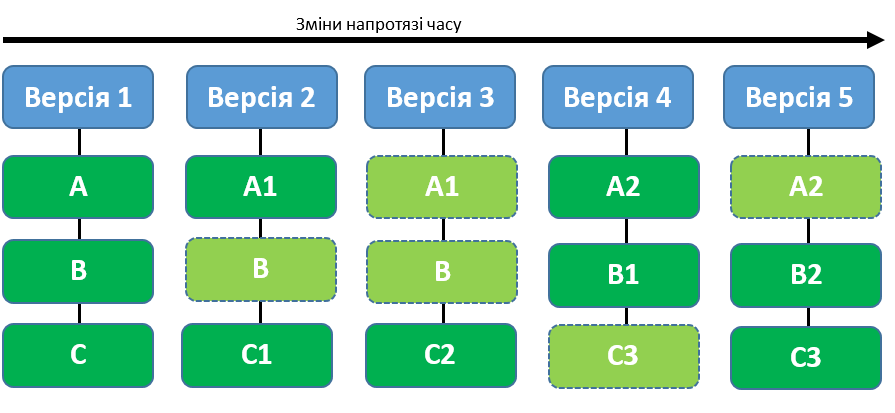
2.1.2 Система керування версіями та місце зберігання коду

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.

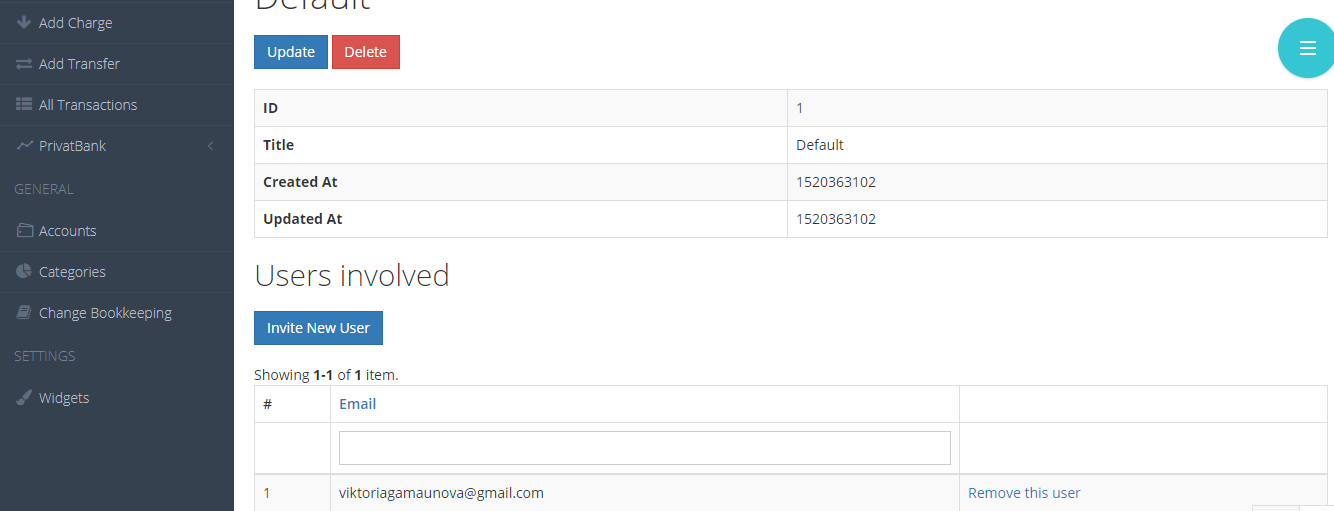
OAuth 2.0 — наступне покоління протоколу OAuth, зворотно не сумісне з OAuth 1.0. OAuth 2.0 фокусується на простоті розробки клієнтської частини, забезпечуючи спеціальні потоки дозволу для [веб-застосунків](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA), 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

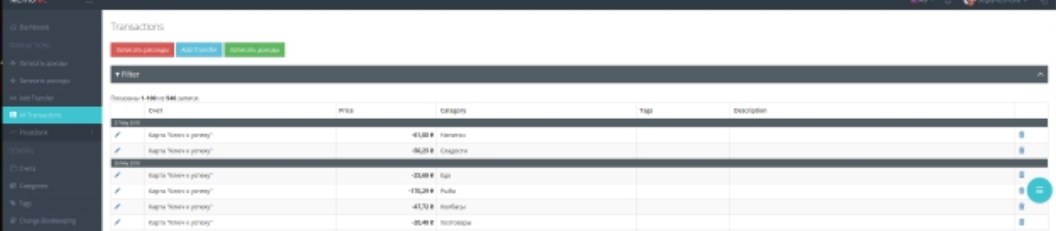
Додаток підтримую багато валют. Завдяки інтеграції API Приват Банку, можливе використання усіх валют, котрі використовує банк, якщо нам недостатньо валют, є можливість запрограмувати додаток на потрібну нам валюту. Конвертація валют виконується по курсу на даний проміжок часу.

### Теги та категорії

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

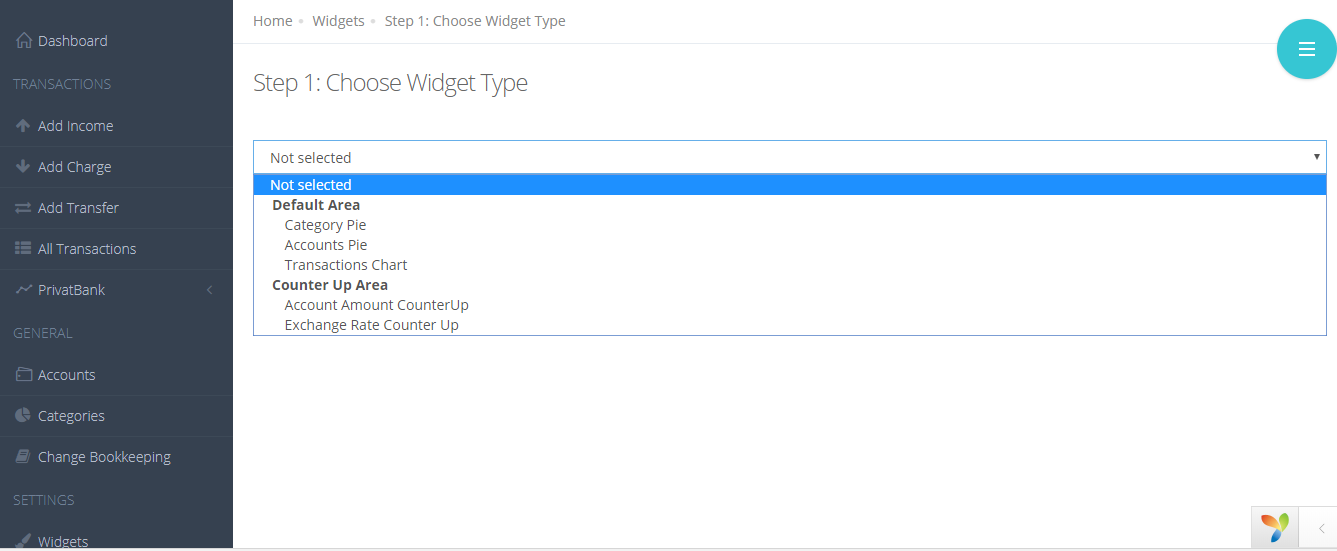
Категорії та теги створюються самим користувачем.

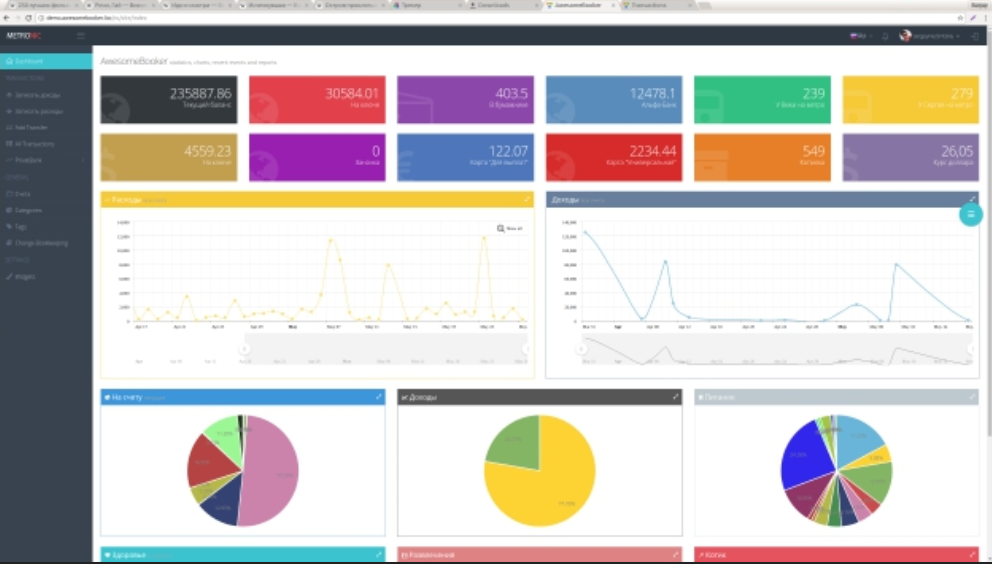
У транзакції може бути лише одна категорія збражено на малюнку 1.9. Категорії мають деревоподібну структуру, тобто кожна категорія може мати під категорію, може бути багато рівнів. У транзакції може бути багато тегів. Теги мають зв'язок багато до багатьох.

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.

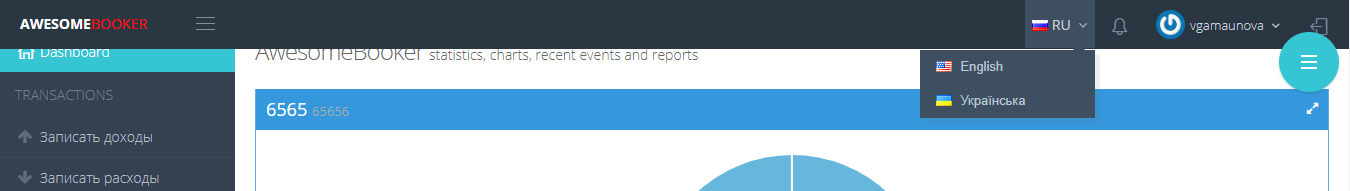
Рисунок 1.10 – Макет першого кроку створення віджету

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

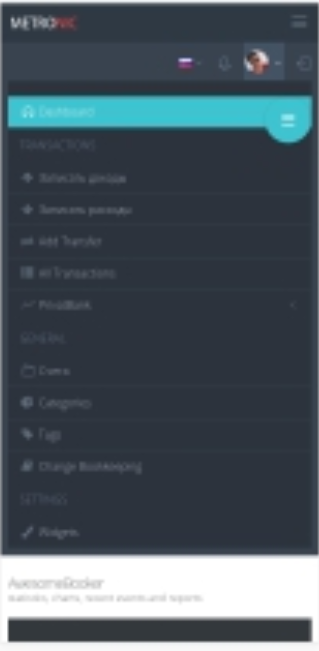
Додаток підтримує українську, російську то англійську мову. Можливе швидке та легке розширення мов рисунок 1.12.

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.

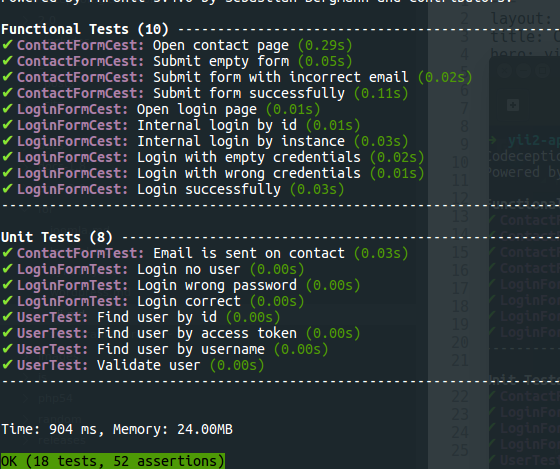
Yii Framework забезпечує базові та розширені шаблони застосунків. Обидва включають зразок Codeception тестів, таким чином, починаючи з Codeception, необхідно запустити новий проект Yii

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:

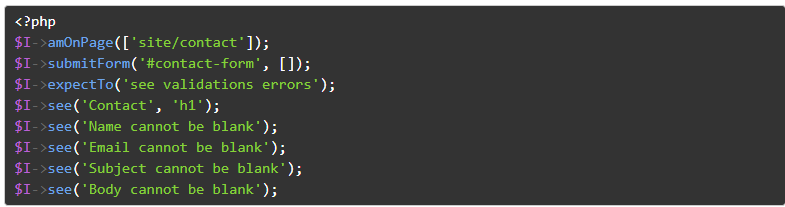
Цей модуль запускає програму Yii для тестового випадку та надає додаткові допоміжні методи для спрощення тестування. Він має лише частину orm та електронної пошти, щоб виключити методи, необхідні лише для функціонального тестування.

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 імітуючий 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

