

Міністерство освіти і науки України
Одеський національний політехнічний університет
Інститут комп'ютерних систем
Кафедра інформаційних систем

КУРСОВА РОБОТА

з дисципліни «Технології створення програмних продуктів»

за темою

«Переналаштуй своє життя - Rewired»

Частина Друга

Виконав:
студент 3-го курсу
групи AI-185
Мудрик В. С.
Перевірив:
Блажко О. А.

Одеса 2020

АНОТАЦІЯ

В курсовій роботі розглядається процес створення програмного продукту «Rewired». В пояснювальній записці у розділах «Проектування» та «Конструювання» детально описано особливості конструювання:

- структур даних в системі керування базами даних MySQL;
- програмних модулів в інструментальному середовищі WordPress з використанням додаткових плагінів та мови програмування PHP.

Результати роботи розміщено на *github*-репозиторіях за адресою:

<https://github.com/RomaNNtic/Rewired>

ПЕРЕЛІК СКОРОЧЕНЬ

ОС – операційна система

ІС – інформаційна система

БД – база даних

СКБД – система керування базами даних

ПЗ – програмне забезпечення

ПП– програмний продукт

UML – уніфікована мова моделювання

ЗМІСТ

1 Вимоги до програмного продукту.....	7
1.1 Визначення потреб споживача.....	7
1.1.1 Ієрархія потреб споживача.....	7
1.1.2 Деталізація матеріальної потреби.....	8
1.2 Бізнес вимоги до ПП.....	8
1.2.1 Опис проблем користувача.....	8
1.2.1.1 Концептуальний опис проблеми споживача.....	8
1.2.1.2 Метричний опис проблеми користувача.....	9
1.2.2 Мета створення ПП.....	9
1.2.2.1 Проблемний аналіз існуючих ПП.....	9
1.2.2.2 Мета створення ПП.....	10
1.2.3 Назва ПП.....	10
1.2.3.1 Гасло ПП.....	10
1.2.3.2 Логотип ПП.....	11
1.3 Вимоги користувача до ПП.....	11
1.3.1 Історія користувача ПП.....	11
1.3.2 Діаграма прецедентів ПП.....	12
1.3.3 Сценарії використання прецедентів ПП.....	12
1.4 Функціональні вимоги до ПП.....	19
1.4.1 Багаторівнева класифікація функціональних вимог.....	19
1.4.2 Функціональний аналіз існуючих ПП.....	20
1.5 Нефункціональні вимоги до ПП.....	22
1.5.1 Опис зовнішніх інтерфейсів.....	22

1.5.1.1	Опис інтерфейса користувача.....	22
1.5.1.1.1	Опис INPUT-інтерфейса користувача.....	22
1.5.1.1.2	Опис OUTPUT-потоків.....	25
1.5.1.2	Опис інтерфейсу з зовнішніми пристроями.....	27
1.5.1.3	Опис програмних інтерфейсів.....	27
1.5.1.4	Опис інтерфейсів передачі інформації.....	28
1.5.1.5	Опис атрибутів продуктивності.....	28
2	Планування процесу розробки програмного продукту.....	29
2.1	Планування ітерацій розробки програмного продукту.....	29
2.2	Концептуальний опис архітектури програмного продукту.....	30
2.3	План розробки ПП.....	30
2.3.1	Оцінка трудомісткості розробки ПП.....	30
2.3.2	Визначення дерева робіт з розробки ПП.....	31
2.3.3	Графік робіт з розробки ПП.....	34
2.3.3.1	Таблиця з графіком робіт.....	34
2.3.3.2	Діаграма Ганта.....	35
3	Проектування ПП.....	36
3.1	Концептуальне та логічне проектування структур даних ПП.....	36
3.1.1	Концептуальне проектування на основі UML-діаграми концептуальних класів.....	36
3.1.2	Логічне проектування структур даних.....	37
3.2	Проектування програмних класів.....	38
3.3	Проектування алгоритмів роботи методів програмних класів.....	38
3.4	Проектування тестових наборів методів програмних класів.....	43
4	Конструювання ПП.....	45

4.1 Особливості конструювання структур даних.....	46
4.1.1 Особливості інсталяції та роботи з СУБД.....	46
4.1.2 Особливості створення структур даних.....	46
4.2 Особливості конструювання програмних модулів.....	48
4.2.1 Особливості роботи з інтегрованим середовищем розробки.....	48
4.2.2 Особливості створення програмної структури з урахуванням спеціалізованого фреймворку.....	49
4.2.3 Особливості створення програмних модулів.....	51
4.2.4 Особливості розробки алгоритмів методів програмних класів або процедур/функцій.....	52
4.3 Тестування програмних модулів.....	55
5 Розгортання та валідація ПП.....	56
5.1 Інструкція з встановлення ПП.....	56
5.2 Інструкція з використання ПП.....	56
5.3 Результати валідації ПП.....	60
Висновки.....	61
Джерела.....	61

1 ВИМОГИ ДО ПРОГРАМНОГО ПРОДУКТУ

1.1 Визначення потреб споживача

1.1.1 Ієрархія потреб споживача

Відомо, що в теорії маркетингу потреби людини можуть бути представлені у вигляді ієрархії потреб ідей американського психолога Абрахама Маслоу включають рівні:

- фізіологія (вода, їжа, житло, сон);
- безпека (особиста, здоров'я, стабільність),
- приналежність (спілкування, дружба, любов),
- визнання (повага оточуючих, самооцінка),
- самовираження (вдосконалення, персональний розвиток).

На рисунку 1.1 представлено одну ієрархію потреби споживача, яку хотілося б задовольнити, використовуючи майбутній програмний продукт.

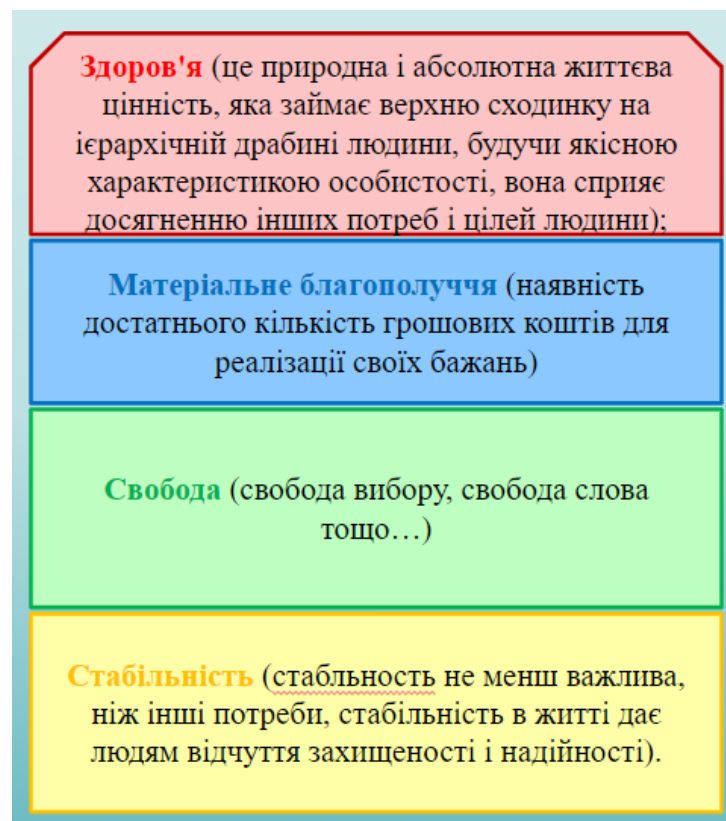


Рис. 1 – Ієрархія потреб споживача наш

1.1.2 Деталізація матеріальної потреби

Матеріальні потреби – це сукупність матеріальних благ, які мають матеріальну форму і виступають як головний спонукальний мотив трудової діяльності людини.

Споживач – це людина, яка споживає продукти виробництва і сфери послуг для відтворення робочої сили.

Тому була винайдена деталізація матеріальної потреби для нашого ПП.

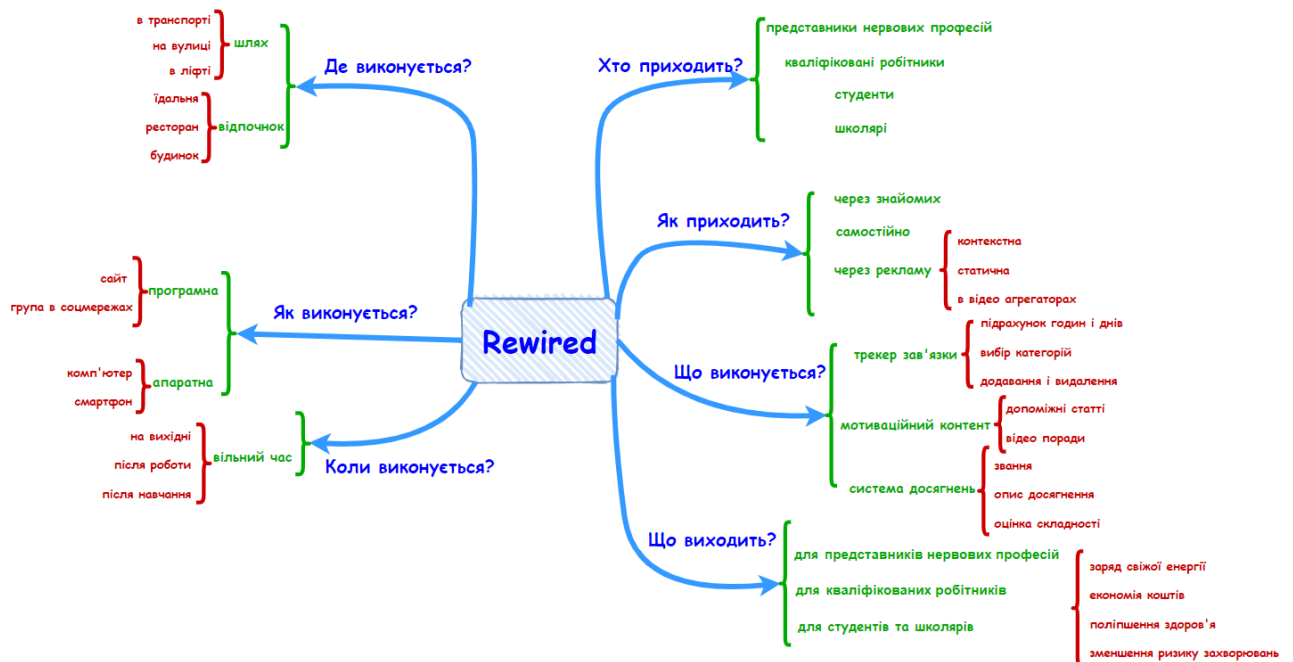


Рис. 2 - матеріальні потреби майбутнього ПП

1.2 Бізнес вимоги до ПП

1.2.1 Опис проблем користувача

1.2.1.1 Концептуальний опис проблеми споживача

Умова задоволення потреб переведення даних в інформацію:

Доступність, Представленість мовою споживача, Інтерес, Актуальність.

Загальний опис проблеми: важко стежити за прогресом і контролювати себе під час позбуття від шкідливих звичок.

1.2.1.2 Метричний опис проблеми користувача

Метричні показники незадоволеності споживача: мала кількість корисних рішень для подолання залежності та високий рівень агітації шкідливих звичок в повсякденному житті.

Формула розрахунку доступності така:

$Availability = (AST - DT) / AST \times 100 = \text{Service or Component Availability (\%)}$ де
AST (agreed service time) - узгоджений час надання послуги;

DT (actual downtime during agreed service time) - фактичний час, коли послуга була недоступна протягом узгодженого часу її надання.

1.2.2 Мета створення ПП

1.2.2.1 Проблемний аналіз існуючих ПП

№	Назва продукту	Вартість	Ступінь готовності	Примітка
1	Sobriety Counter – Bad Habits	Безкоштовно	1	Недоступність на стаціонарних ПК
2	JustSayNo	Безкоштовно	1	Недоступна для більшості користувачів
3	HabitShare	Безкоштовно	1	Немає системи досягнень для користувачів

Таблиця 1 – Аналіз існуючих програмних продуктів

1.2.2.2 Мета створення ПП

Підвищення рівня доступності програмного продукту для всіх соціальних рівнів. Допомога користувачам в налагодженні життєвого ритму, саморозвитку і особистісному зростанні, шляхом позбавлення від залежностей і шкідливих звичок.

1.2.3 Назва ПП

Rewired

1.2.3.1 Гасло ПП

Переналаштуй своє життя - Rewired

1.2.3.2 Логотип ПП



Рис. 3 – Логотип програмного продукту

1.3 Вимоги користувача до ПП

1.3.1 Історія користувача ПП

- Як споживач, я маю можливість зберігати особисті дані;
- Як споживач, я маю можливість змінювати особисті дані;
- Як споживач, я маю можливість переглядати пункти меню і переміщатися між сторінками;
- Як споживач, я маю можливість створювати змінювати і видаляти звички;
- Як споживач, я маю можливість додавати таймер на кожну звичку;
- Як споживач, я маю можливість додавати опис негативної та позитивної сторони кожної звички;
- Як споживач, я маю можливість отримувати досягнення та отримувати і втрачати звання в залежності від часу утримання;
- Як споживач, я маю можливість переглядати мотиваційний контент;
- Як споживач, я маю можливість переглядати цитату дня;
- Як споживач, я маю можливість лайкати або ділитися цим контентом з друзями;
- Як споживач, я маю можливість вести свою статистику і додавати свої спостереження в замітки;
- Як споживач, я маю можливість переглядати контактну інформацію і зв'язуватися з підтримкою.

1.3.2 Діаграма прецедентів ПП

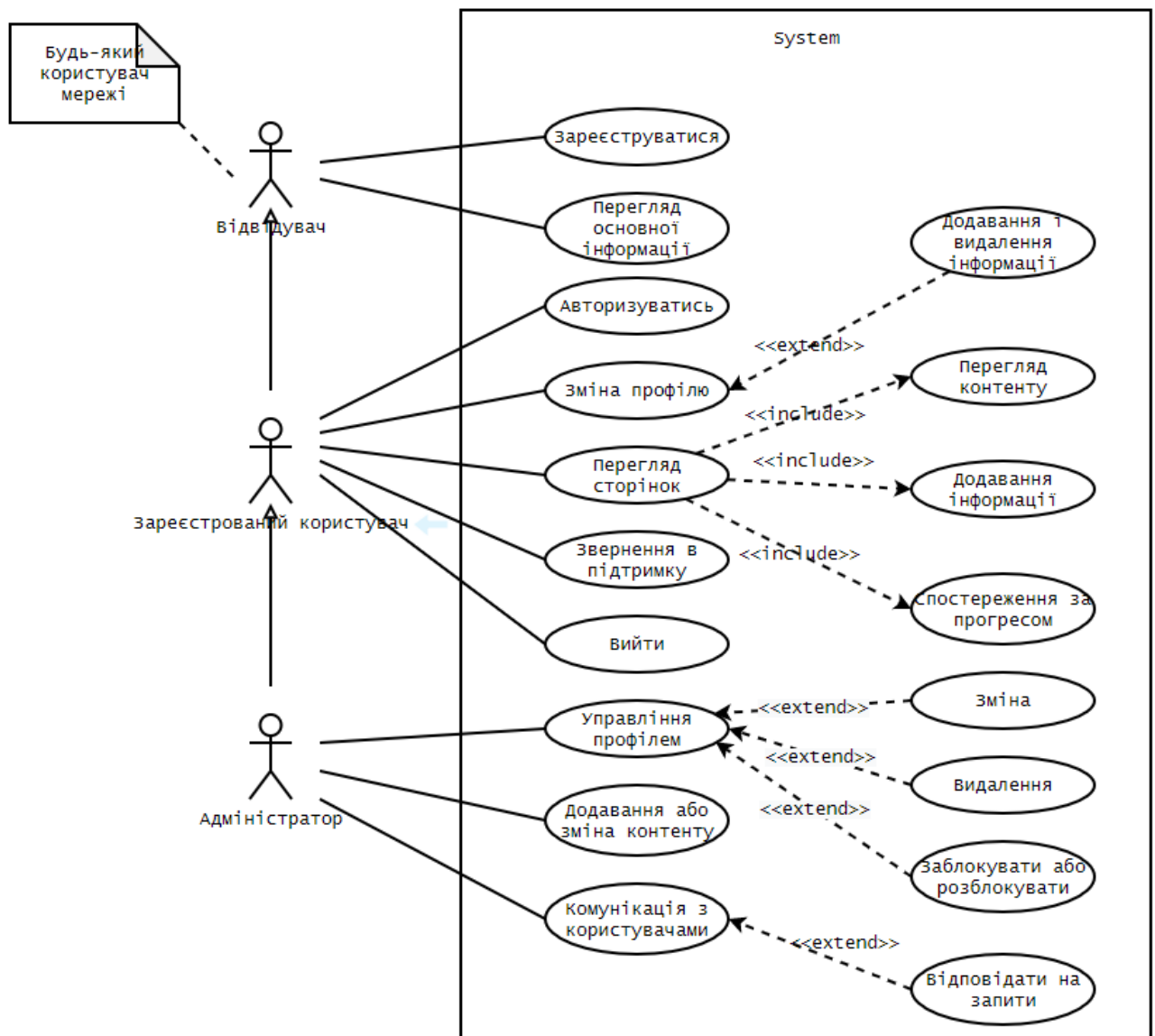


Рис. 4 – Діаграма прецедентів ПП

1.3.3 Сценарії використання прецедентів ПП

Прецедент «Зареєструвати користувача»:

Передумови початку: відвідувача зацікавив ПП і він хоче створити обліковий запис

Актори: Відвідувач

Актор-основна зацікавлена особа: відвідувач

Гарантії успіху: відвідувач отримає власний обліковий запис та доступ до додаткової інформації.

Приклад основного успішного сценарію прецеденту «Зареєструвати користувача»:

1. ПП надає відвідувачеві можливість реєстрації
2. Відвідувач переходить на сторінку реєстрації
3. ПП відображає форму реєстрації у вигляді поля для введення ідентифікації та аутентифікації
4. Відвідувач заповнює поля форми ідентифікації та аутентифікації і підтверджує реєстрацію
5. ПП реєструє відвідувача.

Приклад альтернативного сценарію для попереднього

прикладу основного успішного сценарію прецеденту «Зареєструвати користувача»:

5.1 ПП виявляє, що відвідувач неправильно заповнив форму, видає повідомлення про помилку і переходить до кроку 1.

Прецедент «Перегляд основної інформації»:

Передумови початку: відвідувач тільки зайшов і поки нічого не знає про ПП, він хоче ознайомитися з ПП

Актори: відвідувач та зареєстрований користувач

Актор-основна зацікавлена особа: відвідувач

Гарантії успіху: відвідувач зацікавиться ПП і стане новим користувачем

Приклад основного успішного сценарію прецеденту «Перегляд основної інформації»:

1. ПП надає ресурс з основною ознайомчою інформацією

2. Відвідувач реєструється

Приклад альтернативного сценарію для попереднього прикладу основного успішного сценарію прецеденту «Перегляд основної інформації»:

3.1 Відвідувач покидає ресурс.

Прецедент «Зміна профілю»:

Передумови початку: користувач хоче змінити інформацію в профілі (наприклад: нікнейм, ім'я користувача, ел. адресу)

Актори: зареєстрований користувач

Актор-основна зацікавлена особа: зареєстрований користувач

Гарантії успіху: користувач може розповісти про себе, заявити про себе широкому загалу

Приклад основного успішного сценарію прецеденту «Зміна профілю»:

1. ПП надає можливість користувачам редагувати профіль

2. Користувач переходить на сторінку і змінює особисті дані

3. ПП отримує запит на зміну даних

4. ПП перевіряє і зберігає нові дані.

Приклад альтернативного сценарію для попереднього прикладу основного успішного сценарію прецеденту «Зміна профілю»:

4.1 ПП виявляє, що користувач передав їй неправильні значення, видає повідомлення про помилку і переходить до кроку 2.

Прецедент «Перегляд сторінок»:

Передумови початку: користувач хоче більше дізнатися про ПП

Актори: зареєстрований користувач

Актор-основна зацікавлена особа: зареєстрований користувач

Гарантії успіху: користувач дізнається про існування інших потенційно цікавих йому розділів.

Приклад основного успішного сценарію прецеденту «Перегляд сторінок»:

1. ПП надає можливість навігації між сторінками у вигляді пунктів меню

2. Користувач переглядає пункти меню і переходить на вподобану сторінку

Приклад альтернативного сценарію для попереднього прикладу основного успішного сценарію прецеденту «Перегляд сторінок»:

2.1 Користувач переходить на іншу сторінку.

Прецедент «Звернення в підтримку»:

Передумови початку: у клієнта (користувача), виникла проблема або він хоче задати питання

Актори: зареєстрований користувач та адміністратор

Актор-основна зацікавлена особа: зареєстрований користувач

Гарантії успіху: користувач вирішить проблему, репутація ПП не постраждає.

Приклад основного успішного сценарію прецеденту «Звернення в підтримку»:

1. ПП надає можливість зв'язатися з підтримкою у вигляді поля для введення запиту
2. Користувач заповнює поле і відправляє запитів в технічну підтримку
3. ПП отримує запит від клієнта і фіксує його
4. Адміністратор бачить запит від клієнта і зв'язується з ним

Приклад альтернативного сценарію для попереднього прикладу основного успішного сценарію прецеденту «Звернення в підтримку»:

- 3.1 Користувач неправильно заповнив поле або запит не пройшов
- 3.2 ПП повідомляє користувача про помилку.

Прецедент «Вийти»:

Передумови початку: користувач хоче вийти зі свого облікового запису

Актори: зареєстрований користувач

Актор-основна зацікавлена особа: зареєстрований користувач

Гарантії успіху: користувачів може зайти в інший обліковий запис або перезайти в свій.

Приклад основного успішного сценарію прецеденту «Вийти»:

1. ПП надає можливість виходу з облікового запису
2. Користувач використовує цю можливість і виходить
3. ПП завершує сесію користувача
4. ПП видає користувачеві повідомлення з приводу успішного виходу і переводить його на головну сторінку.

Приклад альтернативного сценарію для попереднього прикладу основного успішного сценарію прецеденту «Вийти»:

3.1 ПП не вдається завершити сесію і він видає повідомлення про помилку при виході.

Прецедент «Управління профілем»:

Передумови початку: обліковий запис користувача неправильно заповнена або містить заборонений контент

Актори: адміністратор

Актор-основна зацікавлена особа: адміністратор

Гарантії успіху: успішне редагування деталей облікового запису користувача

Приклад основного успішного сценарію прецеденту «Управління профілем»:

1. ПП надає можливість пошуку і редагування профілів користувачів адміністратору
2. Адміністратор знаходить потрібного користувача
3. Адміністратор змінити дані користувача і запитує у ПП дозвіл на збереження
4. ПП перевіряє коректність змінюваних даних
5. ПП зберігає нові дані і видає відповідне повідомлення.

Приклад альтернативного сценарію для попереднього прикладу основного успішного сценарію прецеденту «Управління профілем»:

4.1 ПП виявляє неправильні значення, видає повідомлення про помилку і переходить до кроку 1.

Прецедент «Додавання або зміна контенту»:

Передумови початку: матеріал розміщений на сторінках сайту втратив свою актуальність або потребує зміни

Актори: адміністратор

Актор-основна зацікавлена особа: адміністратор

Гарантії успіху: завжди якісна та актуальна інформація на сайті

Приклад основного успішного сценарію прецеденту «Додавання або зміна контенту»:

1. ПП надає можливість адміністратору редагувати контент
2. Адміністратор обирає місце яке потребує редагування або поповнення інформацією
3. Адміністратор запитує дозвіл на збереження змін
4. ПП отримує запит на зміну даних
5. ПП зберігає нові дані і видає відповідне повідомлення.

Приклад альтернативного сценарію для попереднього прикладу основного успішного сценарію прецеденту «Додавання або зміна контенту»:

- 4.1 ПП виявляє неправильно заповнені поля, видає повідомлення про помилку і переходить до кроку 1.

Прецедент «Комунікація з користувачами»:

Передумови початку: прийшов запит від користувача про допомогу

Актори: адміністратор, зареєстрований користувач

Актор-основна зацікавлена особа: адміністратор

Гарантії успіху: утримування потенційних клієнтів, підвищення рейтингу

Приклад основного успішного сценарію прецеденту «Комунікація з користувачами»:

1. ПП надає можливість зв'язку з користувачами
2. ПП надсилає запит від користувача адміністратору
3. Адміністратор бачить запит від клієнта та зв'язується з ним.

1.4 Функціональні вимоги до ПП

1.4.1 Багаторівнева класифікація функціональних вимог

Ідентифікатор функції	Назва функції
FR1	Реєстрація користувача
FR1.1	ПП відображає форму реєстрації у вигляді поля для введення ідентифікації та аутентифікації
FR1.2	Відвідувач заповнює поля форми ідентифікації та аутентифікації і підтверджує реєстрацію
FR1.3	ПП реєструє користувача
FR1.4	ПП виявляє, що відвідувач неправильно заповнив форму, видає повідомлення про помилку і переходить до кроку 1
FR2	Перегляд основної інформації
FR2.1	ПП надає ресурс з основною ознайомчою інформацією
FR2.2	Відвідувач реєструється
FR2.3	Відвідувач покидає ресурс
FR3	Зміна профілю
FR3.1	ПП надає можливість користувачам редагувати профіль

FR3.2	Користувач переходить на сторінку і змінює особисті дані
FR3.3	ПП отримує запит на зміну даних
FR3.4	ПП перевіряє і зберігає нові дані
FR3.5	ПП виявляє, що користувач надав неправильні значення, видає повідомлення про помилку і переходить до кроку 2
FR4.	Перегляд сторінок
FR4.1	ПП надає можливість навігації між сторінками у вигляді пунктів меню
FR4.2	Користувач переглядає пункти меню і переходить на сторінку
FR4.3	Користувач переходить на іншу сторінку
FR5	Звернення в підтримку
FR5.1	ПП надає можливість зв'язатися з підтримкою у вигляді поля для введення запиту
FR5.2	Користувач заповнює поле і відправляє запитів в технічну підтримку
FR5.3	ПП отримує запит від клієнта і фіксує його
FR5.4	Адміністратор бачить запит від клієнта і зв'язується з ним
FR5.5	Користувач неправильно заповнив поле або запит не пройшов
FR5.6	ПП повідомляє користувача про помилку
FR6	Вийти
FR6.1	ПП надає можливість виходу з облікового запису
FR6.2	Користувач виходить
FR6.3	ПП завершує сесію користувача і виводить відповідне повідомлення
FR6.4	ПП не вдається завершити сесію і він видає повідомлення про помилку при виході
FR7	Управління профілем
FR7.1	ПП надає можливість пошуку і редагування профілів користувачів адміністратору

FR7.2	Адміністратор знаходить потрібного користувача
FR7.3	Адміністратор змінити дані користувача і запитує у ПП дозвіл на збереження
FR7.4	ПП перевіряє коректність змінюваних даних
FR7.5	ПП зберігає нові дані і видає відповідне повідомлення
FR7.6	ПП виявляє неправильні значення, видає повідомлення про помилку і переходить до кроку 1
FR8	Додавання або зміна контенту
FR8.1	ПП надає можливість адміністратору редагувати контент
FR8.2	Адміністратор обирає місце яке потребує редагування або поповнення інформацією
FR8.3	Адміністратор запитує дозвіл на збереження змін
FR8.4	ПП отримує запит на зміну даних
FR8.5	ПП зберігає нові дані і видає відповідне повідомлення
FR8.6	ПП виявляє неправильно заповнені поля, видає повідомлення про помилку і переходить до кроку 1
FR9	Комунікація з користувачами
FR9.1	ПП надає можливість зв'язку з користувачами
FR9.2	ПП надсилає запит від користувача адміністратору
FR9.3	Адміністратор бачить запит від клієнта та зв'язується з ним

Таблиця 2 – Функціональні вимоги до ПП

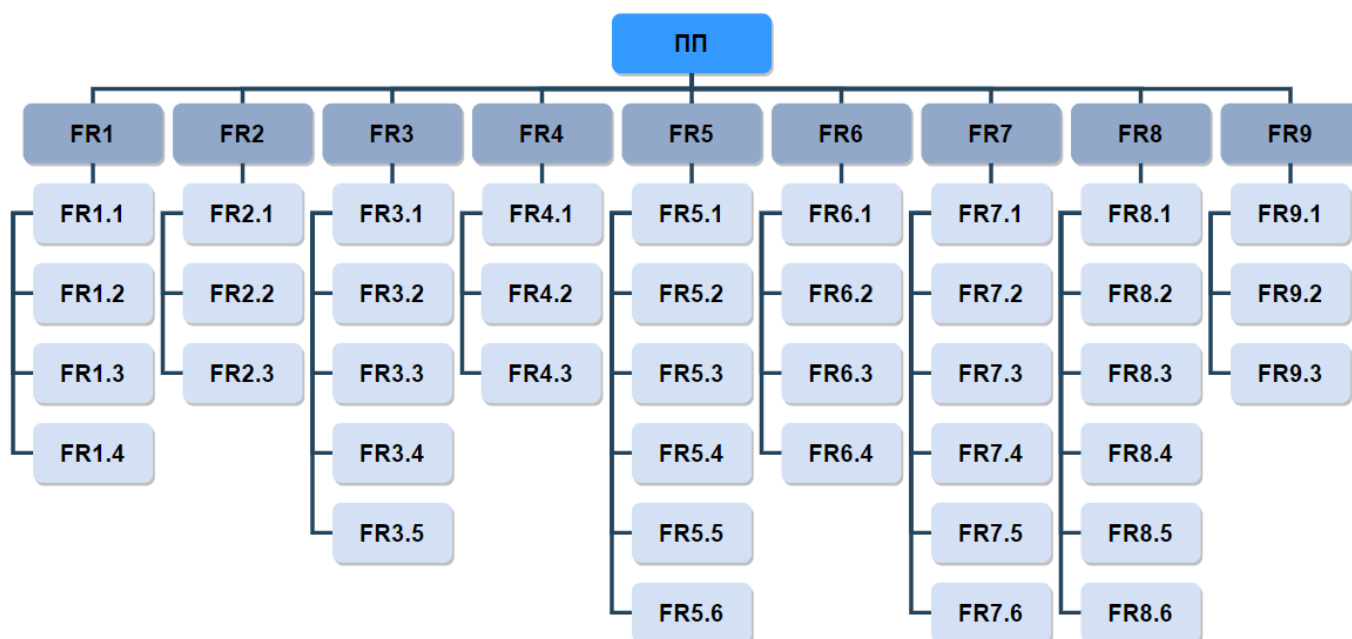


Рис. 5 – WBS структура вимог до ПП

1.4.2 Функціональний аналіз існуючих ПП

Ідентифікатор функції	Sobriety Counter – Bad Habits	JustSayNo	HabitShare
FR1	-	+	+
FR2	+	+	+
FR3	+	+	+
FR4	-	-	-
FR5	+	+	+
FR6	-	+	+
FR7	-	+	+
FR8	-	-	-
FR9	+	+	+

Таблиця 3 – Функціональний аналіз існуючих ПП

1.5 Нефункціональні вимоги до ПП

1.5.1 Опис зовнішніх інтерфейсів

1.5.1.1 Опис інтерфейса користувача

1.5.1.1.1 Опис INPUT-інтерфейса користувача

Ідентифікатор функції	Засіб INPUT-потoku	Особливості використання
FR1.2	- 2/3-кнопочний маніпулятор типу "миша"; - сенсорний екран (Touchscreen, Touchpad, Multi-touch);	Використання миші і кнопки миші або сенсорного екрану для переходу на сторінку
FR1.3	- 2/3-кнопочний маніпулятор типу "миша"; - стандартна комп'ютерна клавіатура; - сенсорний екран (Touchscreen, Touchpad, Multi-touch);	Використання миши або сенсорного екрану для переміщення і підтвердження, а клавіатуру для введення даних
FR1.4		
FR2.1		
FR2.2		
FR2.3		
FR3.2	- 2/3-кнопочний маніпулятор типу "миша"; - стандартна	Використання миші і сенсорного екрану для переміщення, а клавіатури для введення








	комп'ютерна	змінюваних даних
FR4.1	клавіатура;	Використання миші або сенсорного екрану для переміщення по сторінці і між сторінками
FR4.2	- сенсорний екран	
FR4.3	(Touchscreen, Touchpad, Multi-touch);	
FR5.2		
FR5.4		Використання миші і сенсорного екрану для переміщення, а клавіатури для введення повідомлення
FR6.2	- 2/3-кнопочний	Використання миші або сенсорного екрану для навігації
FR6.3	маніпулятор	
FR6.4	типу "миша";	
	- сенсорний екран (Touchscreen, Touchpad, Multi-touch);	
FR7.2	- 2/3-кнопочний	Використання миші і сенсорного екрану для переміщення, а клавіатури для введення змінюваних даних
FR7.3	маніпулятор	
	типу "миша";	
	- стандартна комп'ютерна клавіатура;	
	- сенсорний екран (Touchscreen, Touchpad, Multi-touch);	
FR7.5	- 2/3-кнопочний	Використання миші або сенсорного екрану для
FR7.6	маніпулятор	

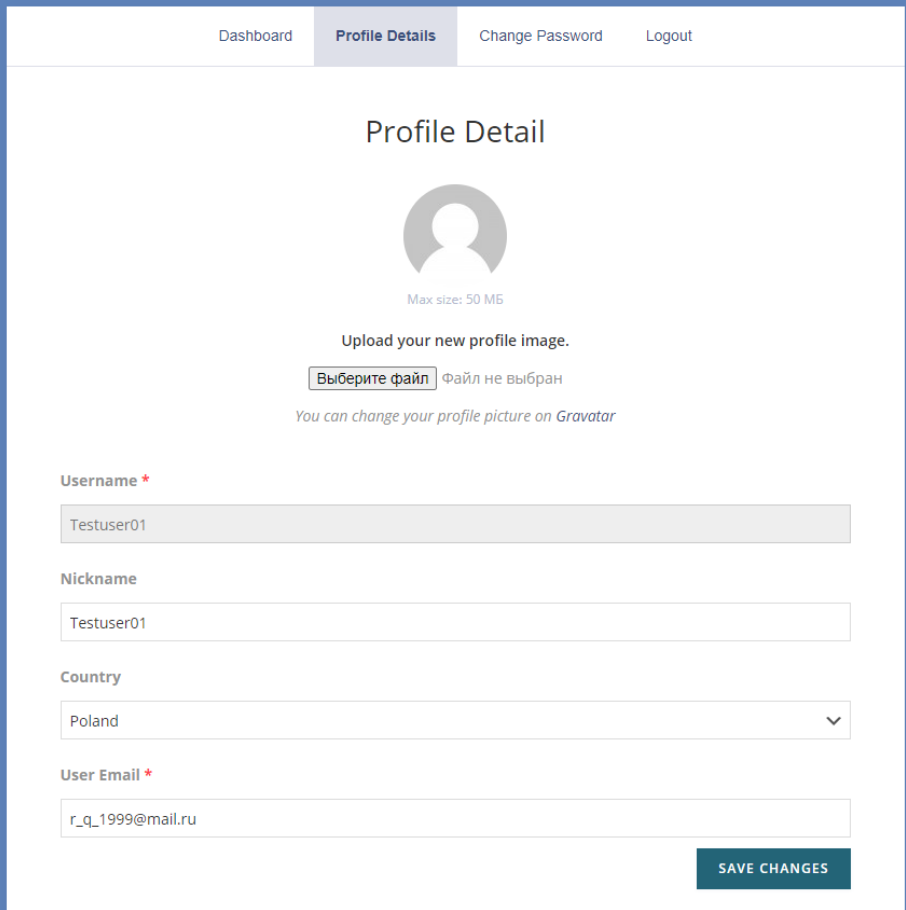
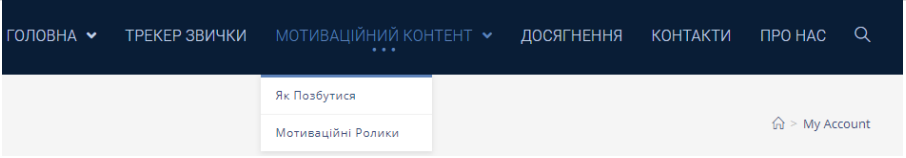
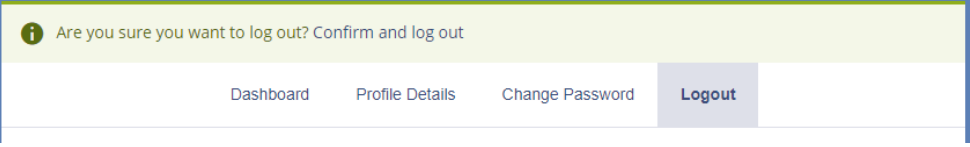
	типу "миша"; - сенсорний екран (Touchscreen, Touchpad, Multi-touch);	навігації
FR8.2	- 2/3-кнопочний маніпулятор типу "миша"; - стандартна комп'ютерна клавіатура; - сенсорний екран (Touchscreen, Touchpad, Multi-touch);	Використання миші і сенсорного екрану для переміщення, а клавіатури для введення змінюваних даних
FR8.3		
FR8.5	- 2/3-кнопочний маніпулятор типу "миша"; - сенсорний екран (Touchscreen, Touchpad, Multi-touch);	Використання миші або сенсорного екрану для навігації
FR8.6		
FR9.2		
FR9.3		

Таблиця 4 – INPUT-інтерфейси

1.5.1.1.2 Опис OUTPUT-потоків

Ідентифікатор функції	Засіб OUTPUT - потоків
--------------------------	---------------------------

FR1	<div data-bbox="526 152 1457 728"> <h3>Логин и пароль</h3> <div> <div>Username </div> <input data-bbox="568 259 1423 288" type="text"/> </div> <div> <div>Nickname</div> <input data-bbox="568 331 1423 360" type="text"/> </div> <div> <div>Country</div> <div>Afghanistan </div> </div> <div> <div>User Password </div> <input data-bbox="568 479 1423 508" type="password"/>  </div> <div> <div>User Email </div> <input data-bbox="568 553 1423 582" type="text"/> </div> <div> <div>Confirm Password </div> <input data-bbox="568 627 1423 656" type="password"/>  </div> <div>SUBMIT</div> </div>
FR2	<div data-bbox="526 745 1457 1370"> <h3>Цитата дня</h3> <p>«Хочете знати, хто ви? Не питайте. Дійте! Дія буде описувати і визначати вас », - Томас Джефферсон.</p> <h2>ШКІДЛИВІ ЗВИЧКИ І ЗАЛЕЖНОСТІ</h2> <p>У сучасному світі практично неможливо знайти людину, у якого не було б шкідливих звичок або залежностей. До некорисних пристрастей відносять ті, які згубно впливають на здоров'я людини. Це стандартний набір: надмірне вживання алкоголю, тютюнопаління, наркоманія, переїдання. Насправді шкідливих звичок, негативно діють на людський організм, набагато більше, просто люди не замислюються про це, хоча варто було б. Подолання залежності - це важко. За допомогою нашого проекту ви зможете відслідковувати всі ваші шкідливі звички і залежності. Використовуйте ресурс, щоб проаналізувати їх і звільнитися від них на завжди. Прискорте своє саморозвиток і особистісний ріст - позбудьтесь від своїх залежностей і шкідливих звичок. Покращуйте свої пізнання за допомогою статей, матівіруйте себе роликами і не здавайтесь!</p> <h2>ФУНКЦІОНАЛ</h2> </div>

FR3	
FR4	
FR6	

Таблиця 5 – Засоби OUTPUT-потоків

1.5.1.2 Опис інтерфейсу із зовнішніми пристроями

Ідентифікатор функції	Зовнішній пристрій
FR1	- Desktop-персональний комп`ютер; - Notebook;
FR2	
FR3	
FR4	
FR5	

FR6	- смартфон; - мобільний телефон;
FR7	
FR8	
FR9	

Таблиця 6 – Опис інтерфейсу з зовнішніми пристроями

1.5.1.3 Опис програмних інтерфейсів

Версії операційних систем та програмних бібліотек, які знадобляться при реалізації більшості функцій ПП.

Версії операційних систем та програмних бібліотек, які знадобляться при реалізації більшості функцій ПП.	<ul style="list-style-type: none"> - Windows - Linux - PHP - CMS
--	--

Таблиця 7 – Опис програмних інтерфейсів

1.5.1.4 Опис інтерфейсів передачі інформації

Інтерфейси передачі інформації, які знадобляться при реалізації більшості функцій ПП:

- провідні інтерфейси:
 - Ethernet
- безпроводні інтерфейси:
 - Wi-Fi

1.5.1.5 Опис атрибутів продуктивності

Ідентифікатор функції	Максимальний час реакції ПП на дії користувачів, секунди
FR1	5
FR2	2
FR3	5
FR4	2
FR5	1
FR6	3
FR7	5
FR8	2
FR9	2

Таблиця 8 – Опис атрибутів продуктивності

2 ПЛАНУВАННЯ ПРОЦЕСУ РОЗРОБКИ ПРОГРАМНОГО ПРОДУКТУ

2.1 Планування ітерацій розробки програмного продукту

Ідентифікатор функції	Назва функції	Функціональні залежності	Вплив на досягнення мети, %	Пріоритет функції
FR1	Реєстрація користувача	-	10%	M (Must)
FR2	Перегляд основної інформації	-	20%	S (Should)
FR3	Зміна профілю	FR1	5%	S (Should)
FR4	Перегляд	FR2	15%	M (Must)

	сторінок			
FR5	Звернення в підтримку	-	5%	C (Could)
FR6	Вийти	FR1	5%	C (Could)
FR7	Управління профілем	FR1	10%	W (Want)
FR8	Додавання або зміна контенту	-	20%	M (Must)
FR9	Комунікація з користувачами	-	10%	C (Could)

Таблиця 9 – Планування ітерацій розробки

2.2 Концептуальний опис архітектури програмного продукту

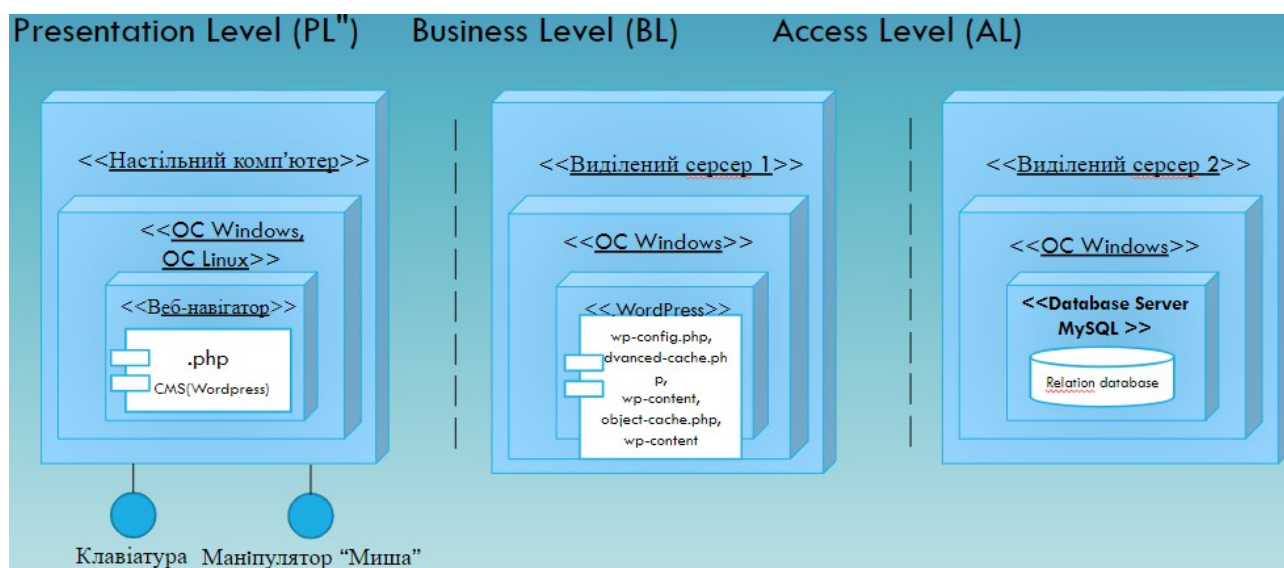


Рис. 6 – Концептуальний опис архітектури ПП

2.3 План розробки ПП

2.3.1 Оцінка трудомісткості розробки ПП

1. Визначення вагових показників акторів

Назва актора	Тип актора	Ваговий коефіцієнт
Відвідувач	Простий	1
Зареєстрований користувач	Складний	3
Адміністратор	Середній	2

Таблиця 10 – Вагові коефіцієнти акторів

$$A = 1 + 3 + 2 = 6$$

2. Визначення вагових показників прецедентів UC

Назва прецедента	Тип прецедента	Кількість кроків сценарію	Ваговий коефіцієнт
FR1	Складний	4-7	10
FR2	Середній	≤ 3	5
FR3	Середній	4-7	10
FR4	Середній	≤ 3	5
FR5	Середній	4-7	10
FR6	Середній	4-7	10
FR7	Складний	4-7	10
FR8	Складний	4-7	10
FR9	Простий	≤ 3	5
Визначення UUCP: $A = 1 + 3 + 2 = 6$; $UC = 10 + 5 + 10 + 5 + 10 + 10 + 10 + 10 +$			

$$5 = 75; \text{UUCP} = A + UC = 6 + 75 = 81$$

Таблиця 11 – Вагові показники прецедентів

3. Визначення технічної складності проекту

Показник	Опис показника	Вага
T1	Распределенная система	1
T2	Высокая производительность (пропускная способность)	3
T3	Работа конечных пользователей в режиме онлайн	2
T4	Сложная обработка данных	1
T5	Повторное использование кода	0,5
T6	Простота установки	0
T7	Простота использования	3
T8	Переносимость	2
T9	Простота внесения изменений	2
T10	Параллелизм	1
T11	Специальные требования к безопасности	3
T12	Непосредственный доступ к системе со стороны внешних пользователей	1
T13	Специальные требования к обучению пользователей	0,5
Визначення TCF: $TCF = 0,6 + (0,01 * (ST_i * \text{Вага}_i)) = 0,6 + (0,01 * (37)) = 0,97$		

Таблиця 12 – Визначення технічної складності проекту

4. Визначення рівня кваліфікації розробників

Показник	Опис показника	Вага
F1	Знакомство с технологией	1,5
F2	Опыт разработки приложений	1
F3	Опыт использования объектно-ориентированного подхода	2
F4	Наличие ведущего аналитика	3
F5	Мотивация	4
F6	Стабильность требований	1
F7	Частичная занятость	2
F8	Сложные языки программирования	1
Визначення EF: $EF = 1,4 + (-0,03 * (SF_i * Вага_i)) = 1,4 + (-0,03 * (21)) = 0,77$		

Таблиця 13 – Визначення рівня кваліфікації розробників

5. Визначення UCP

$$UCP = UUCP * TCF * EF$$

$$UUCP = 81$$

$$TCF = 0,97$$

$$EF = 0,77$$

$$UCP = 81 * 0,97 * 0,77 = 60,4989$$

6. Оцінка трудомісткості проекту

$$\text{Трудомісткість} = UCP * 20 = 1\,209,978 \text{ люд\год}$$

2.3.2 Визначення дерева робіт з розробки ПП

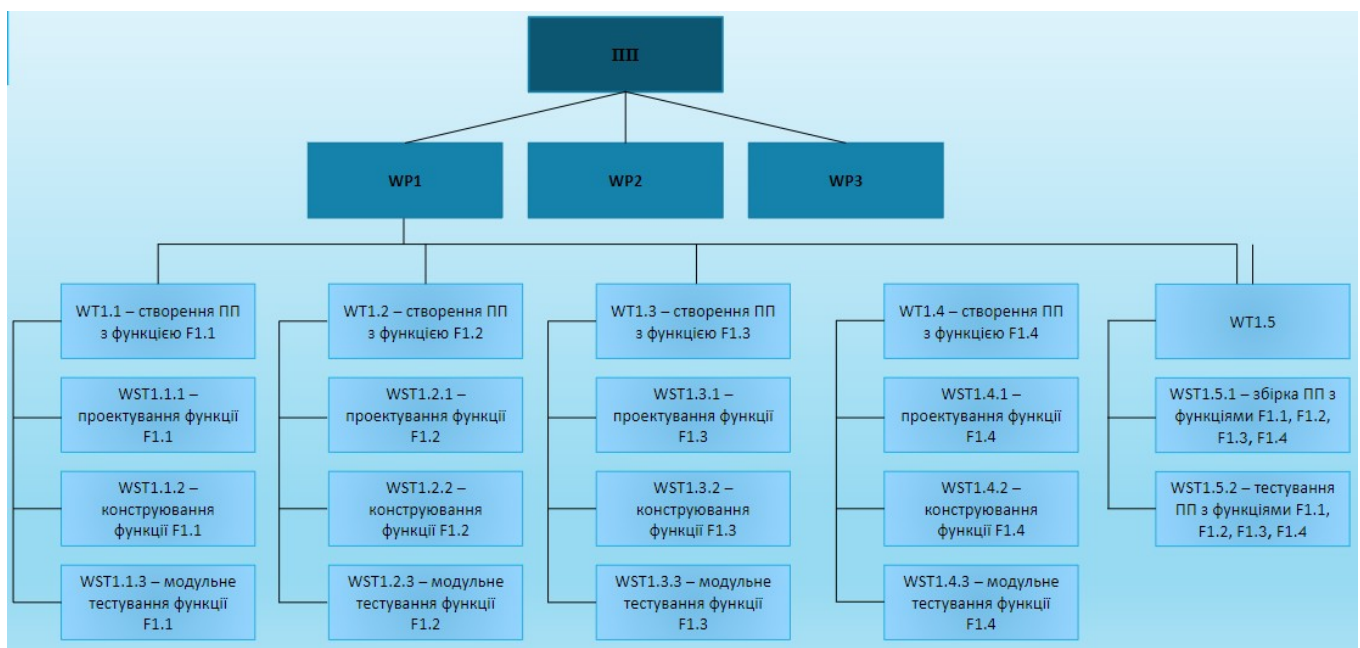


Рис. 7 – WBS дерево робіт

Ідентифікатор	Хто виконував
1.1*	Мудрик В. С.
1.2*	Мудрик В. С.
1.3*	Глаголев Р. Ю.
1.4*	Глаголев Р. Ю.

1.5*	Глаголев Р. Ю.
------	----------------

Таблиця 14 –Опис підзадач із закріпленням виконавців

2.3.3 Графік робіт з розробки ПП

2.3.3.1 Таблиця з графіком робіт

WST	Дата початку	Дні	Дата завершення	Виконавець
1.1.1	03.10.2020	1	04.10.2020	Мудрик В. С.
1.1.2	03.10.2020	2	05.10.2020	Мудрик В. С.
1.1.3	04.10.2020	1	05.10.2020	Мудрик В. С.
1.2.1	05.10.2020	2	07.10.2020	Мудрик В. С.
1.2.2	05.10.2020	2	07.10.2020	Мудрик В. С.
1.2.3	06.10.2020	1	07.10.2020	Мудрик В. С.
1.3.1	07.10.2020	2	09.10.2020	Глаголев Р. Ю.
1.3.2	07.10.2020	2	09.10.2020	Глаголев Р. Ю.
1.3.3	09.10.2020	1	10.10.2020	Глаголев Р. Ю.
1.4.1	09.10.2020	1	10.10.2020	Глаголев Р. Ю.
1.4.2	09.10.2020	2	11.10.2020	Глаголев Р. Ю.
1.4.3	11.10.2020	1	12.10.2020	Глаголев Р. Ю.
1.5.1	12.10.2020	1	13.10.2020	Глаголев Р. Ю.
1.5.2	13.10.2020	1	14.10.2020	Глаголев Р. Ю.

Рис. 8 – Графік робіт

2.3.3.2 Діаграма Ганта

WST	Дата початку	Дні	Дата завершення	Виконавець	03 10	04 10	05 10	06 10	07 10	09 10	10 10	11 10	12 10	13 10	14 10
1.1.1	03.10.2020	1	04.10.2020	Мудрик В. С.											
1.1.2	03.10.2020	2	05.10.2020	Мудрик В. С.											
1.1.3	04.10.2020	1	05.10.2020	Мудрик В. С.											
1.2.1	05.10.2020	2	07.10.2020	Мудрик В. С.											
1.2.2	05.10.2020	2	07.10.2020	Мудрик В. С.											
1.2.3	06.10.2020	1	07.10.2020	Мудрик В. С.											
1.3.1	07.10.2020	2	09.10.2020	Глаголев Р. Ю.											
1.3.2	07.10.2020	2	09.10.2020	Глаголев Р. Ю.											
1.3.3	09.10.2020	1	10.10.2020	Глаголев Р. Ю.											
1.4.1	09.10.2020	1	10.10.2020	Глаголев Р. Ю.											
1.4.2	09.10.2020	2	11.10.2020	Глаголев Р. Ю.											
1.4.3	11.10.2020	1	12.10.2020	Глаголев Р. Ю.											
1.5.1	12.10.2020	1	13.10.2020	Глаголев Р. Ю.											
1.5.2	13.10.2020	1	14.10.2020	Глаголев Р. Ю.											

Рис. 9 – Діаграма Ганта

3 ПРОЕКТУВАННЯ ПП

3.1 Концептуальне та логічне проектування структур даних ПП

3.1.1 Концептуальне проектування на основі UML-діаграми концептуальних класів

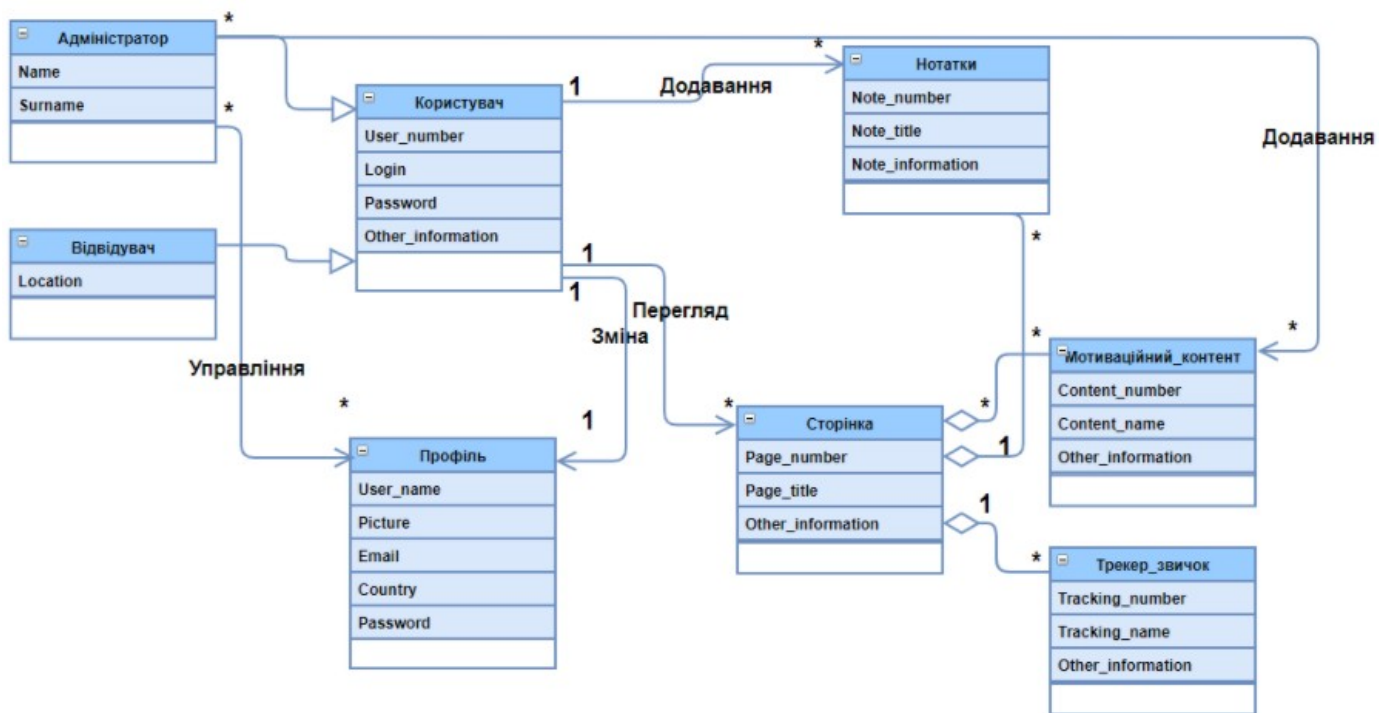


Рис. 10 – UML-діаграма концептуальних класів

3.1.2 Логічне проектування структур даних

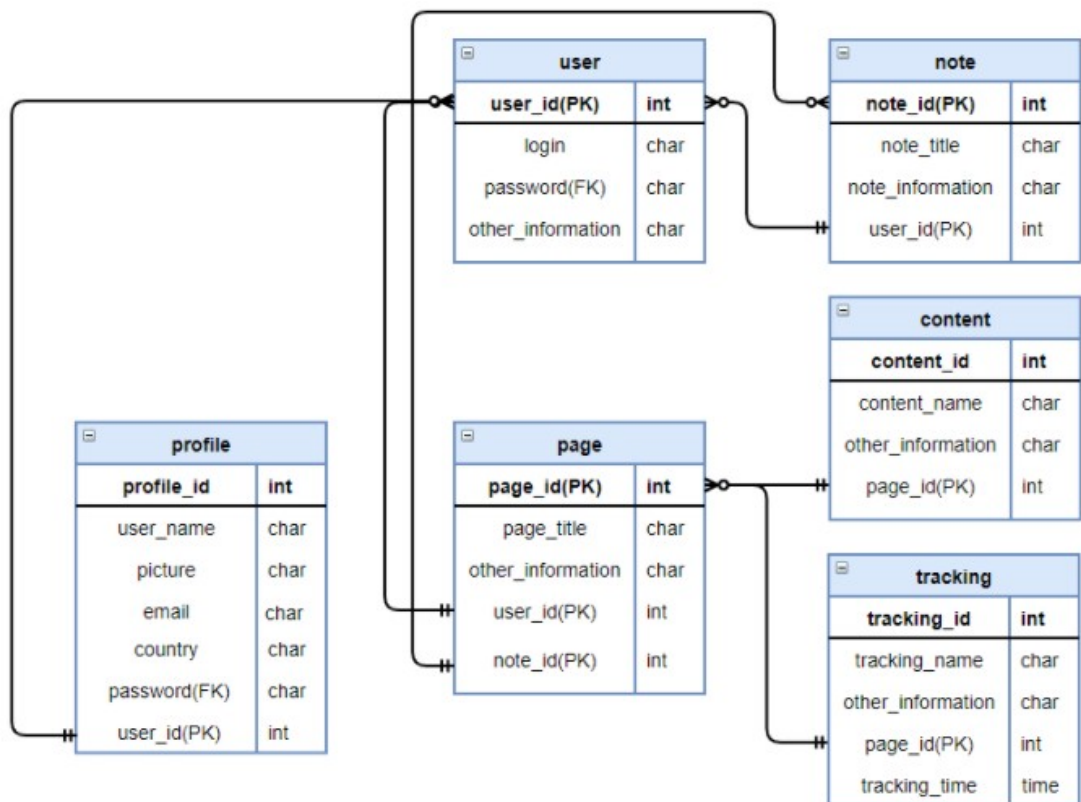


Рис. 11 – Структура БД

3.2 Проектування програмних класів

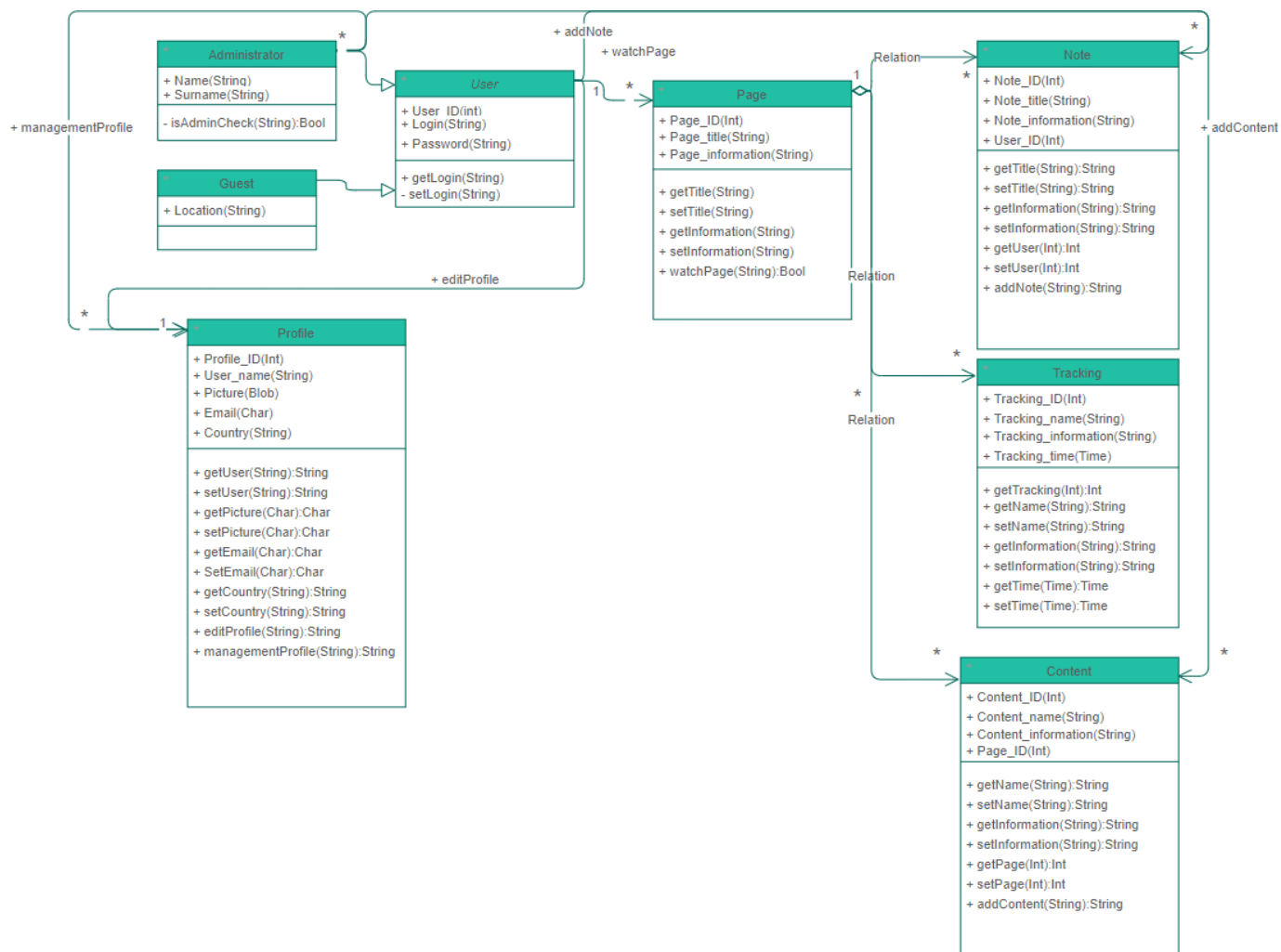


Рис. 12 – Проектування програмних класів

3.3 Проектування алгоритмів роботи методів програмних класів

@startuml

title signup()

start

:Вивід на екран відвідувача полів даних для реєстрації;

if (Дані введені не в усі обов'язкові поля або вказані невірно) then (Так);

:Вивід помилки реєстрації;

stop

else (Ні);

if (Немає підключення до бази даних) then (Так);


```

:Вивід помилки реєстрації;
stop
else (Hi);
:Введення в базу даних нового облікового запису користувача;
:Висновок користувачеві повідомлення про вдалу реєстрацію;
:Перенаправлення користувача на головну сторінку;
stop
@enduml

```



Рис. 13 – UML діаграма для Signup

```

@startuml
title editingprofile()
start
:Завантаження шаблону сторінки редагування профілю;
if (Дані змінено невірно або введені не в усі поля) then (Так);
:Вивід помилки редагування;
stop
else (Ні);
if ( Немає підключення до бази даних ) then (Так);

```

```

:Вивід повідомлення про помилку;
stop
else (Hi);
:Введення в базу даних нових параметрів облікового запису
користувача;
:Вивід користувачеві повідомлення про вдалу зміну даних;
:Перенаправлення користувача на попередню сторінку;
stop
@enduml

```

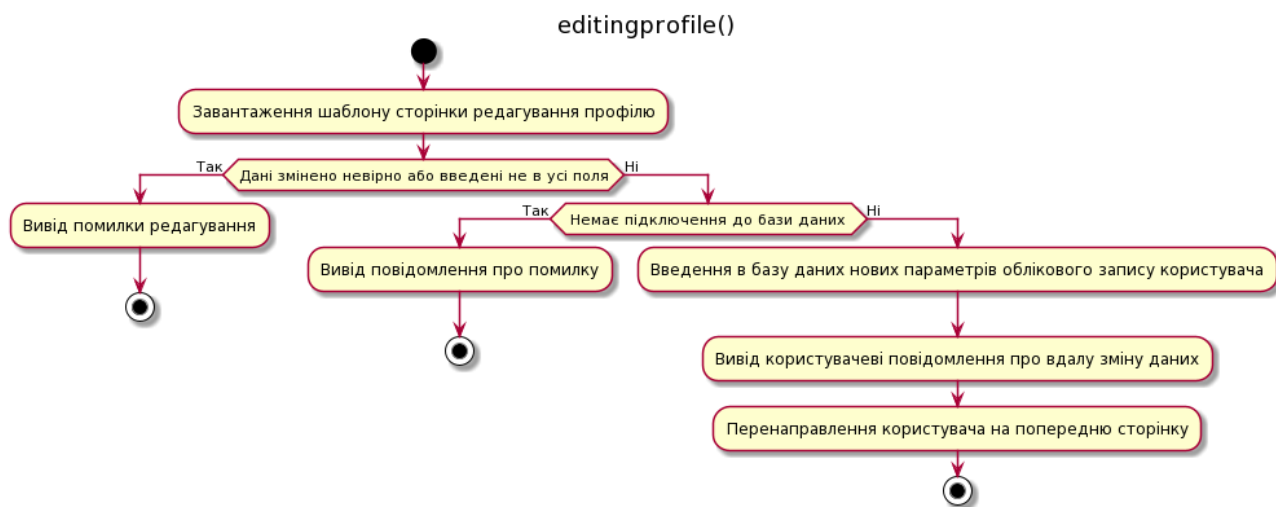


Рис. 14 – UML діаграма для Editingprofile

```

@startuml
title login()
start
:Введення даних для ідентифікації і аутентифікації користувача;
:Отримання даних з глобального масиву;
if(Якщо немає підключення до бази даних) then (Так);
:Висновок повідомлення про помилки;
stop
else (Hi);
if ( У таблиці користувачі не існує запису з заданими параметрами ) then
(Так);

```

```

:Вивід повідомлення про неіснуючого користувача;
stop
else (Hi);
:Створення в глобальному масиві поля залогіневшогося користувача і
присвоювання йому значення які були отримані з глобального масиву;
:Перенаправлення користувача на головну сторінку сайту;
stop
@enduml

```

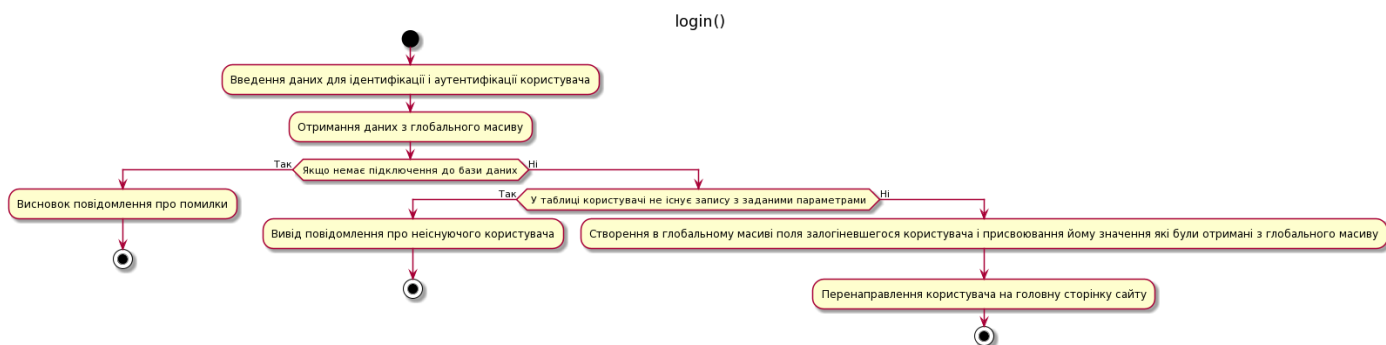


Рис. 15 – UML діаграма для Login

```

@startuml
title contentAdd()
start
repeat
:Завантаження шаблону сторінки для додавання нових записів;
:Додавання заголовка і основного змісту запису;
if( Якщо немає з'єднання з базою даних) then (Так);
:Вивід повідомлення про помилку;
stop
else (Hi);
:Аналіз існуючих записів на сторінці;
:Перевірка на існування подібного запису;
if(Така запись вже була створена раніше) then (Так);

```

```

        :Вивід повідомлення про помилку;
    else (Hi);
        :Збереження в базу даних;
        :Інформація про успішне додавання запису;
    stop
@enduml

```

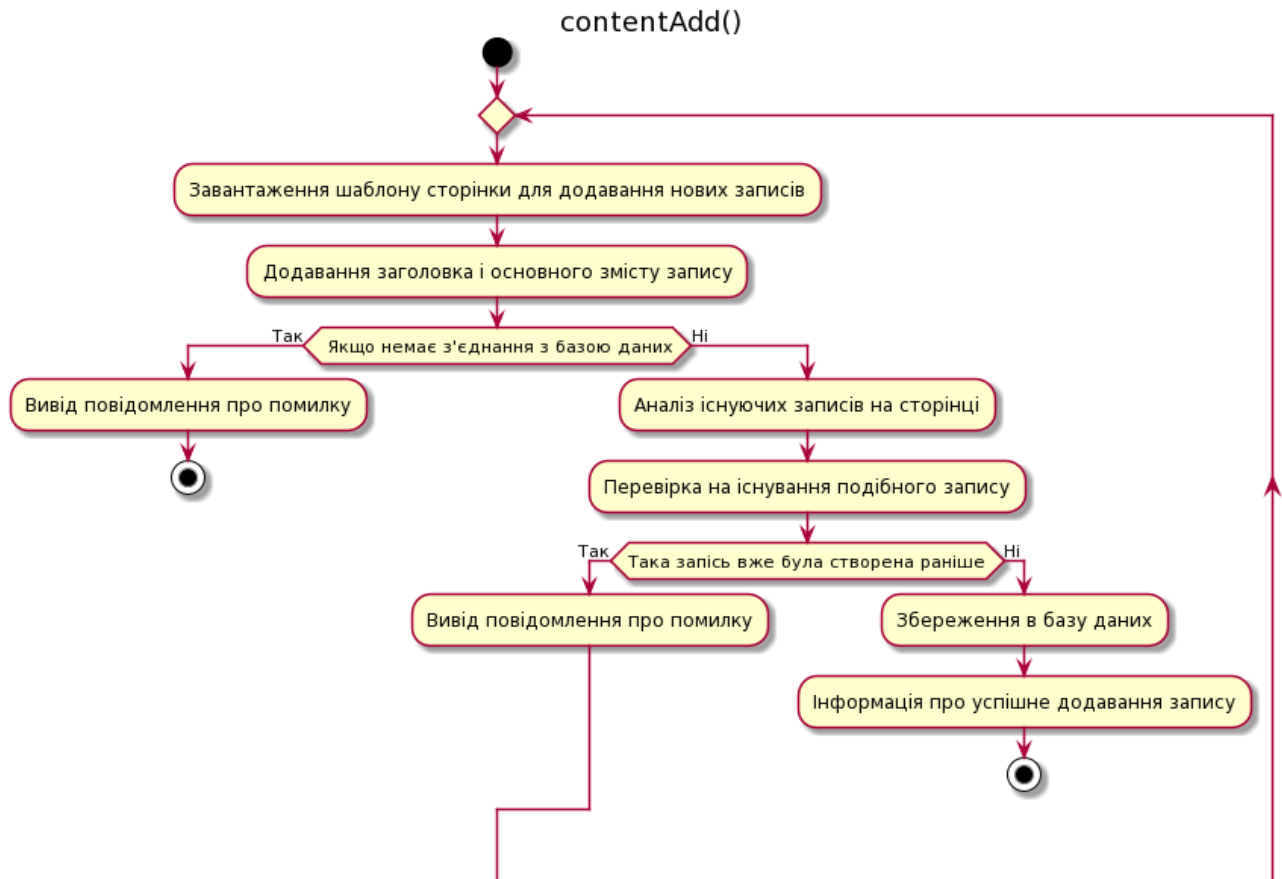


Рис. 16 – UML діаграма для Contentadd

№	Назва методу	Глаголев Р.Ю.	Мудрик В.С.
1	Signup.php	+	
2	Showinfo.php		+
3	Editingprofile.php	+	
4	Support.php		+

5	Login.php	+	
6	Logout.php		+
7	Contentadd.php	+	
8	Search.php		+

Таблиця -15 Таблиця учасників

3.4Проектування тестових наборів методів програмних класів

Назва функції	Номер тесту	Опис значень вхідних даних	Опис очікуваних значень результату
Signup()	1	user_name = user01 password =qwertyu123	Користувач успішно зареєстрований
Signup()	2	user_name = 01user password =qwertyu123	Помилка: рядок user_name повинна починатися з латинської літери
Signup()	3	user_name = 01uuussseeerrr01 password =qwertyu123	Помилка: перевищення розміру логіна користувача
Signup()	4	user_name = user01 password =1234	Помилка: довжина рядка password повинна бути не менше 8 символів
Login()	1	user_name = “правильне значення” password = “правильне значення”	Користувач успішно увійшов в акаунт
Login()	2	user_name = “правильне значення логіна” password = “правильне значення пароля”	Помилка: логін або пароль введено невірно
Login()	3	user_name = “неправильне значення логіна”	Помилка: логін або пароль введено

		password = “неправильне значення пароля”	невірно
Contentadd()	1	title = “Чи не усувайте погані звички - поміняйте їх” text = “вводимо будь-який текст”	Запис успішно додано
Contentadd()	2	title = “вводимо неприпустимі символи” text = “вводимо будь-який текст”	Помилка: будь ласка, введіть коректну назву статті
Contentadd()	3	title = “вводимо більше 50 символів” text = “вводимо будь-який текст”	Помилка: назва занадто довга, будь ласка скоротіть його
Contentadd()	4	title = “залишаємо поле порожнім” text = “залишаємо поле порожнім”	Помилка: запис не може бути порожнім, будь ласка внесіть дані
SetCountry()	1	“Вибираємо будь-яку країну їхнього списку”	Країна успішно змінена
SetNickname()	2	nickname = firstuser01	Нові дані успішно збережені
SetNickname()	3	nickname = “вводимо неприпустимі символи”	Помилка: нікнейм містить неприпустимі символи, будь ласка спробуйте ще раз
SetNickname()	4	nickname = “вводимо більше 15 символів”	Помилка: нікнейм занадто довгий
SetNickname()	5	“Залишаємо поле порожнім і зберігаємо”	Помилка: поле нікнейму не може бути порожнім, будь ласка внесіть дані
SetEmail()	6	email = namesurname@gmail.com	Нова електронна пошта успішно

			збережена
SetEmail()	7	“Залишаємо поле порожнім і зберігаємо”	Помилка: поле не може бути порожнім, будь ласка вкажіть пошту

Таблиця 16– Тестові набори

№	Назва методу	Глаголев Р.Ю.	Мудрик В.С.
1	Signup.php	+	
2	Showinfo.php		+
3	Editingprofile.php	+	
4	Support.php		+
5	Login.php	+	
6	Logout.php		+
7	Contentadd.php	+	
8	Search.php		+

Таблиця 17 - Таблиця учасників

4 КОНСТРУЮВАННЯ ПП

4.1 Особливості конструювання структур даних

4.1.1 Особливості інсталяції та роботи з СУБД

Будь-якому починаючому розробнику WordPress необхідно ознайомитися з двома речами: з технічною документацією WordPress і з особливостями архітектури бази даних, як мінімум, для того щоб вміти створювати складні

WordPress шаблони і простенькі плагіни, сьогодні ми познайомимося з архітектурою бази даних сайту на WordPress.

Перше, що слід сказати: як системи управління базами даних WordPress використовує MySQL сервер версії 5.0.15 і вище. Як вже говорилося: кінцевому користувачеві інформація про БД WordPress навряд чи колись стане в нагоді, багатьом розробникам для створення тем та плагінів WordPress буде досить набору функцій WordPress для роботи з базами даних, але іноді бувають ситуації, коли API WordPress недостатньо.

Тобто, іноді виникає потреба звертатися до баз даних безпосередньо. Тому потрібна інформація про те, як WordPress зберігає свої дані і які є залежності і обмеження між таблицями бази даних WordPress. Щоб звернутися до баз даних WordPress безпосередньо, слід використовувати WPDB клас

4.1.2 Особливості створення структур даних

Функція dbDelta ()

Вбудована функція dbDelta () дозволяє створювати таблиці в базі даних WordPress, і вносити зміни в їх структуру. Але перед створенням нової таблиці, необхідно визначитися з її найменуванням і кодуванням.

Назва нової таблиці повинно мати той же префікс, який використовується ядром WordPress (за замовчуванням wp_) і додатковий префікс для нашого плагіна або проекту, наприклад: wp_my_table_name. Префікс wp_ можна отримати за допомогою методу get_blog_prefix () глобального об'єкта \$wpdb, а за допомогою властивостей charset і collate можна визначити використовувану кодування:

```
global $wpdb;
```

```
$table_name = $wpdb->get_blog_prefix() . 'my_products';
```



```
$charset_collate = "DEFAULT CHARACTER SET {$wpdb->charset} COLLATE  
{ $wpdb->collate}";
```

Далі за допомогою функції dbDelta () ми можемо створити нову таблицю:

```
require_once( ABSPATH . 'wp-admin/includes/upgrade.php' );
```

```
$sql = "CREATE TABLE {$table_name} (
```

```
    id int(11) unsigned NOT NULL auto_increment,
```

```
    name varchar(255) NOT NULL default "",
```

```
    price int(11) unsigned NOT NULL default '0',
```

```
    PRIMARY KEY (id),
```

```
    KEY price (price)
```

```
) {$charset_collate}";
```

```
// Создать таблицу.
```

```
dbDelta( $sql );
```

Функція dbDelta () відрізняється від простого MySQL запиту в базу даних WordPress. Вона розбиває структуру таблиці на частини і порівнює її з тією таблицею, яка вже існує. Це дозволяє dbDelta () вносити зміни в структуру таблиці, без необхідності видаляти і знову створювати таблицю.

Наприклад, щоб додати нову колонку в нашу таблицю, ми можемо змінити наш основний запит CREATE TABLE наступним чином:

```
$sql = "CREATE TABLE {$table_name} (
```

```
id int(11) unsigned NOT NULL auto_increment,  
  
name varchar(255) NOT NULL default "",  
  
price int(11) unsigned NOT NULL default '0',  
  
color varchar(255) NOT NULL default "",  
  
PRIMARY KEY (id),  
  
KEY price (price)  
  
) {$charset_collate};";
```

4.2 Особливості конструювання програмних модулів

4.2.1 Особливості роботи з інтегрованим середовищем розробки

При верстці сайту можна використовувати різні редактори коду або системи керування вмістом. Чому ми вибрали саме WordPress?

Основні переваги це:

- легкість поновлення
- професійні шаблони
- неймовірно потужний движок
- повний контроль і право власності

Але існують і недоліки, це:

- процес навчання
- технічне обслуговування

Що краще? WordPress або HTML?

Якщо в майбутньому оновлення сайту не настільки важливо, тоді в будь-якому випадку вибрати HTML сайт буде правильно. Так ви швидше за все досягнете своєї мети.

Якщо ж вам необхідно вільніше, контролювати свій сайт і не витратити даремно гроші щомісяця наймаючи розробників для додавання зображень, то WordPress очевидно в перевазі.

Використовуючи WordPress, ви можна легко додавати контент на сайт. легко створити будь-яку кількість сторінок, скільки забажаєте, в будь-який момент. Ви можете додати блог, щоб ваші клієнти дізнавалися про оновлення, або зробити власну розсилку, щоб залучити більше клієнтів. Замість того, щоб платити розробникам багато грошей за виконання простих завдань, ви можете робити це самостійно з такою ж швидкістю.

4.2.2 Особливості створення програмної структури з урахуванням спеціалізованого фреймворку

Це просунутий спосіб, він більш складний, але разом зі складністю він відкриває широкі можливості. За допомогою цього способу можна задати шаблон будь-якій сторінці, записи, категорії, будь публікації на сайті або взагалі групі будь-яких публікацій.

// фільтр передає змінну \$ template - шлях до файлу шаблону.

// Змінюючи цей шлях ми змінюємо файл шаблону.

```
add_filter( 'template_include', 'my_template' );
```

```
function my_template( $template ) {
```

```
# Аналог другого способу
```

```
// якщо це сторінка з складаючи portfolio, використовуємо файл шаблону page-portfolio.php
```

```
// використовуємо умовний тег is_page ()
```

```
if( is_page('portfolio') ){
```

```

        if ( $new_template = locate_template( array( 'page-portfolio.php' ) ) )
            return $new_template ;
    }

# Шаблон для групи рубрик
// цей приклад буде використовувати файл з папки теми tpl_special-cats.php,
// як шаблон для рубрик з ID 9, назвою "Без рубрики" і складаючи "php"
    if( is_category( array( 9, 'Без рубрики', 'php' ) ) ){
        return get_stylesheet_directory() . '/tpl_special-cats.php';
    }

# Шаблон для запису по ID
// файл шаблону розташований в папці плагіна /my-plugin/site-template.php
    global $post;
    if( $post->ID == 12 ){
        return wp_normalize_path( WP_PLUGIN_DIR ) . '/my-plugin/site-
template.php';
    }

# Шаблон для сторінок довільного типу "book"
// передбачається, що файл шаблону book-tpl.php лежить в папці теми
    global $post;
    if( $post->post_type == 'book' ){
        return get_stylesheet_directory() . '/book-tpl.php';
    }
    return $template;
}

```

4.2.3 Особливості створення програмних модулів

Имя	Дата изменения	Тип	Размер
wp-admin	12.10.2020 18:06	Папка с файлами	
wp-content	27.10.2020 21:01	Папка с файлами	
wp-includes	12.10.2020 18:06	Папка с файлами	
.htaccess	12.10.2020 18:19	Файл "HTACCESS"	1 КБ
index.php	06.02.2020 08:33	Исходный файл ...	1 КБ
license.txt	12.10.2020 18:21	Текстовый докум...	20 КБ
readme.html	12.10.2020 18:21	Firefox HTML Doc...	11 КБ
wp-activate.php	28.07.2020 20:20	Исходный файл ...	7 КБ
wp-blog-header.php	06.02.2020 08:33	Исходный файл ...	1 КБ
wp-comments-post.php	23.07.2020 03:52	Исходный файл ...	3 КБ
wp-config.php	12.10.2020 18:13	Исходный файл ...	3 КБ
wp-config-sample.php	12.10.2020 18:21	Исходный файл ...	5 КБ
wp-cron.php	06.02.2020 08:33	Исходный файл ...	4 КБ
wp-links-opml.php	06.02.2020 08:33	Исходный файл ...	3 КБ
wp-load.php	06.02.2020 08:33	Исходный файл ...	4 КБ
wp-login.php	07.07.2020 06:59	Исходный файл ...	48 КБ
wp-mail.php	14.04.2020 14:32	Исходный файл ...	9 КБ
wp-settings.php	06.07.2020 13:50	Исходный файл ...	20 КБ
wp-signup.php	24.07.2020 00:11	Исходный файл ...	31 КБ
wp-trackback.php	06.02.2020 08:33	Исходный файл ...	5 КБ
xmlrpc.php	08.06.2020 22:55	Исходный файл ...	4 КБ

Рис. 17 - Файлова структура

При розробці сайту було використано Open Server Panel. Це портативна серверна платформа і програмне середовище, створена спеціально для веб-розробників з урахуванням їх рекомендацій і побажань.

Всі програмні модулі знаходяться на локальному диску і в разі необхідності запускаються з відкритим портом в глобальну мережу Інтернет.

4.2.4 Особливості розробки алгоритмів методів програмних класів або процедур/функцій

Класичний клас

Це найбільш частий і зрозумілий метод. Використовується метод `__construct ()` для створення екземпляра класу. Але він же найбільш проблемний через те що багато програмісти можуть його неправильно готувати.

Приклад:

```

class MyPlugin1

{

    public function __construct()

    {

        add_action('plugins_loaded', array($this, 'loaded'));

    }

    public function loaded()

    {

        // do stuff

    }

}

```

Статичний клас

Цей підхід використовую найчастіше. За його гнучкість і можливість перемикання в інші стилі і підходи. Часто такі механіки можна зустріти в WooCommerce.

Приклад:

```

class My_Plugin2

{

    public static function init ()

    {

        add_action ('plugins_loaded', array (__ CLASS__, 'loaded'));

    }

}

```

```

public static function loaded ()

{

    // do stuff

}

}

My_Plugin2 :: init ();

```

Сінглтон

Часто застосовуються там де потрібно гарантовано мати один екземпляр об'єкта в системі без дублювання.

Плюс більш передбачувана і зрозуміла робота зі станом об'єкта ніж у Статичної підходу.

Бував два типу старту: через `init ()` або `instance ()`

Приклад:

```

class MyPlugin3

{

    private static $ ins;

    public static function init ()

    {

        add_action ('plugins_loaded', array (__ CLASS__, 'instance'));

    }

    public static function instance ()

    {

```

```

        is_null (self :: $ ins) && self :: $ ins = new self;

        return self :: $ ins;

    }

    private function __construct ()

    {

        add_action ('plugins_loaded', array ($ this, 'loaded'), 20);

    }

    public function loaded ()

    {

        // do stuff

    }

}

```

4.3 Тестування програмних модулів

Сценарій тестування функції реєстрації:

Передумови початку: користувач хоче зареєструватися

Сценарій:

- відвідувач переходить на сторінку реєстрації;
- потім вводить логін, нікнейм, країну і пошту;
- після чого вводить два рази пароль;
- потім користувач натискає кнопку реєстрації;
- підтверджує свої дії;
- останній зареєстрований учасник переходить на сторінку входу в аккаунт;

- вводить логін;
- вводить пароль;
- потім користувач натискає на кнопку входу;
- потім користувач може перейти в особистий кабінет і при бажанні може змінити свої дані.

Сценарій тестування функції редагування профілю:

Передумови початку: користувач хоче змінити особисті дані

Сценарій:

- користувач переходить в особистий кабінет
- заходить в розділ детальної інформації
- змінити логін він не може
- він змінює нікнейм, при бажанні
- вказує іншу країну при бажанні
- пошту змінити він теж не може
- потім підтверджує збереження
- проходить тест на бота
- оновлює сторінку

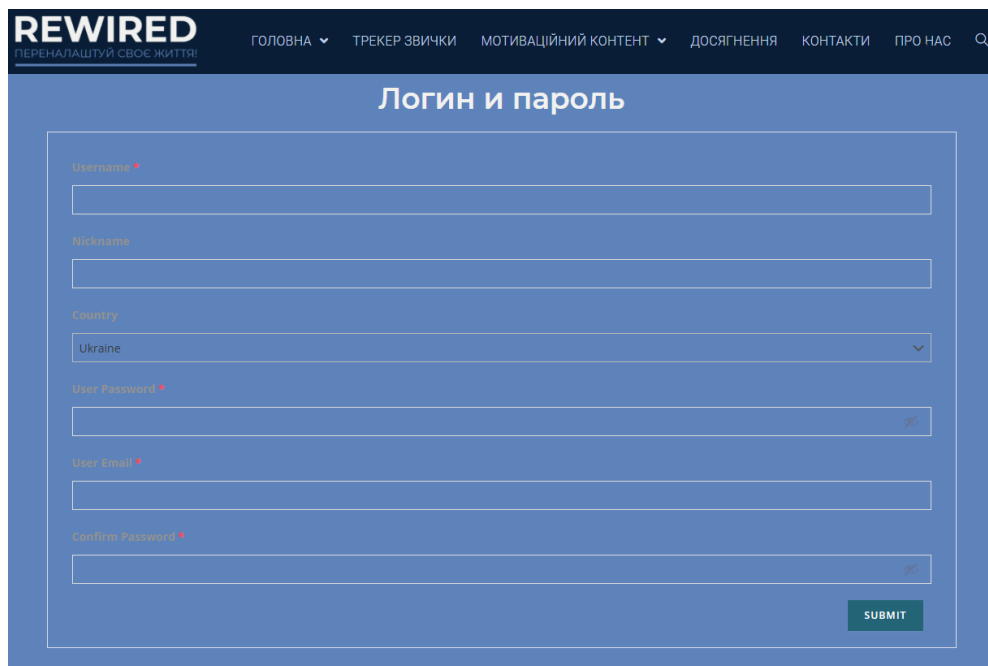
5 РОЗГОРТАННЯ ТА ВАЛІДАЦІЯ ПП

5.1 Інструкція з встановлення ПП

Основні кроки установки програмного продукту:

- 1) Клонувати репозиторій з сайтом
- 2) Зберегти всі дані зі сховищ в корінну доменну папку веб сервера
- 3) Запустити веб сервер
- 4) На веб сервері створити базу даних з відповідною назвою
- 5) Перейти на розташування веб серверу (домене ім'я:порт)

5.2 Інструкція з використання ПП



The image shows a registration form for the REWIRED website. The header is dark blue with the REWIRED logo and tagline 'ПЕРЕНАЛАШТУЙ СВОЄ ЖИТТЯ!' on the left, and navigation links (ГОЛОВНА, ТРЕКЕР ЗВИЧКИ, МОТИВАЦІЙНИЙ КОНТЕНТ, ДОСЯГНЕННЯ, КОНТАКТИ, ПРО НАС) and a search icon on the right. The main section has a blue background with the title 'Логин и пароль'. The form contains several input fields: Username, Nickname, Country (a dropdown menu with 'Ukraine' selected), User Password, User Email, and Confirm Password. Each field has a red asterisk indicating it is required. The User Password and Confirm Password fields include a strength indicator icon. A green SUBMIT button is located at the bottom right of the form.

REWIRED
ПЕРЕНАЛАШТУЙ СВОЄ ЖИТТЯ!

ГОЛОВНА ▼ ТРЕКЕР ЗВИЧКИ МОТИВАЦІЙНИЙ КОНТЕНТ ▼ ДОСЯГНЕННЯ КОНТАКТИ ПРО НАС 🔍

Логин и пароль

Username *

Nickname

Country
Ukraine ▼

User Password *

User Email *

Confirm Password *

SUBMIT

Рис. 22 – Регестрація

Profile Detail



Max size: 50 MB

Upload your new profile image.

REMOVE

Username *

Testuser01

Nickname

RomaNN

Country

Italy



User Email *

r_q_1999@mail.ru

SAVE CHANGES

Рис. 18 – Редагування профілю як результат успішної реєстрації

10 НАЙШКІДЛИВИШИХ ЗВИЧОК ЛЮДИНИ

Вредные привычки мешают человеку успешно реализовать себя как личность. Большинство таких привычек негативно влияют либо на человека с такой привычкой, либо на окружающих его людей. В любом случае нужно постараться как можно быстрее и эффективнее справиться с этой проблемой, что бы она больше никогда не мешала ни вам, не окружающим. В этом рейтинге мы расскажем о самых вредных привычках и зависимостях.

Куріння

Про те, що куріння є шкідливим для здоров'я, знають усі. Однак кожен курець думає, що наслідки куріння його не торкнуться, і він живе сьогоднішнім днем, не думаючи про хвороби, які неминуче з'являться у нього через 10-20 років. Відомо, що за кожну шкідливу звичку рано чи пізно доведеться розплачуватися своїм здоров'ям. З курінням пов'язано до 90% смертності від раку легень, 75% від бронхіту і 25% від ішемічної хвороби серця серед чоловіків у віці до 65 років. Куріння або пасивне вдихання тютюнового диму може послужити причиною безпліддя у жінок. Атрофія і руйнування білої речовини головного та спинного мозку при розсіяному склерозі більш виражена у пацієнтів, які курили хоча б 6 місяців протягом життя в порівнянні з ніколи не курившими хворими.



Залежність від тютюнопаління може бути як психологічної, так і фізичної. При психологічній залежності людина тягнеться за сигаретою, коли перебуває в кращій компанії, або в стані стресу, нервової напруги, для стимуляції розумової діяльності. При фізичній залежності вимога організмом нікотинової дози так сильно, що вся увага кращого зосереджується на пошуку сигарети, ідея куріння стає настільки нав'язливою, що більшість інших потреб йдуть на другий план. З'являється неможливість сконцентруватися на чому-небудь, крім сигарети, може наступити апатія, небажання що-небудь робити.

Алкоголь

Алкоголь присутній в житті майже кожної людини. Хтось п'є лише у свята, хтось любить відпочити з порцією алкоголю у вихідні, а хтось зловживає спиртним постійно. Під дією етанолу, який знаходиться в алкогольних напоях валитися все, в першу чергу — нервова і серцево-судинна системи. Слабкі м'язи, тромби в судинах, діабет, висохлі головний мозок, роздута печінка, ослаблені нирки, імпотенція, депресія, виразка шлунка — це лише частковий перелік того, що ви можете отримати від регулярного вживання пива чи чогось міцнішого. Будь-яка порція алкоголю — це удар по інтелекту, по здоров'ю, по майбутньому.



Пляшка горілки, випита за годину, може вас вбити на місці, в прямому сенсі. Наступного разу, перед тим як пити 100 грам, уявіть свій організм, повільно вмираючий під впливом етанолу, в той час як ви будете веселитися. Уявіть, що ваші клітини повільно задихаються, що мозок, рятуючись, блокує безліч мозкових центрів, через що з'являється безладна мова, порушення просторового відчуття, порушена координація рухів і провали в пам'яті. Уявіть, як згущується ваша кров, утворюють смертельно небезпечні тромби, як зашкалює рівень цукру в крові, як гинуть структури мозку, що відповідають за інтелект і кмітливість, як алкоголь пропалює стінки шлунка, утворюючи незагойні виразки.

Наркотики

Вживання наркотиків призводить до важких порушень, в першу чергу, психічних і фізичних функцій організму. У сучасному суспільстві мало хто не знає про шкоду наркотиків, але, не дивлячись на це, вони як і раніше залучають людей, стаючи згубними для багатьох. У вживають наркотики



Рис. 19 – Перегляд статей на сайті

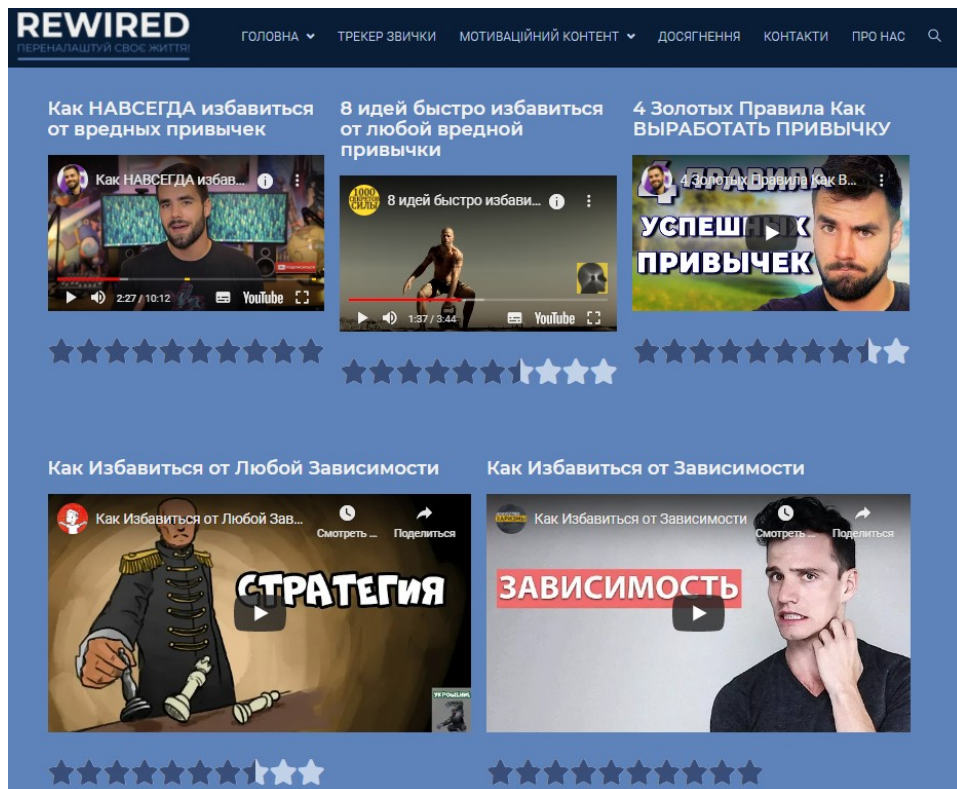


Рис. 20 – Перегляд роликів на сайті

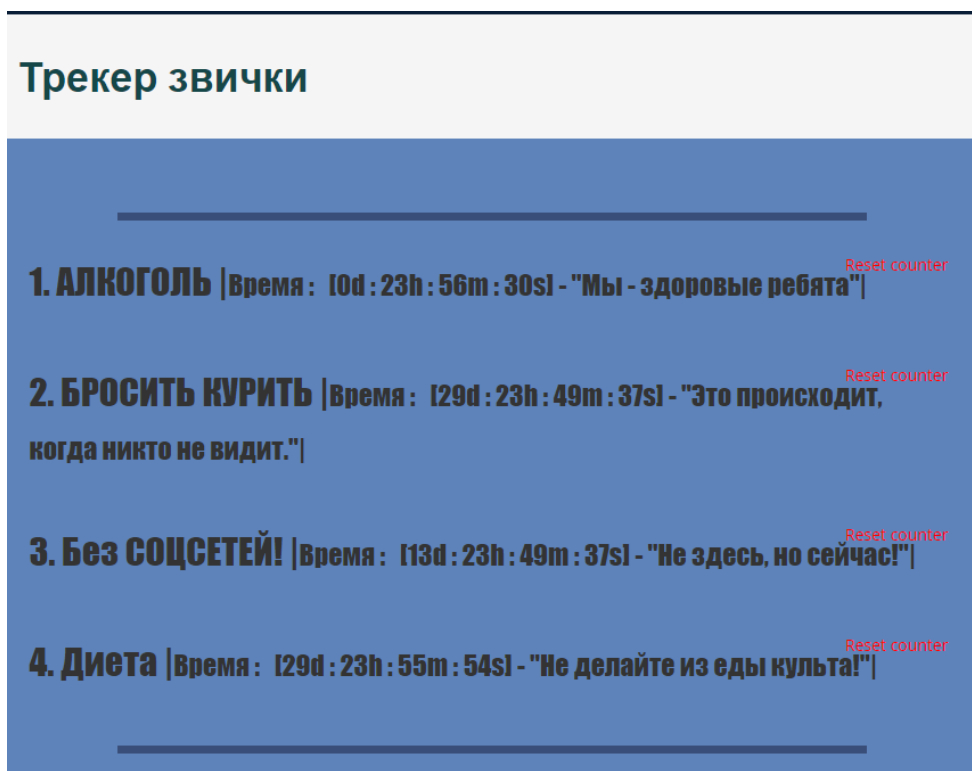


Рис. 21 Работа таймера звички

5.3 Результати валідації ПП

Мета створення ПП була підвищення рівня доступності програмного продукту для всіх соціальних рівнів и допомога користувачам в налагодженні життєвого ритму, саморозвитку і особистісному зростанні.

Формула розрахунку доступності така:

$$\text{Availability} = (\text{AST} - \text{DT}) / \text{AST} \times 100 = \text{Service or Component Availability (\%)}$$

де AST (agreed service time) - узгоджений час надання послуги;

DT (actual downtime during agreed service time) - фактичний час, коли послуга була недоступна протягом узгодженого часу її надання.

Поки що можна лише припустити, що ПП пройшов валідацію і буде виконувати свої функції, так як програмний продукт вийшов непоганий, але поки не вийшов на ринок.

ВИСНОВКИ

Як результат, було отримано непоганий програмний продукт, який допоможе людям подолати свої залежності і страхи, не на самоті, а разом.

Переналаштуй своє життя - Rewired

В процесі створення програмного продукту виникли такі труднощі:

- 1) обмеженість в часі;
- 2) складність розробки ПП;
- 3) недостатні знання де-яких інструментів.

ДЖЕРЕЛА

1. <https://wpmag.ru/2014/custom-tables-dbdelta/>
2. https://wp-kama.ru/id_5177/3-sposoba-sozdat-shablon-stranitsyi.html
3. <https://wpcraft.ru/blog/3-podhoda-k-programmirovaniyu-klassov-v-wordpress/>