

Міністерство освіти і науки України
Державний університет «Одеська політехніка»
Інститут комп'ютерних систем
Кафедра інформаційних систем

КУРСОВА РОБОТА

з дисципліни «Технології створення програмних продуктів»

за темою

«Розробка мобільного додатку для продажу ігрових ресурсів»

Пояснювальна записка (комплексної роботи)

Виконав:
студент 3-го курсу
групи АІ-194
Шинкаренко А.В.
Перевірив:
Блажко О.А.

Одеса-2021

Анотація

В курсовій роботі розглядається процес створення програмного продукту «Розробка мобільного додатку для продажу ігрових ресурсів» на основних етапах: визначення вимог до програмного продукту, планування процесів розробки, проектування, конструювання, верифікація, розгортання та валідація програмного продукту.

Робота виконувалась студентом групи AI-194 Шинкаренко А.В..

Робота пов'язана з такими матеріальними потребами споживача як неможливість отримати достатню кількість знань про гру, щоб поліпшити навички . Аналіз вказаних потреб визначив інформаційну потребу - отримати знання для поліпшення ігрових навичок, для вдосконалення свого рівня гри; отримати інформацію для кращого розуміння навичок та вмінь ігрових персонажів/зброї/предметів тощо.

При визначенні ступеня готовності існуючих програмних продуктів до вирішення інформаційної потреби проаналізовано наступні програмні продукти: G2A, G2G, FunPay, Gym Keeper.

Поточну версію пояснювальної записки до результатів роботи розміщено на *GitHub*-репозиторії за адресою: <https://github.com/MieliAries/WatchAndLearn>

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

	стор.
1 Вимоги до програмного продукту	8
1.1 Визначення потреб споживача	8
1.1.1 Ієрархія потреб споживача	8
1.1.2 Деталізація матеріальної потреби	9
1.2 Бізнес-вимоги до програмного продукту	10
1.2.1 Опис проблеми споживача	10
1.2.1.1 Концептуальний опис проблеми споживача	10
1.2.1.2 Опис цільової групи споживача	10
1.2.1.3 Метричний опис проблеми споживача	10
1.2.2 Мета створення програмного продукту	11
1.2.2.1 Проблемний аналіз існуючих програмних продуктів	11
1.2.2.2 Мета створення програмного продукту	11
1.2.3 Назва програмного продукту	11
1.2.3.1 Гасло програмного продукту	12
1.2.3.2 Логотип програмного продукту	12
1.3 Вимоги користувача до програмного продукту	12
1.3.1 Пригодницька історія користувача програмного продукту	12
1.3.2 Історія користувача програмного продукту	13
1.3.3 Діаграма прецедентів програмного продукту	14
1.3.4 Сценаріїв використання прецедентів програмного продукту	14
1.4 Функціональні вимоги до програмного продукту	18
1.4.1. Багаторівнева класифікація функціональних вимог	18

1.4.2 Функціональний аналіз існуючих програмних продуктів	19
1.5 Нефункціональні вимоги до програмного продукту	20
1.5.1 Опис зовнішніх інтерфейсів	20
1.5.1.1 Опис інтерфейсів користувача	20
1.5.1.1.1 Опис INPUT-інтерфейсів користувача	20
1.5.1.1.2 Опис OUTPUT-інтерфейсів користувача	21
1.5.1.2 Опис інтерфейсу із зовнішніми пристроями	24
1.5.1.3 Опис програмних інтерфейсів	24
1.5.1.4 Опис інтерфейсів передачі інформації	24
1.5.1.5 Опис атрибутів продуктивності	24
2 Планування процесу розробки програмного продукту	25
2.1 Планування ітерацій розробки програмного продукту	25
2.2 Концептуальний опис архітектури програмного продукту	26
2.3 План розробки програмного продукту	27
2.3.1 Оцінка трудомісткості розробки програмного продукту	27
2.3.2 Визначення дерева робіт з розробки програмного продукту	29
2.3.3 Графік робіт з розробки програмного продукту	31
2.3.3.1 Таблиця з графіком робіт	31
2.3.3.2 Діаграма Ганта	32
3 Проектування програмного продукту	33
3.1 Концептуальне та логічне проектування структур даних програмного продукту	33

3.1.1 Концептуальне проектування на основі <i>UML</i> -діаграми концептуальних класів	33
3.1.2 Логічне проектування структур даних	34
3.2 Проектування програмних класів	34
3.3 Проектування алгоритмів роботи методів програмних класів	35
3.4 Проектування тестових сценаріїв верифікації роботи програмних модулів	37
3.4.1 Проектування тестових сценаріїв верифікації функціональних вимог	37
3.4.2 Проектування тестових сценаріїв верифікації нефункціональних вимог	44
3.4.3 Створення матриці відповідності вимог до програмного продукту	48
4 Конструювання програмного продукту	49
4.1 Особливості конструювання структур даних	50
4.2 Особливості конструювання програмних модулів	50
4.2.1 Конструювання програмної структури з урахуванням спеціалізованого <i>Framework</i> для <i>FrontEnd</i> -компонент архітектури	50
4.2.2 Конструювання алгоритмів методів програмних класів або процедур/функцій	50
4.2.3 Особливості використання спеціалізованих програмних бібліотек та API	54
5 Верифікація програмного продукту	56
5.1 Тестування апаратно-програмних інтерфейсів програмного продукту	56

5.2 Тестування інтерфейсу користувача програмного продукту	56
5.3 Тестування часу реакції програмного продукту на дії користувача	60
6 Розгортання та валідація програмного продукту	61
6.1 Інструкція з встановлення системного програмного забезпечення	61
6.2 Інструкція з використання програмного продукту	62
6.3 Результати валідації програмного продукту	66
Висновки	68

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.

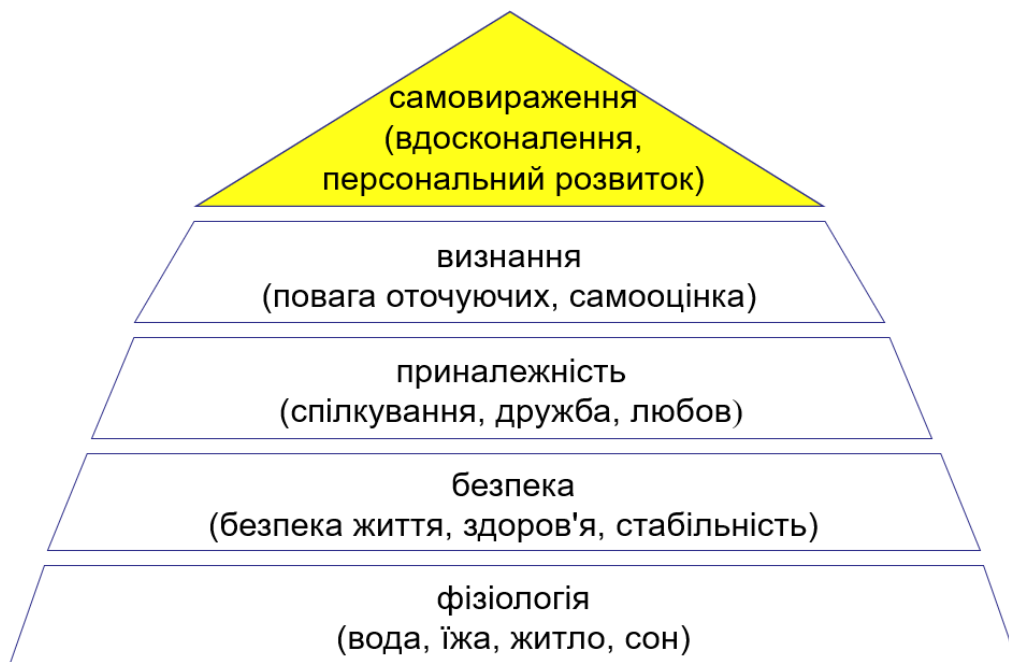


Рис. 1.1 – Приклад ієрархії потреби споживача

1.1.2 Деталізація матеріальної потреби

На рисунках 1.2 і 1.3 продемонстровано деталізацію матеріальної потреби, розбиту на 2 частини для кращого розуміння

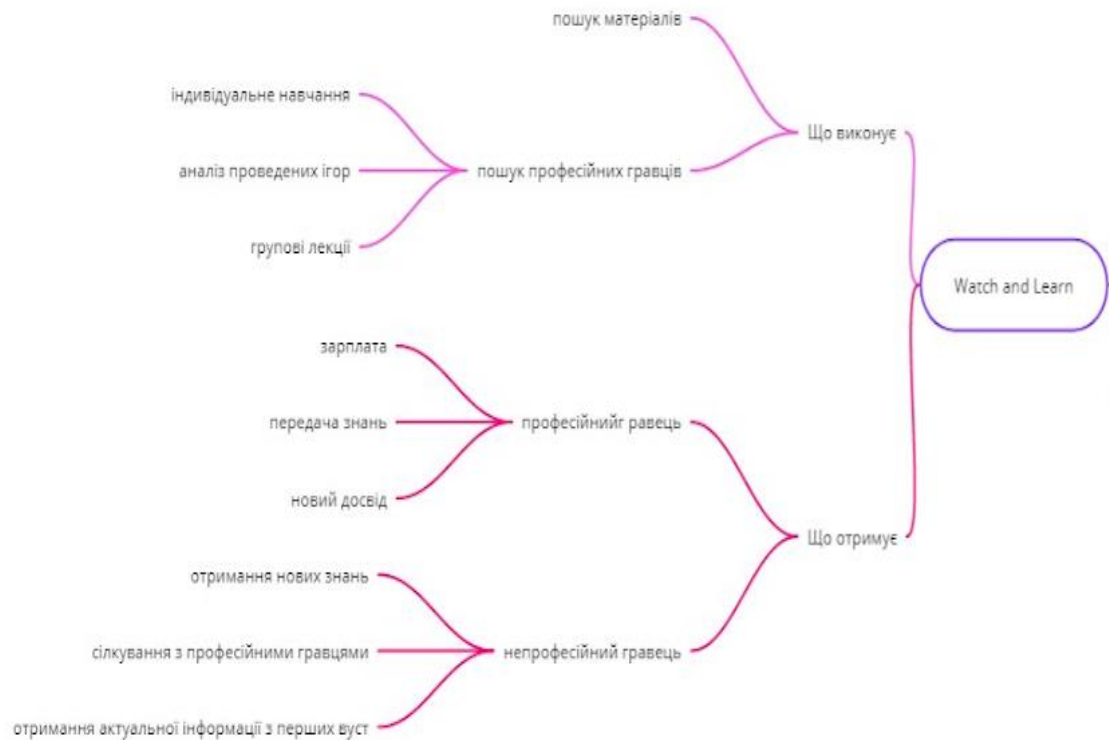


Рис 1.2 - Деталізація матеріальної потреби (частина 1)



Рис 1.3 - Деталізація матеріальної потреби (частина 2)

1.2 Бізнес-вимоги до програмного продукту

1.2.1 Опис проблеми споживача

1.2.1.1 Концептуальний опис проблеми споживача

На сьогоднішній день в більшості ігор використовується змагальна система прогресу, в основі якої часто лежить прокачування героїв та отримання й використання внутрішньоігрових ресурсів. Офіційна покупка у розробників у грі коштує великих грошей та не приносить жодної допомоги гравцю. Одним із головних недоліків онлайн ігор є високий поріг входу для новачків. У гравця не має одного ресурсу, в якому буде актуальна та цінна для нього інформація, вона уся розкидана і шукати її, зазвичай, не має ані бажання, ані часу. Так, зазвичай, гравці або отримують задоволення навіть граючи погано, або покидають гру, адже так і не знайшли собі місця.

1.2.1.2 Опис цільової групи споживачів

У наш час основними користувачами ігор є як діти 5ти років, так і дорослі, котрим вже далеко за 50. Основними користувачами, які тільки створюють свій шлях до кращої гри, на нашу думку, є діти віком від 14ти років та дорослі до 30 років. Зазвичай до віку 14 та після віку 30, люди відносяться до ігор більше, як до забави, аніж як до дійсно спортивного задоволення.

Більшість користувачів будуть юнаками, адже їх у іграх більший відсоток, особливо у більш високих рангах, але також активними користувачами будуть дівчата, адже їм більш важко почати вчитися більш високому рівню гри через недостатню внутрішню концентрацію.

1.2.1.3 Метричний опис проблеми споживача

Низький рівень знань про гру.

Рівень знань = $(N * A) / (K * 100)$,

де N – кількість персонажів;

A – Abilities, тобто вміння персонажів;

K – knowledge, тобто вже знайомі вміння

Анкетування було проведено на 20 студентах потоку AI19x на грі League of Legends, 15 хлопців та 5 дівчат. Результати показали, що зазвичай хлопці мають більш високий ранг, адже більш серйозно відносяться до свого хобі і знають набагато більше персонажів та їх вміння.

1.2.2 Мета створення програмного продукту

1.2.2.1 Проблемний аналіз існуючих програмних продуктів

У таблиці 1 можна побачити, що ні один з протестованих ПП не відповідає одразу всім можливостям ПП, що створюють

Таблиця 1 - Проблемний аналіз існуючих програмних продуктів

Можливості	Наявність у ПП				
	G2A	G2G	FunPay	Gym Keeper	Система, що проектується
Аутентифікація	-	+	+	-	+
Профіль	+	-	+	-	+
Навігація	+	-	+	+	+
Мобільний додаток	+	-	-	-	+

1.2.2.2 Мета створення програмного продукту

Поліпшення навичок гри клієнтів, покращення розуміння гри та надання зрозумілих ресурсів, які допоможуть адаптуватися у грі, надання можливості розміщення матеріалів та продажі ігрових курсів.

1.2.3 Назва програмного продукту

Програма «WatchAndLearn»

1.2.3.1 Гасло програмного продукту

«Для геймерів, що дійсно хочуть стати краще.»

1.2.3.2 Логотип програмного продукту

На рисунку 1.4 продемонстровано логотип ПП «Watch And Learn»



Рис 1.4 – Логотип «Watch And Learn»

1.3 Вимоги користувача до програмного продукту

1.3.1 Пригодницька історія користувача програмного продукту

Недавно я почав грати у гру League of Legends разом з друзями. Але вони чомусь перестали грати зі мною. Я мало розумію у грі, але я дійсно хочу навчитися. Проте в інтернеті це дуже важко, а всі друзі лише сміються наді мною.

Але нещодавно я знайшов крутий мобільний додаток Watch&Learn, який допоміг мені впоратися з усіма важкими моментами. Там я не тільки зміг прочитати багато цікавого про гру та новинки, що в ній трапились, а й зміг

записатися на курси, де краще зрозумів, на якій ролі я хочу грати. Після цього Watch&Learn допомогли мені ще й підняти навички у грі на новій улюбленій позиції.

Нещодавно я потрапив у команду проти моїх друзів. Вони думали, що зможуть легко мене перемогти, але яке ж в них було здивування, коли я показав високий рівень гри і привів свою команду до перемоги. Разом з Watch&Learn легко вчитися грати і перемогати.

1.3.2 Історія користувача програмного продукту

Як гість я хочу:

- Мати змогу завести новий профіль
- Мати змогу авторизуватися до профілю, що я створив
- Мати змогу виставити фільтри для пошуку серед ігор, які цікавлять мене

Як авторизований користувач я хочу:

- Мати змогу редагувати свій профіль
- Мати змогу передивлятися всі курси
- Мати змогу купляти нові курси, матеріали.

1.3.3 Діаграма прецедентів програмного продукту

На рисунку 1.5 можна побачити діаграму прецедентів програмного продукту

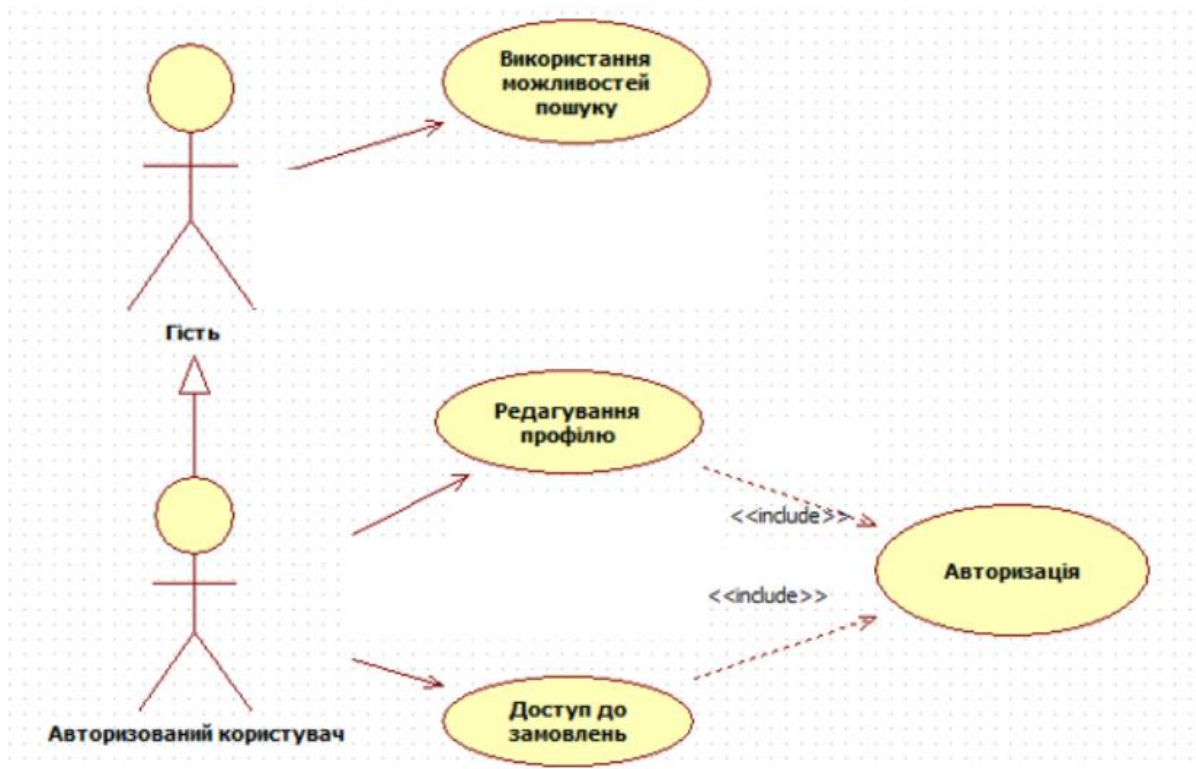


Рис 1.5 - Діаграма прецедентів програмного продукту

1.3.4 Сценаріїв використання прецедентів програмного продукту

1. Прецедент ПП «Авторизувати користувача»:

Таблиця 1. 1 – Прецедент авторизації користувача

Пункт	Опис
Прецедент	Авторизація користувача
Передумова початку виконання прецеденту	Відкриття застосунку на пристрої
Актори як зацікавлені особи у виконанні прецеденту	Гість, користувач
Актор-основна зацікавлена особа як ініціатор початку прецеденту	Гість
Гарантія успіху	Отримання доступу до свого профіля та всіх його можливостей

Приклад основного успішного сценарію прецедента «Авторизувати користувача»:

1. Програма запитує у гостя його параметри авторизації (адреса ел.пошти та пароль)
2. Гість вводить свої власні параметри (адреса ел.пошти та пароль)
3. Програма надає авторизованому користувачу доступ до прецедентів

Приклад альтернативного сценарію для успішного прикладу основного сценарію прецедента «Авторизувати користувача»:

1. Програма запитує у гостя його параметри авторизації (адреса ел.пошти та пароль)
2. Гість вводить свої власні параметри (адреса ел.пошти та пароль)
3. Програма виявляє, що параметри (адреса ел.пошти та пароль) користувача не є вірними
4. Програма видає повідомлення про помилку і повертається на виконання 1 кроку

2. Прецедент ПП «Редагування профіля користувача»:

Таблиця 1. 2 – Прецедент редагування профіля користувача

Пункт	Опис
Прецедент	Редагування профіля користувача
Передумова початку виконання прецеденту	Успішна авторизація
Актори як зацікавлені особи у виконанні прецеденту	Авторизований користувач
Актор-основна зацікавлена особа як ініціатор початку прецеденту	Авторизований користувач
Гарантія успіху	Оновлена інформація у профілі

Приклад основного успішного сценарію прецедента «Редагування профіля користувача»:

1. Програма видає користувачеві його власну інформацію(ПІБ, дата народження, пошта)
2. Користувач обирає необхідні для зміни параметри та робить зміну.
3. Програма перевіряє цілісність інформації та її цензурованість та зберігає нову інформацію у профіль.

Приклад альтернативного сценарію для успішного прикладу основного сценарію прецедента «Редагування профіля користувача»:

1. Програма видає користувачеві його власну інформацію(ПІБ, дата народження, пошта)

2. Користувач обирає необхідні для зміни параметри та робить зміну.
3. Програма виявляє, що нові параметри користувача не відповідають стандарту або не є цензурованими.
4. Програма видає повідомлення про помилку і просить перевірити введені дані без збереження.

3. Прецедент ПП «Пошук нового курсу»:

Таблиця 1. 3 – Прецедент пошуку нового курсу

Пункт	Опис
Прецедент	Пошук нового курсу
Передумова початку виконання прецеденту	Успішний запуск програми
Актори як зацікавлені особи у виконанні прецеденту	Гість, авторизований користувач
Актор-основна зацікавлена особа як ініціатор початку прецеденту	Гість
Гарантія успіху	Знаходження інформації за введеними фільтрами

Приклад основного успішного сценарію прецедента «Пошук нового курсу»:

1. Користувач заходить у програму та відкриває вікно пошука.
2. Користувач вводить необхідні для нього фільтри і запускає функцію пошуку по курсам та файлам.
3. Програма робить обробку по фільтрам та видає результат, якого чекає користувач.

Приклад альтернативного сценарію для успішного прикладу основного сценарію прецедента «Пошук нового курсу»:

1. Користувач заходить у програму та відкриває вікно пошука.
2. Користувач вводить необхідні для нього фільтри і запускає функцію пошуку по курсам та файлам.
3. Програма виявляє, що по даним фільтрам не має ні одного курсу.
4. Програма видає повідомлення про неможливість пошуку за даними фільтрами і просить перевірити введені дані, повертаючись до пункту 2.

4. Прецедент ПП «Оплата нового курсу»:

Таблиця 1. 4 – Прецедент оплати курсу

Пункт	Опис
Прецедент	Оплата нового курсу
Передумова початку виконання прецеденту	Успішна авторизація
Актори як зацікавлені особи у виконанні прецеденту	Авторизований користувач
Актор-основна зацікавлена особа як ініціатор початку прецеденту	Авторизований користувач
Гарантія успіху	Авторизований користувач

Приклад основного успішного сценарію прецедента «Оплата нового курсу»:

1. Користувач знаходить необхідний для нього курс через пошук використовуючи фільтри і обирає варіант покупки курсу.
2. Програма запитує у користувача номер картки та власні дані для підтвердження замовлення.
3. Програма отримує необхідну суму і надає користувачу можливість зайти до інформації цього курсу та під'єднує користувача до онлайн групи курсу

Приклад альтернативного сценарію для успішного прикладу основного сценарію прецедента «Оплата нового курсу»:

1. Користувач знаходить необхідний для нього курс через пошук використовуючи фільтри і обирає варіант покупки курсу.
2. Програма запитує у користувача номер картки та власні дані для підтвердження замовлення.
3. Програма виявляє, що карта користувача введена невірно, або на карті недостатній баланс.
4. Програма видає повідомлення про помилку і просить перевірити введені дані, повертаючись до пункту 2.

1.4 Функціональні вимоги до програмного продукту

1.4.1. Багаторівнева класифікація функціональних вимог

Таблиця 1. 5 – Багаторівнева класифікація функціональних вимог

Ідентифікатор функції(назва)	Назва функції
FR1	Авторизація користувача
FR1.1	Створення запиту параметрів авторизації (адреса ел.пошти та пароль)
FR1.2	Передача від користувача його параметрів авторизації (адреса ел.пошти та пароль)
FR1.3	Надання авторизованому користувачу доступ до повного функціоналу
FR2	Редагування профілю користувача
FR2.1	Створення запиту параметрів персональної інформації(ім'я, інформація про себе (короткий текст 250 символів), пароль, адреса ел.пошти)
FR2.2	Передача від користувача його нових параметрів
FR2.3	Оновлення і збереження нових персональних даних
FR3	Пошук нового курсу(гість)
FR3.1	Програмний продукт запропонує користувачеві почати пошук.
FR3.2	Користувач передає необхідні для нього критерії(жанр, назва) і запускає функцію пошуку по курсам та файлам.
FR3.3	Програма робить обробку по критеріям та видає результат, якого чекає користувач.
FR4	Оплата нового курсу(авторизований користувач)
FR4.1	Користувач знаходить необхідний для нього курс через пошук використовуючи фільтри і обирає варіант покупки курсу.
FR4.2	Програма запитує у користувача номер картки та власні дані для підтвердження замовлення.
FR4.3	Програма отримує необхідну суму і надає користувачу можливість зайти до інформації цього курсу та під'єднує користувача до онлайн групи курсу
FR5	Реєстрація користувача

FR5.1	Створення запиту параметрів реєстрації (адреса ел.пошти, ім'я та пароль)
FR5.2	Передача від користувача його параметрів авторизації (адреса ел.пошти, ім'я та пароль)

На рисунку 1.6 можна побачити багаторівневу класифікацію функціональних вимог

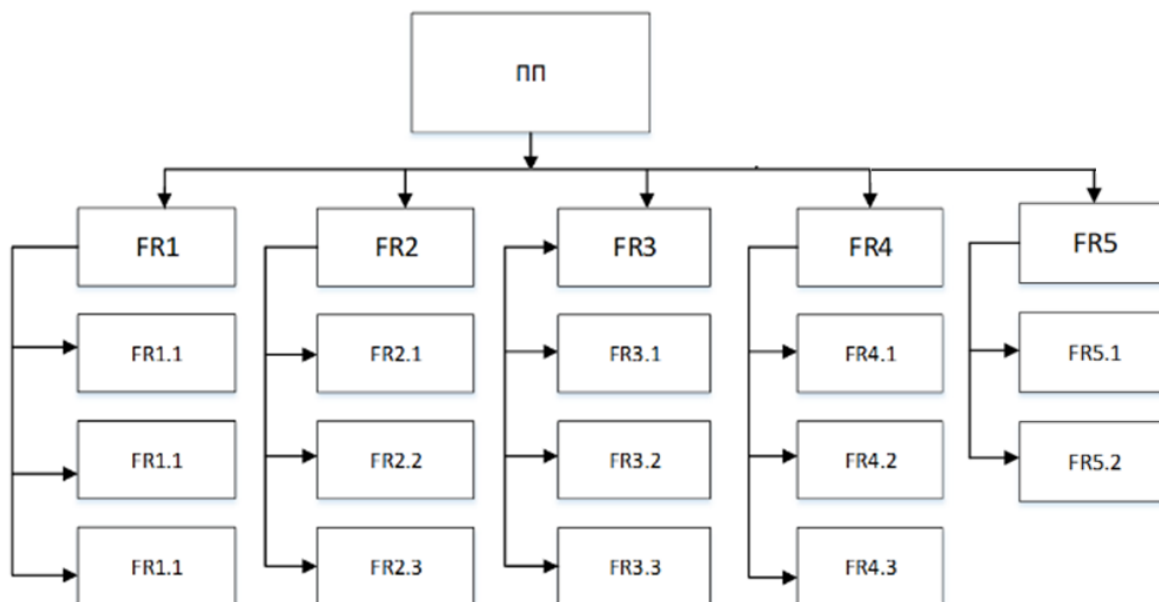


Рис 1.6 – Багаторівнева класифікація функціональних вимог

1.4.2 Функціональний аналіз існуючих програмних продуктів

Таблиця 1. 6 - Функціональний аналіз існуючих програмних продуктів

Можливості	Наявність у додатку				
	G2A	G2G	FunPay	Gym Keeper	Система, що проектується
Аутентифікація	-	+	+	-	+
Профіль	+	-	+	-	+
Пошук нового курсу	+	+	+	+	+
Оплата нового курсу(авторизований користувач)	+	+	+	+	+

1.5 Нефункціональні вимоги до програмного продукту

1.5.1 Опис зовнішніх інтерфейсів

1.5.1.1 Опис інтерфейсів користувача

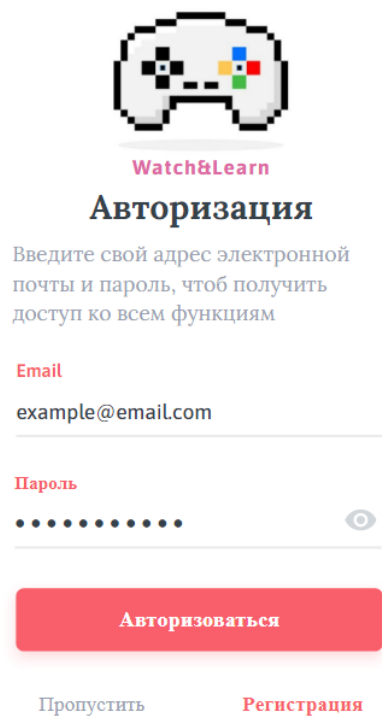
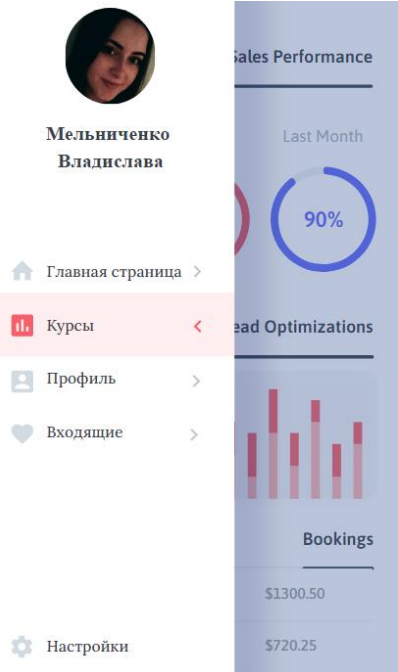
1.5.1.1.1 Опис INPUT-інтерфейсів користувача

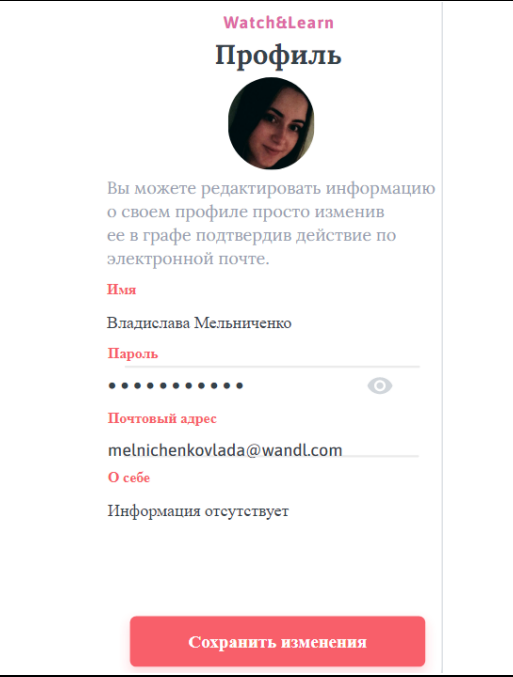
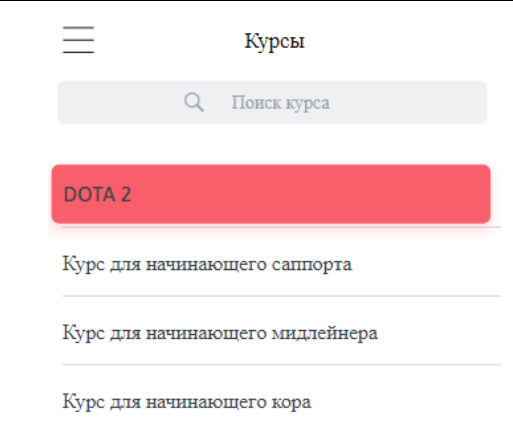
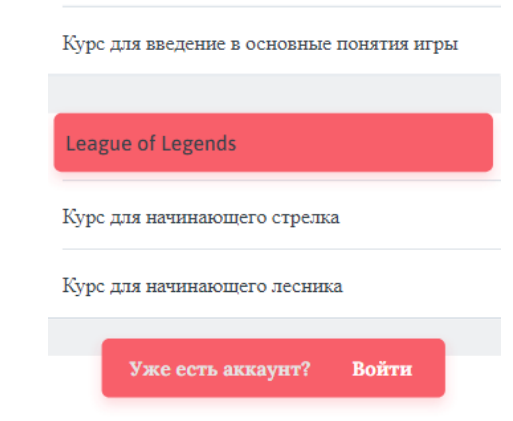
Таблиця 1. 7 – Опис INPUT-інтерфейсів користувача

Ідентифікатор функції (назва)	Засіб INPUT-потoku	Особливості використання
FR1.2 Передача від користувача його параметрів авторизації (адреса ел.пошти та пароль)	Сенсорна віртуальна клавіатура	
FR2.2 Передача від користувача його нових параметрів: інформації(ім'я, інформація про себе (короткий текст 250 символів), пароль, адреса ел.пошти)	Сенсорна віртуальна клавіатура	
FR3.2 Передача від користувача необхідних для нього критеріїв(жанр, назва) і запуск функції пошуку по курсам та файлам.	сенсорна віртуальна клавіатура, сенсорний екран.	Фільтри, що можна вибрати пальцем на сенсорному екрані, які були заздалегідь створені.
FR4.1 Користувач знаходить необхідний для нього курс через пошук використовуючи фільтри і обирає варіант покупки курсу.	сенсорна віртуальна клавіатура, сенсорний екран.	Фільтри, що можна вибрати мишкою, або пальцем на сенсорному екрані, що були заздалегідь створені.
FR5.2 Передача від користувача його параметрів авторизації (адреса ел.пошти, ім'я та пароль)	сенсорна віртуальна клавіатура, сенсорний екран.	

1.5.1.1.2 Опис OUTPUT-інтерфейсів користувача

Таблиця 1. 8 – Опис OUTPUT-інтерфейсів користувача

Ідентифікатор функції (назва)	Засіб OUTPUT- потoku	Особливості використання
FR1.1 Створення запиту параметрів авторизації (адреса ел.пошти та пароль)	Графічний інтерфейс	
FR1.2 Передача від користувача його параметрів авторизації (адреса ел.пошти та пароль)	Графічний інтерфейс	
FR1.3 Надання авторизованому користувачу доступ до повного функціоналу	Графічний інтерфейс	

<p>FR2.1 Створення запиту параметрів персональної інформації(ім'я, інформація про себе (короткий текст 250 символів), пароль, адреса ел.пошти</p>	<p>Графічний інтерфейс</p>	
<p>FR3. Пошук нового курсу(гість)</p>	<p>Графічний інтерфейс</p>	
<p>FR3.1. Програмний продукт запропоновує користувачеві почати пошук.</p>	<p>Графічний інтерфейс</p>	

<p>FR3.2. Користувач передає необхідні для нього критерії(жанр, назва) і запускає функцію пошуку по курсам та файлам.</p>	<p>Графічний інтерфейс</p>	
<p>FR4.2 Програма запитує у користувача номер картки та власні дані для підтвердження замовлення.</p>	<p>Графічний інтерфейс</p>	
<p>FR4.3 Програма отримує необхідну суму і надає користувачу можливість зайти до інформації цього курсу та під'єднує користувача до онлайн групи курсу</p>	<p>Графічний інтерфейс</p>	
<p>FR5.2 Передача від користувача його параметрів авторизації (адреса ел.пошти, ім'я та пароль)</p>	<p>Графічний інтерфейс</p>	

1.5.1.2 Опис інтерфейсу із зовнішніми пристроями

Рекомендується використовувати безпроводний Wi-Fi.

1.5.1.3 Опис програмних інтерфейсів

Програма буде працювати на Android версії 5.0+.

Програма буде взаємодіяти с Google pay для заповнення інформації про картку.

1.5.1.4 Опис інтерфейсів передачі інформації

1.5.1.5 Опис атрибутів продуктивності

На більшість дій ПП буде затримка не більше, ніж 3 секунди.

Єдині функції ПП, що можуть викликати більшу затримку:

- Авторизація користувача(до 10 секунд)
- Оплата курсу(залежить від швидкості банківського платежу)

Одночасно буде обслуговуватися не більше 1000 клієнтів. Якщо кількість клієнтів буде перевищувати це число, у ПП з'явиться спеціальний індикатор про можливі затримки у роботі ПП.

2 Планування процесу розробки програмного продукту

2.1 Планування ітерацій розробки програмного продукту

З метою забезпечення вимог таких рекомендацій IEEE-стандарту, як необхідність, корисність при експлуатації, здійсненість функціональних вимог до ПП, визначено функціональні пріоритети, які будуть використані при плануванні ітерацій розробки ПП. Результати представлено в таблиці 2.1

Таблиця 2.1 – приклад опису функцій з наданням унікальних ієрархічних ідентифікаторів

Ідентифікатор функції(назва)	Назва функції
FR1	Авторизація користувача
FR1.1	Створення запиту параметрів авторизації (адреса ел.пошти та пароль)
FR1.2	Передача від користувача його параметрів авторизації (адреса ел.пошти та пароль)
FR1.3	Надання авторизованому користувачу доступ до повного функціоналу
FR2	Редагування профілю користувача
FR2.1	Створення запиту параметрів персональної інформації(ім'я, інформація про себе (короткий текст 250 символів), пароль, адреса ел.пошти)
FR2.2	Передача від користувача його нових параметрів
FR2.3	Оновлення і збереження нових персональних даних
FR3	Пошук нового курсу(гість)
FR3.1	Програмний продукт запропонує користувачеві почати пошук.
FR3.2	Користувач передає необхідні для нього критерії(жанр, назва) і запускає функцію пошуку по курсам та файлам.
FR3.3	Програма робить обробку по критеріям та видає результат, якого чекає користувач.
FR4	Оплата нового курсу(авторизований користувач)

FR4.1	Користувач знаходить необхідний для нього курс через пошук використовуючи фільтри і обирає варіант покупки курсу.
FR4.2	Програма запитує у користувача номер картки та власні дані для підтвердження замовлення.
FR4.3	Програма отримує необхідну суму і надає користувачу можливість зайти до інформації цього курсу та під'єднує користувача до онлайн групи курсу
FR5	Реєстрація користувача
FR5.1	Створення запиту параметрів реєстрації (адреса ел.пошти, ім'я та пароль)
FR5.2	Передача від користувача його параметрів авторизації (адреса ел.пошти, ім'я та пароль)

2.2 Концептуальний опис архітектури програмного продукту

Mobile Application (MA) – мобільний ПП, розроблений під одну з платформ Android/iOS/ J2ME/Blackberry, який, як правило, виконується на пристроях з обмеженими апаратними ресурсами.

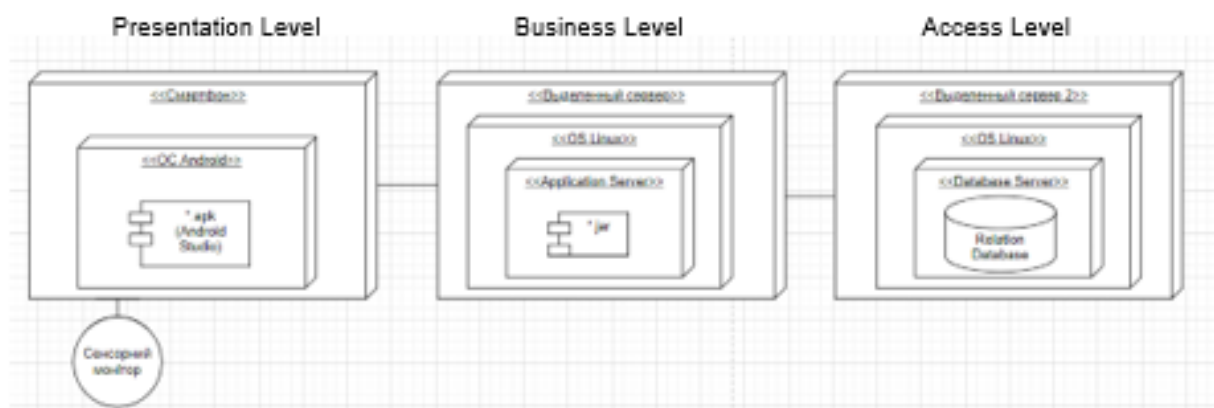


Рис. 2.1 – UML-діаграма розгортання ПП

2.3 План розробки програмного продукту

2.3.1 Оцінка трудомісткості розробки програмного продукту

Таблиця 2.2 - Визначення вагових показників акторів А

Актор	Ваговий коефіцієнт
Гість	1
Авторизований користувач	3

$$A=3+1=4$$

Таблиця 2.3 - Визначення вагових показників прецедентів UC

Прецедент	Кількість кроків сценарію	Ваговий коефіцієнт
Авторизація користувача	4	10
Редагування профілю користувача	4	5
Пошук нового курсу(гість)	4	10
Оплата нового курсу (авторизований користувач)	4	5
Реєстрація користувача	4	10

$$UC = 2*5 + 3*10 = 40$$

$$UUCP=A+UC$$

$$UUCP=4+30=34$$

Таблиця 2.4 - Визначення технічної складності проекту

Показник	Опис показника	Вага
T1	Розподілена система	1
T2	Висока продуктивність (пропускна здатність)	3
T3	Робота кінцевих користувачів в режимі он-лайн	5
T4	Складна обробка даних	1
T5	Повторне використання коду	1
T6	Простота установки	1
T7	Простота використання	5

T8	Переносимість	4
T9	Простота внесення змін	4
T10	Паралелізм	2
T11	Спеціальні вимоги до безпеки	5
T12	Безпосередній доступ до системи з боку зовнішніх користувачів	1
T13	Спеціальні вимоги до навчання користувачів	0

$$TCF = 0,6 + (0,01 * (ST_i * Вага_i))$$

$$TCF = 0,6 + (0,01 * (1 * 2 + 3 * 1 + 5 * 1 + 1 * (-1) + 1 * 1 + 1 * 0,5 + 5 * 0,5 + 4 * 2 + 4 * 1 + 2 * 1 + 5 * 1 + 1 * 1 + 0 * 1)) = 0,6 + (0,01 * 33) = 0,6 + 0,33 = 0,93$$

Таблиця 2.5 - Визначення рівня кваліфікації розробників

Показник	Опис показника	Вага
F1	Знайомство з технологією	1
F2	Досвід розробки додатків	0
F3	Досвід використання об'єктно-орієнтованого підходу	1
F4	Наявність провідного аналітика	1
F5	Мотивація	5
F6	Стабільність вимог	3
F7	Часткова зайнятість	2
F8	Складні мови програмування	5

$$EF = 1,4 + (-0,03 * (SF_i * Вага_i))$$

$$EF = 1,4 + (-0,03 * (1 * 1 + 2 * 0 + 3 * 1 + 4 * 1 + 5 * 5 + 6 * 3 + 7 * 2 + 8 * 5)) = -1,75$$

$$UCP = UUCP * TCF * EF$$

$$UCP = 34 * 0,93 * (-1,75) = -55,335$$

2.3.2 Визначення дерева робіт з розробки програмного продукту

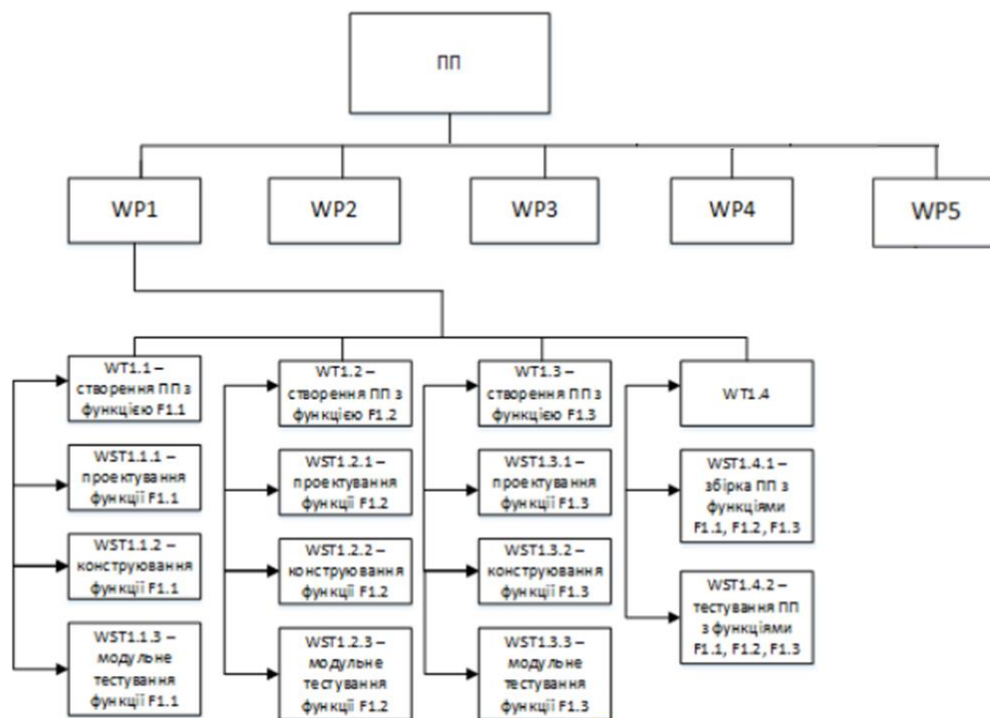


Рис 2.2 – розгортання WP1

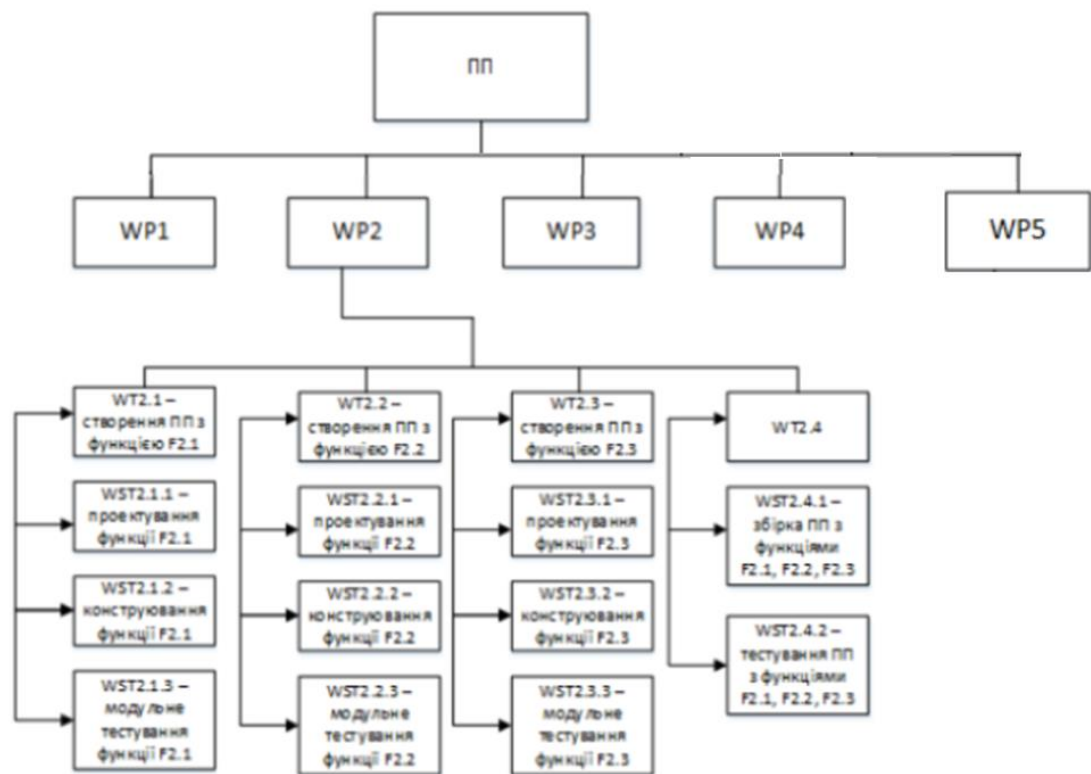


Рис 2.3 – розгортання WP2

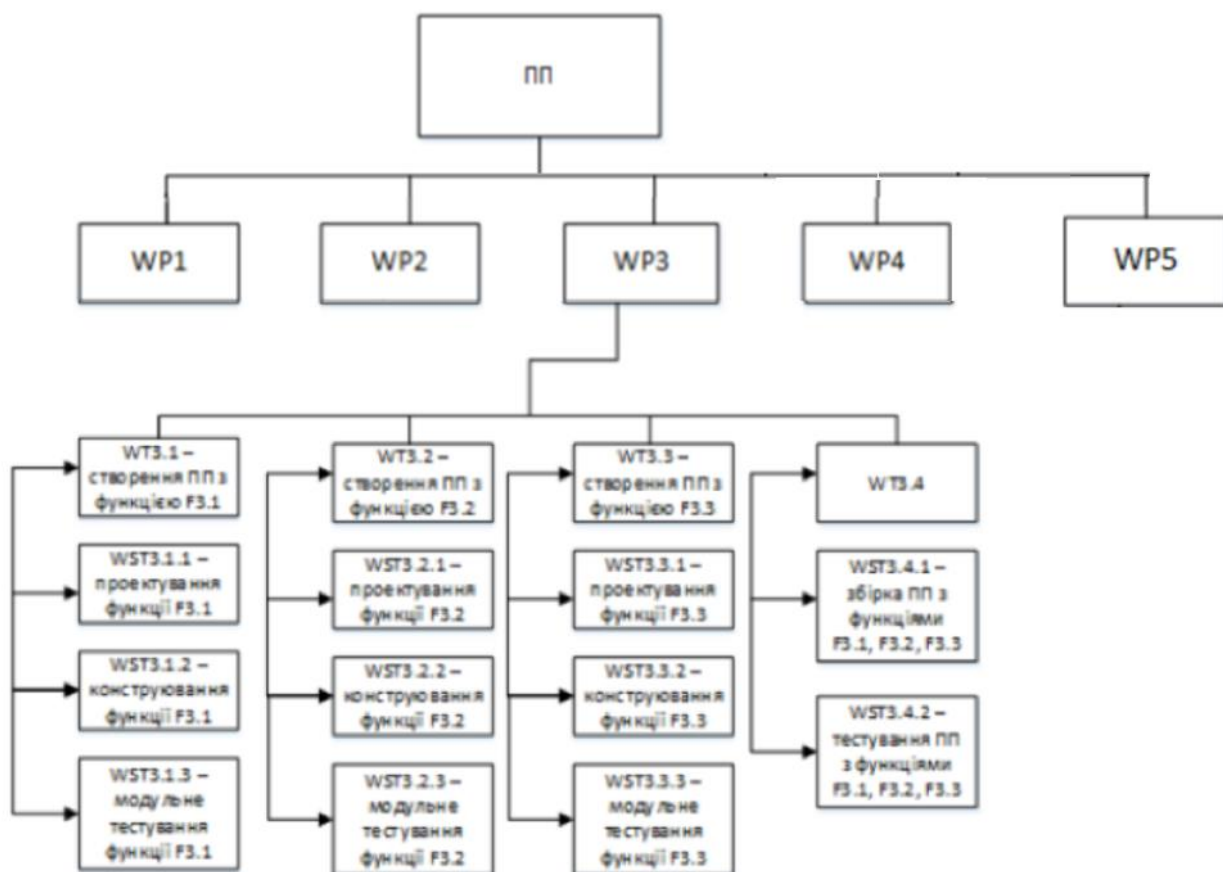


Рис 2.4 – розгортання WP3

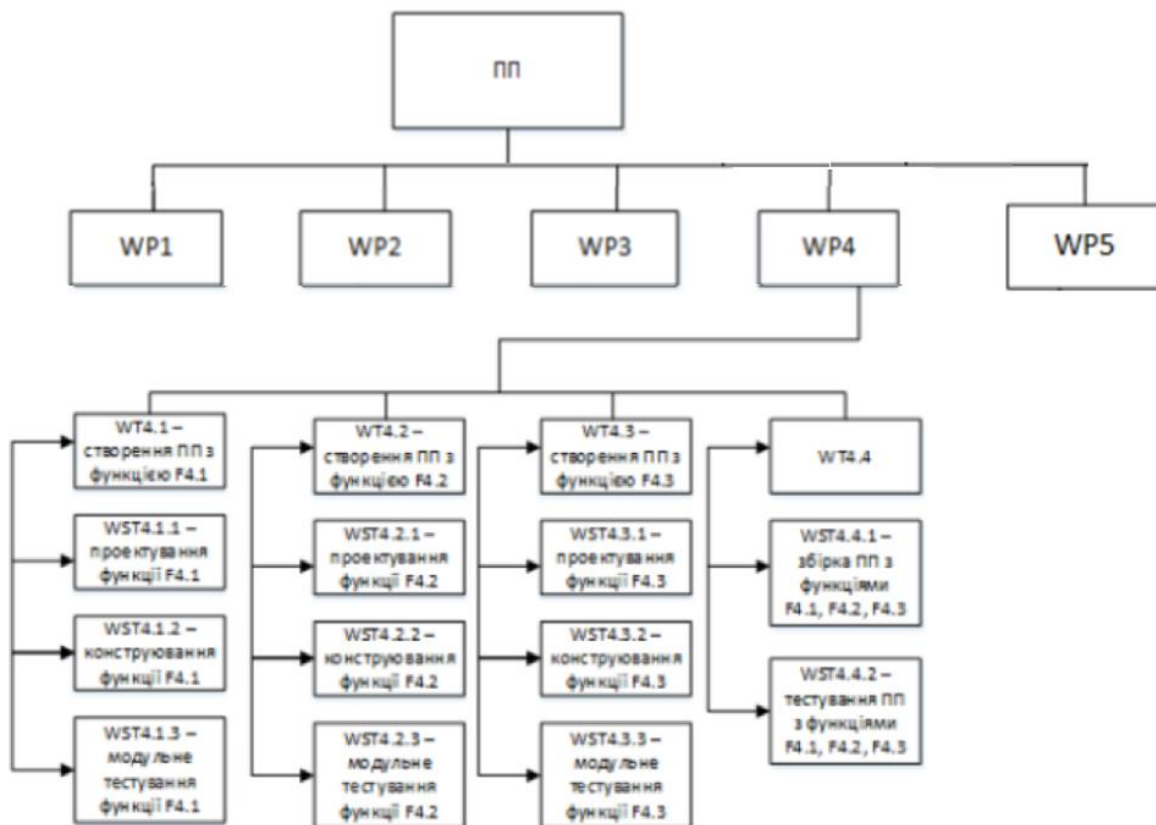


Рис 2.5 – розгортання WP4

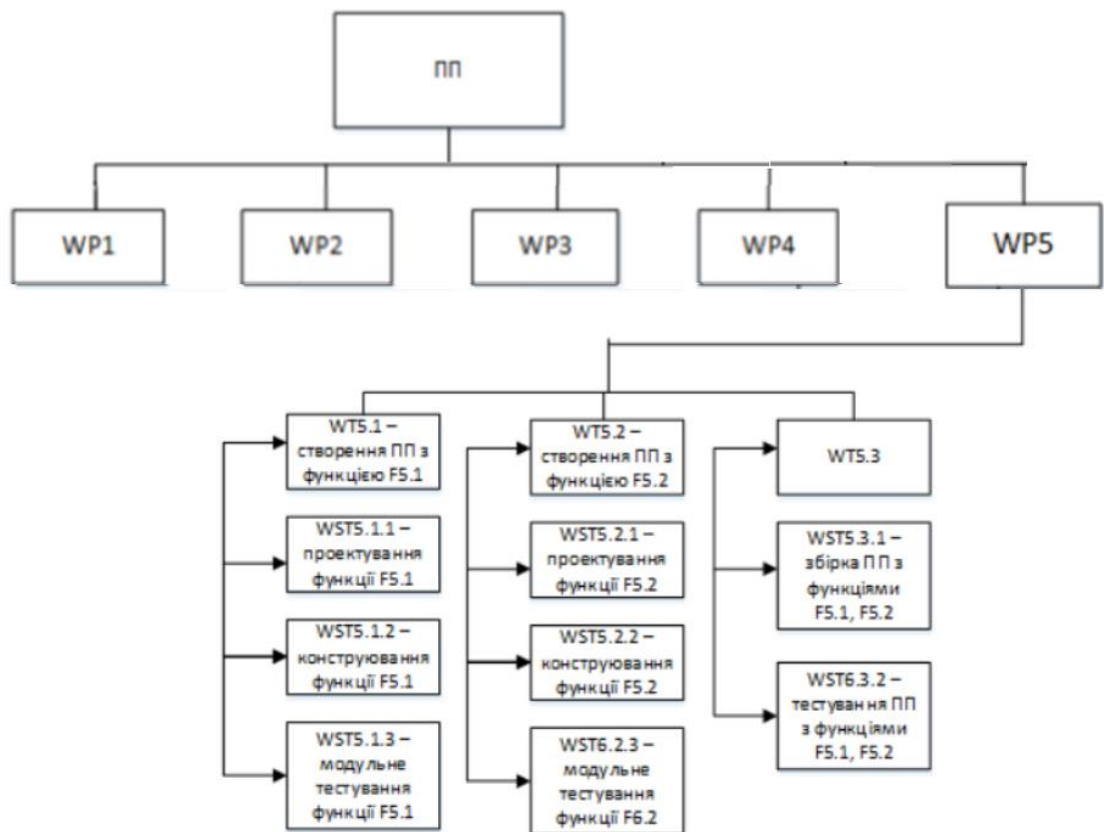


Рис 2.6 – розгортання WP5

2.3.3 Графік робіт з розробки програмного продукту

2.3.3.1 Таблиця з графіком робіт

Таблиця 2.6 - Таблиця з графіком робіт

WST	Дата початку	Дні	Дата завершення	Виконавець
1.1	25.10.2021	2	27.10.2021	Шинкаренко А.В.
1.2	28.10.2021	2	30.10.2021	Шинкаренко А.В.
1.3	30.10.2021	2	02.11.2021	Шинкаренко А.В.
1.4	02.11.2021	2	04.11.2021	Шинкаренко А.В.
2.1	04.11.2021	3	07.11.2021	Шинкаренко А.В.
2.2	07.11.2021	4	11.11.2021	Шинкаренко А.В.
2.3	11.11.2021	4	15.11.2021	Шинкаренко А.В.
2.4	15.11.2021	3	18.11.2021	Шинкаренко А.В.
3.1	18.11.2021	2	20.11.2021	Шинкаренко А.В.

3.2	20.11.2021	3	23.11.2021	Шинкаренко А.В.
3.3	23.11.2021	1	24.11.2021	Шинкаренко А.В.
3.4	24.11.2021	2	26.11.2021	Шинкаренко А.В.
4.1	26.11.2021	2	28.11.2021	Шинкаренко А.В.
4.2	28.11.2021	1	29.11.2021	Шинкаренко А.В.
4.3	29.11.2021	1	30.11.2021	Шинкаренко А.В.
4.4	30.11.2021	2	02.12.2021	Шинкаренко А.В.
5.1	02.12.2021	2	04.12.2021	Шинкаренко А.В.
5.2	04.12.2021	2	06.12.2021	Шинкаренко А.В.
5.3	26.11.2021	2	08.12.2021	Шинкаренко А.В.

2.3.3.2 Діаграма Ганта

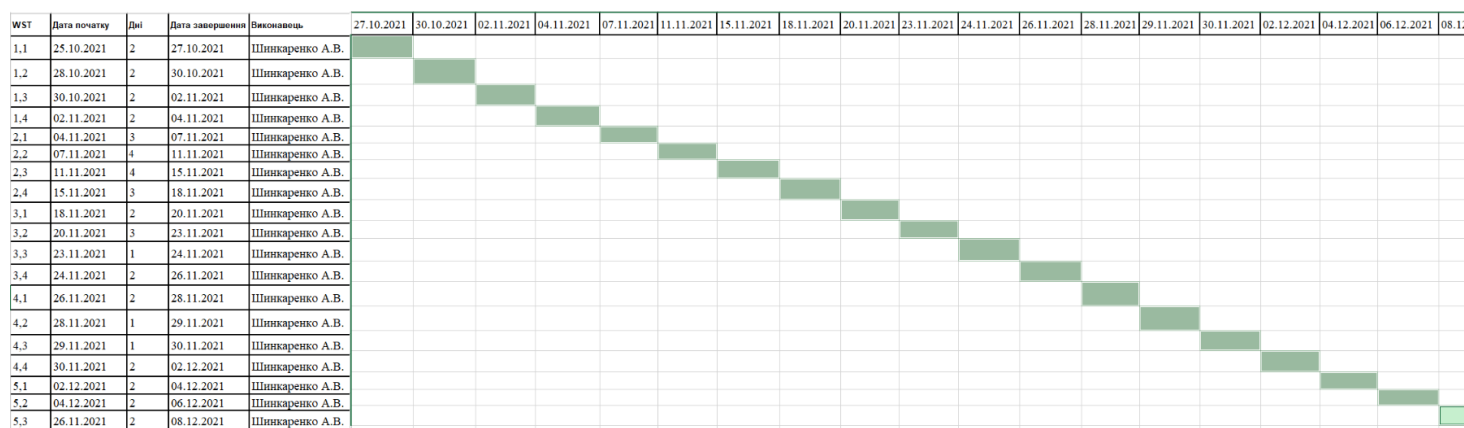


Рис 2.7 – Діаграма Ганта.

3 Проектування програмного продукту

3.1 Концептуальне та логічне проектування структур даних програмного продукту

3.1.1 Концептуальне проектування на основі *UML*-діаграми концептуальних класів

На рисунку 3.1.1 можна побачити *UML*-діаграму концептуальних класів

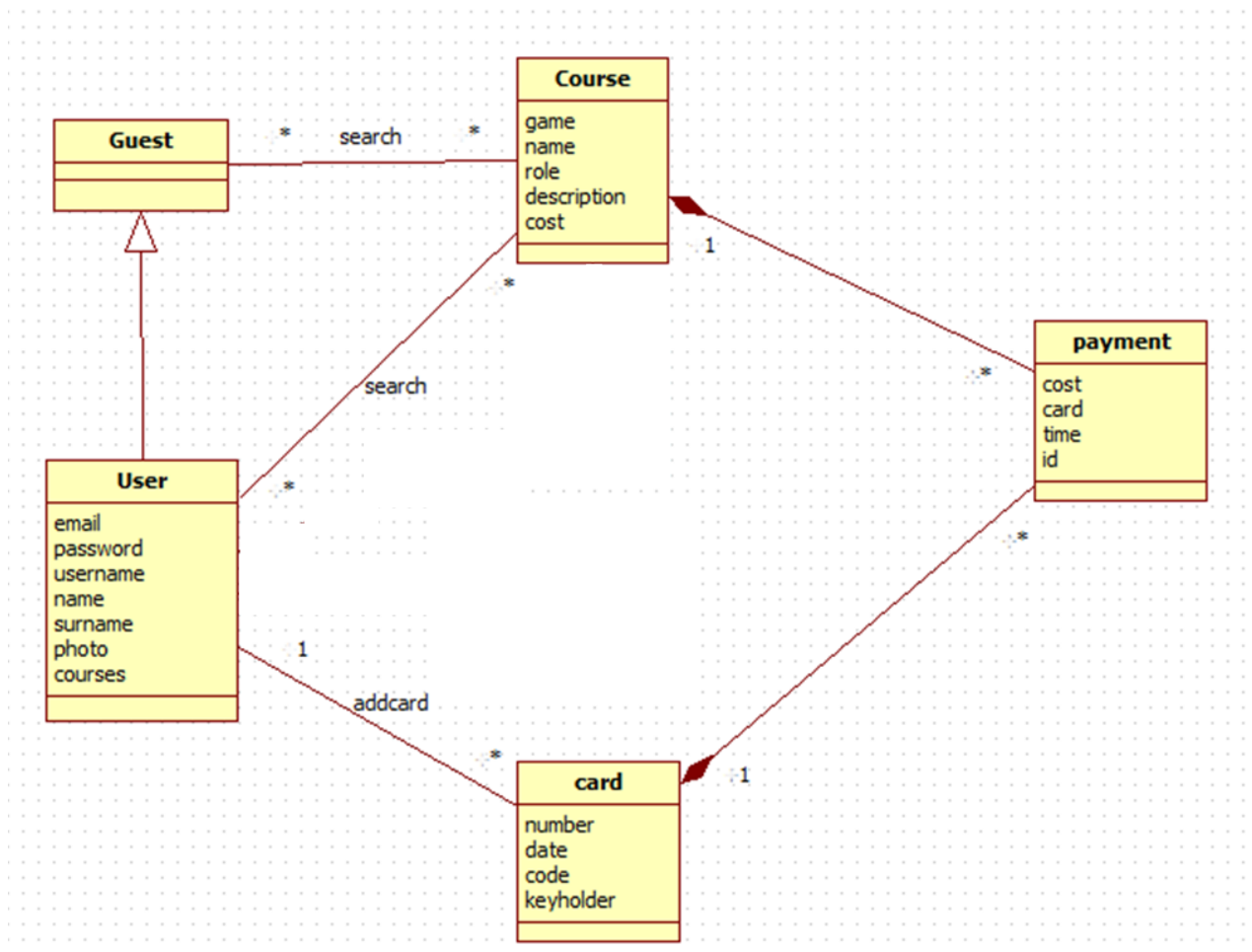


Рис 3.1.1 - *UML*-діаграма концептуальних класів

3.1.2 Логічне проектування структур даних

На рисунку 3.1.2 продемонстровано спроектовану структуру даних

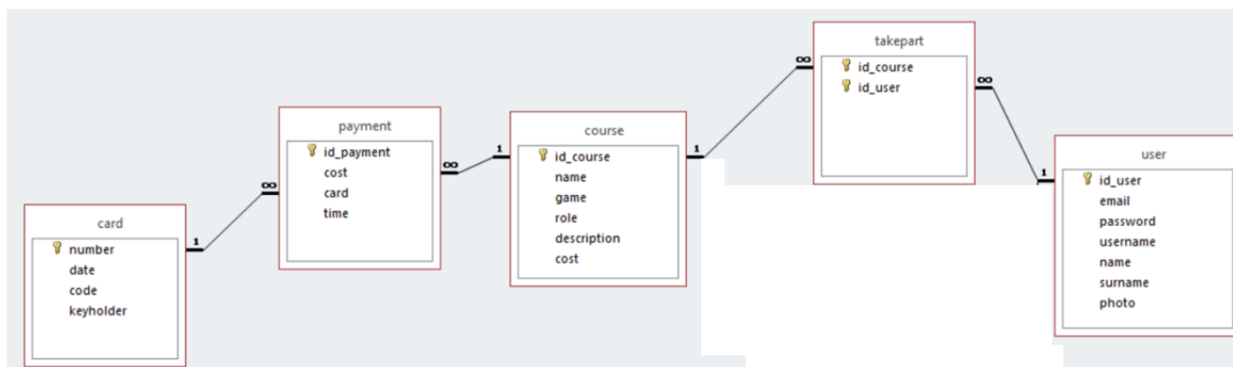


Рис 3.1.2 – спроектована структура даних

3.2 Проектування програмних класів

У таблиці 3.2 продемонстровано спроектовані програмні класи.

Таблиця 3.2 – програмні класи

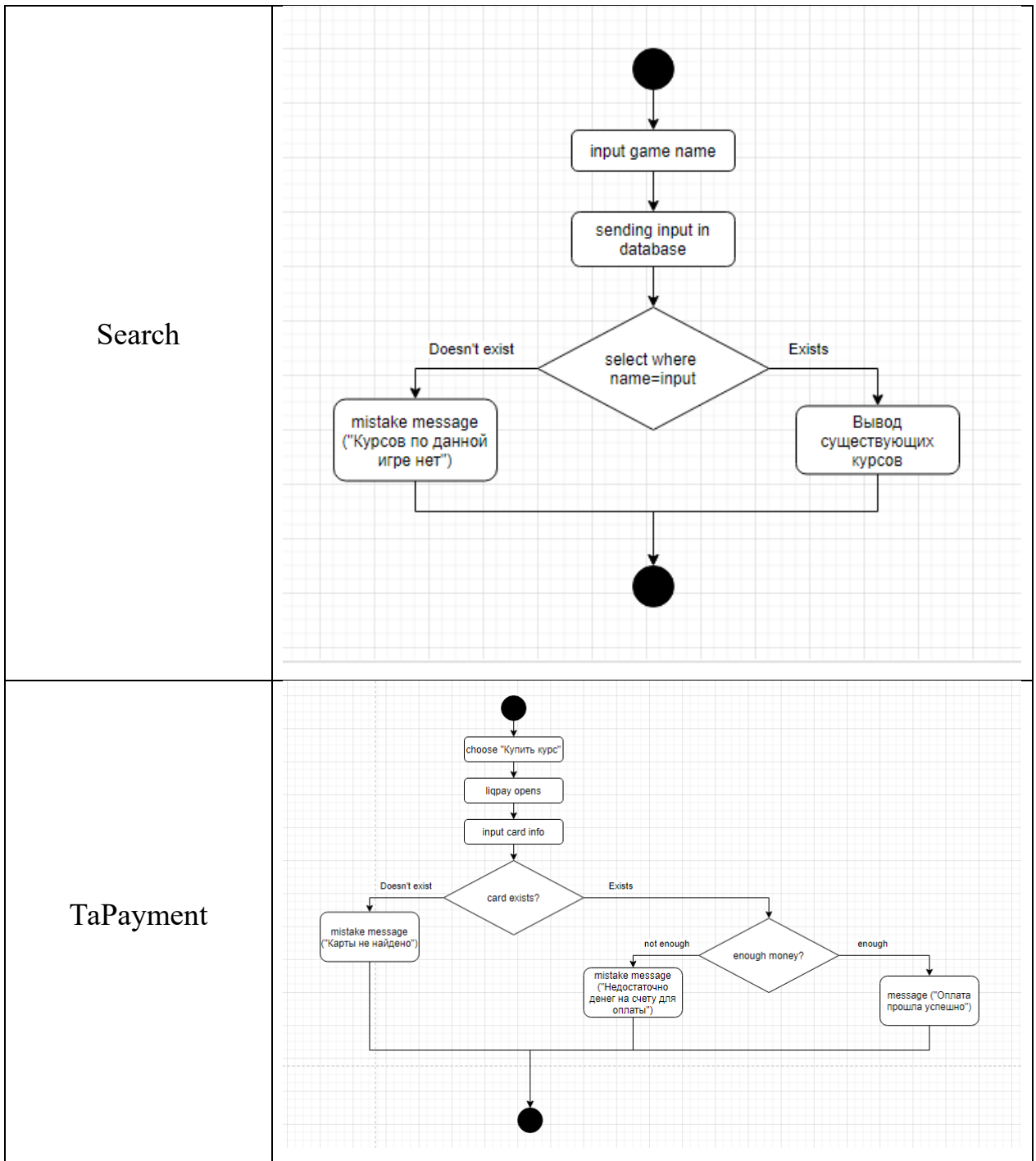
Ідентифікатор функції	Назва функції	Назва класу	Назва методу класу з параметрами
FR1	Авторизація користувача	User	AuthUser(email: string, password: string)
FR2	Редагуванню профілю користувача	User	EditUser(name: string, surname: string, photo: string)
FR3	Пошук нового курсу	Course	Search(game: string, name: string, role: enum, cost: float)
FR4	Оплата нового курсу	Payment	Payment(number: int, date: datetime, code: int, keyholder: string, course: int)
FR5	Рєєстрація	Register	Register(name: string, email: string, password: string)

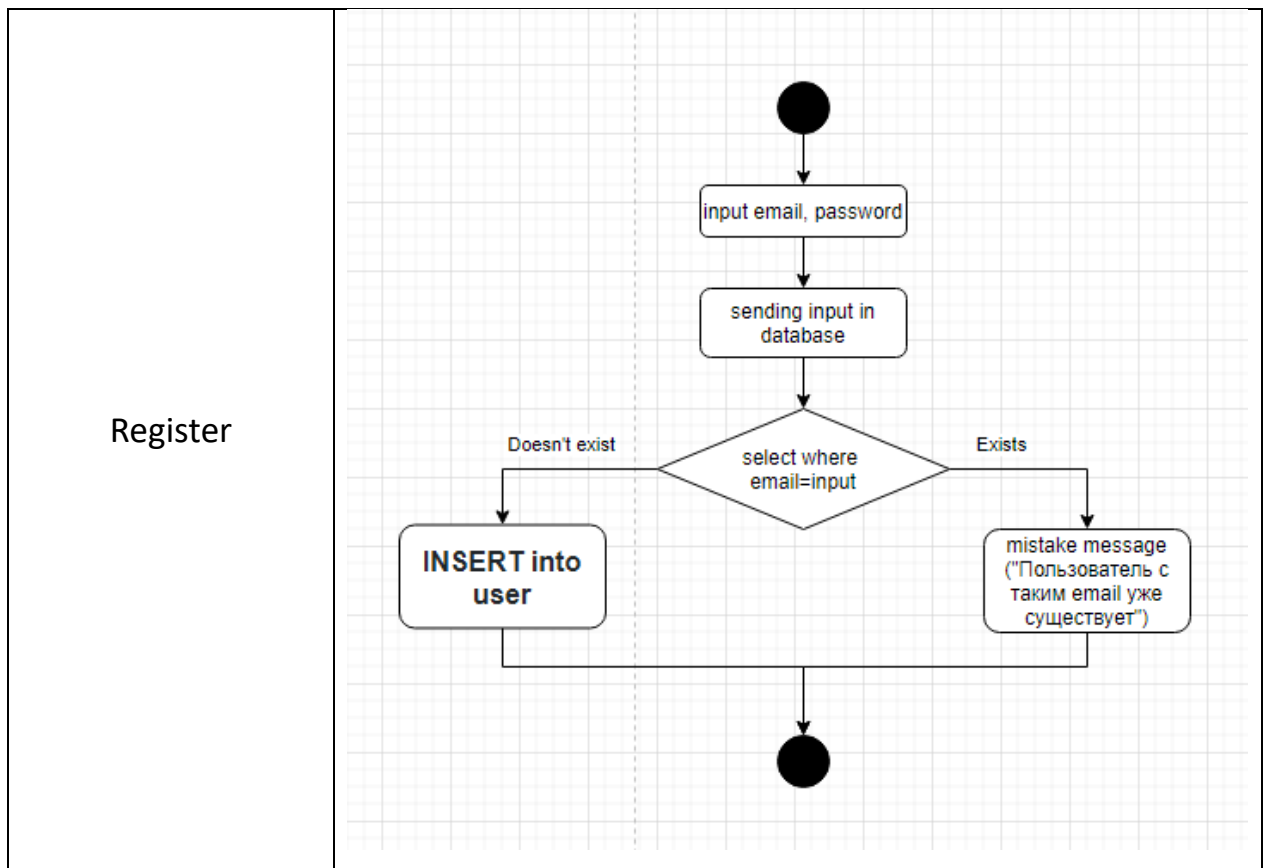
3.3 Проектування алгоритмів роботи методів програмних класів

У таблиці 3.3 можна побачити алгоритми роботи методів програмних класів.

Таблиця 3.3 – методи програмних класів

Назва методу	Алгоритм роботи
AuthUser	<pre> graph TD Start(()) --> Input[input email, password] Input --> Send[sending input in database] Send --> Decision{select where email=input, password=input} Decision -- Doesn't exist --> Mistake[mistake message ("Неправильный логин или пароль")] Decision -- Exists --> Login[log in] Mistake --> End(()) Login --> End </pre>
EditUser	<pre> graph TD Start(()) --> Input[input changes in name, surname, password] Input --> Send[sending input in database] Send --> Update[UPDATE into user] Update --> End(()) </pre>





3.4 Проектування тестових сценаріїв верифікації роботи програмних модулів

3.4.1 Проектування тестових сценаріїв верифікації функціональних вимог

Для тестування функціональних вимог було використано BlackBox-методи та WhiteBox-методи. Функціональні вимоги продемонстровано у таблиці 3.4.1.

Таблиця 3.4.1 - Функціональні вимоги

Ідентифікатор функції(назва)	Назва функції
FR1	Авторизація користувача
FR2	Редагування профілю користувача
FR3	Пошук курсів
FR4	Оплата курсу
FR5	Реєстрація користувача

У таблиці 3.4.2 наведено початкове проектування тестових сценаріїв для FR1.

Вхідні параметри:

Email – адреса електронної пошти(рядок символів)

Password – пароль(рядок символів)

Обмеження поля адреси електронної пошти – повинно містити адресу електронної пошти

Обмеження паролю – довжина від 8 до 64 символів

Таблиця 3.4.2 - Проектування тестових сценаріїв для FR1

FR ID	Test Case ID	Опис значень вхідних даних до методу (процедури, функції)	Опис очікуваних значень результату виконання методу
FR1	TC1	Email="watchandlearn@gmail.com" password="watchandlearn"	Користувача авторизовано, Повідомлення "Добро пожаловать, пользователь!"
FR1	TC2	Email="watchandlearn@gmail.com" password="watchandlearn"	Повідомлення "Неправильно написан адрес электронной почты!"
FR1	TC3	Email="watchandlearn@gmail.com" password="watch"	Повідомлення "Пароль должен иметь минимум 8 символов!"

У таблиці 3.4.3 наведено початкове проектування тестових сценаріїв для FR2.

Вхідні параметри:

Username – ім'я та прізвище користувача(рядок символів)

bio – інформація про користувача(рядок символів)

email – адреса електронної пошти(рядок символів)

password – пароль(рядок символів)

Обмеження поля інформації – повинно містити до 250 символів

Обмеження поля адреси електронної пошти – повинно містити адресу електронної пошти

Обмеження поля імені користувача – повинно містити від 3 до 30 символів

Обмеження паролю – довжина від 8 до 64 символів

Таблиця 3.4.3 - Проектування тестових сценаріїв для FR2

FR ID	Test Case ID	Опис значень вхідних даних до методу (процедури, функції)	Опис очікуваних значень результату виконання методу
FR2	TC1	username = user bio = loremipsum email = <u>username@gmail.com</u> password = 12345678	Профіль відредаговано, Повідомлення «Дані оновлено»
FR2	TC2	username = user bio = [текст, що перевищує 250 символів] email = <u>username@gmail.com</u> password = 12345678	Повідомлення про помилку: «Біо не повинно містити більше 250 символів»
FR2	TC3	username = user bio = loremipsum email = usernamegmail.co m password = 12345678	Повідомлення про помилку: «Некоректно введена адреса ел. пошти»
FR2	TC4	username = [текст, що перевищує 30 символів] bio = loremipsum email = <u>username@gmail.com</u> password = 12345678	Повідомлення про помилку: «Ім'я користувача повинно містити не більше 30 символів»
FR2	TC5	username = [empty]	Повідомлення про помилку:

		bio = loremipsum email = <u>username@gmail.com</u> password = 12345678	«Ім'я користувача не повинно бути порожнім»
FR2	TC6	username = user bio = loremipsum email = <u>username@gmail.com</u> password = 1234567	Повідомлення про помилку: «Пароль повинен містити більше ніж 7 символів»

У таблиці 3.4.4 наведено початкове проектування тестових сценаріїв для FR3.

Вхідні параметри:

game_name – назва гри(рядок символів)

Role – вибор ролі для курсу(рядок символів)

Cost – вартість(цифрове значення від * до *)

обмеження поля назви гри – повинно містити до 30 символів

обмеження поля ролі – повинно містити до 20 символів

обмеження вартості – вартість від 0 до 2000 гривень

Таблиця 3.4.4 - Проектування тестових сценаріїв для FR3

FR ID	Test Case ID	Опис значень вхідних даних до методу (процедури, функції)	Опис очікуваних значень результату виконання методу
FR3	TC1	Назва гри="League of Legends" Роль="Support" Вартість="0 - 1000"	Виведення курсів, що підходять за фільтрами.

FR3	TC2	Назва гри="League of Legends of League of Legends of League" Роль="Support" Вартість="0 - 1000"	Помилка, повідомлення "Название игры должно быть более 30 символов!"
FR3	TC3	Назва гри="League of Legends" Роль="SupportAD CMiderToplanerJungler" Вартість="0 - 1000"	Помилка, повідомлення "Роль должна иметь не более 20 символов!"
FR3	TC4	Назва гри="League of Legends" Роль="Support" Вартість="-10 - 1000"	Помилка, повідомлення "Курс не может стоять меньше, чем 0"

У таблиці 3.4.5 наведено початкове проектування тестових сценаріїв для FR4.

Вхідні параметри:

Cardnumber – номер картки користувача(цифрове значення)

Date – дата, вказана на картці(цифрове значення)

CVV – код, вказаний на обороті карти(цифрове значення)

Номер карти повинен мати 16 цифр

Срок дії повинен бути введений у форматі ММ/РР, та ММ від 0 до 12, РР від 21 до 35.

CVV2 code повинен мати 3 цифри

Таблиця 3.4.5 - Проектування тестових сценаріїв для FR4

FR ID	Test Case ID	Опис значень вхідних даних до методу (процедури, функції)	Опис очікуваних значень результату виконання методу
FR4	TC1	Cardnumber="516 8755900168309" Date="12/22" CVV="133"	Курс успішно оплачено, Повідомлення "Поздравляем с покупкой!"
FR4	TC2	Cardnumber="516 8755900168309" Date="13/22" CVV="133"	Помилка, Повідомлення "У році всього 12 місяців, перевірте введені дані."
FR4	TC3	Cardnumber="Ф1 68755900168309" Date="12/22" CVV="133"	Помилка, Повідомлення "Номер картки не повинен містити букв!"
FR4	TC3	Cardnumber="516 8755900168309" Date="12/22" CVV="T33"	Помилка, Повідомлення "CVV код не повинен містити букв!"

У таблиці 3.4.6 наведено початкове проектування тестових сценаріїв для FR5.

Вхідні параметри:

Username – ім'я та прізвище користувача(рядок символів)

email – адреса електронної пошти(рядок символів)

password – пароль(рядок символів)

обмеження поля адреси електронної пошти – повинно містити адресу
електронної пошти

обмеження поля імені користувача – повинно містити від 0 до 30 символів

обмеження паролю – довжина від 8 до 64 символів

Таблиця 3.4.6 - Проектування тестових сценаріїв для FR5



FR ID	Test Case ID	Опис значень вхідних даних до методу (процедури, функції)	Опис очікуваних значень результату виконання методу
FR5	TC1	username = user email = <u>username@gmail.com</u> password = 12345678	Повідомлення «Добро пожалувать, пользователь!»
FR5	TC2	username = user email = usernamegmail.co m password = 12345678	Повідомлення про помилку: «Некоректно введена адреса ел. пошти»
FR5	TC3	username = [текст, що перевищує 30 символів] email = <u>username@gmail.com</u> password = 12345678	Повідомлення про помилку: «Ім'я користувача повинно містити не більше 30 символів»
FR5	TC4	username = [empty] email = <u>username@gmail.com</u> password = 12345678	Повідомлення про помилку: «Ім'я користувача не повинно бути порожнім»
FR5	TC5	username = user email = <u>username@gmail.com</u> password = 1234567	Повідомлення про помилку: «Пароль повинен містити більше ніж 7 символів»

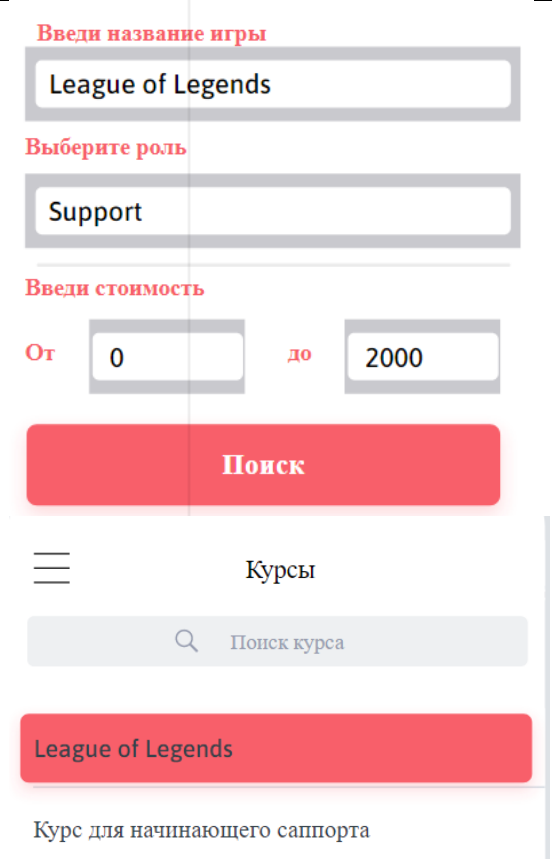
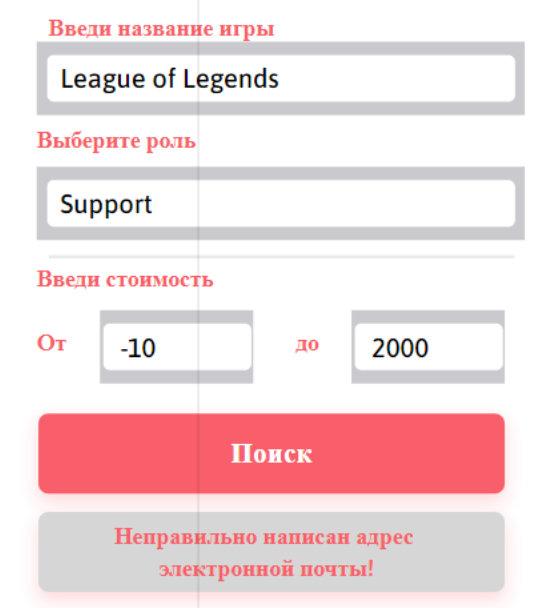
3.4.2 Проектування тестових сценаріїв верифікації нефункціональних вимог

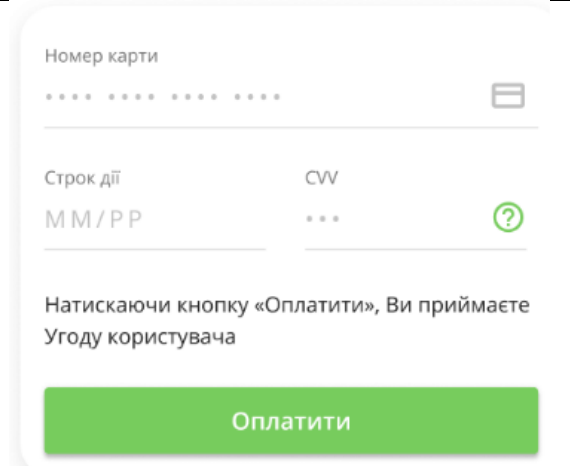
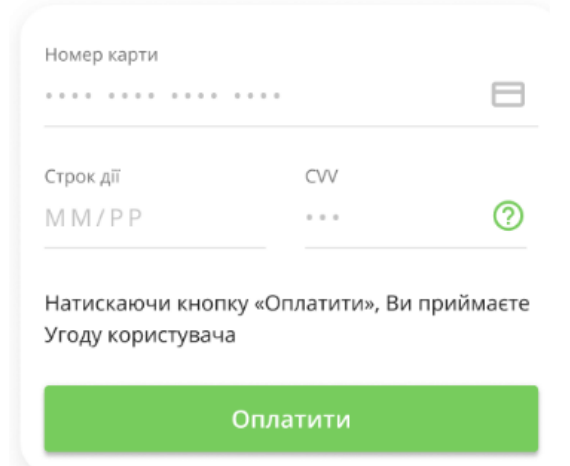
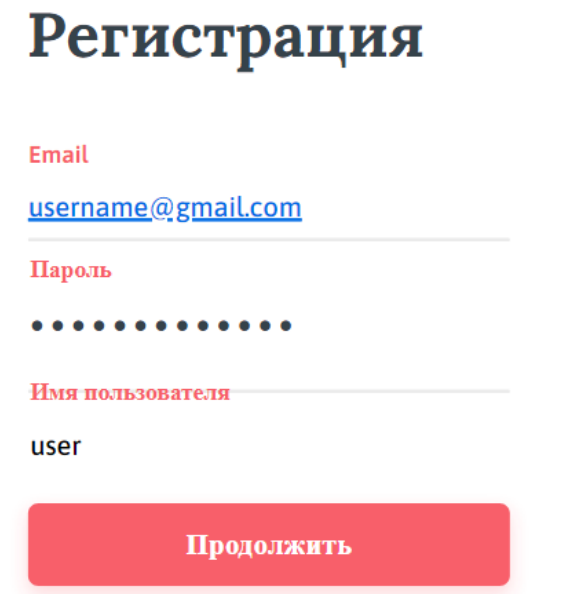
Таблиця 3.4.7 (наступні сторінки містять таблицю 3.4.7 та її продовження) представляє собою проектування тестових сценаріїв верифікації нефункціональних вимог до програмного продукту.

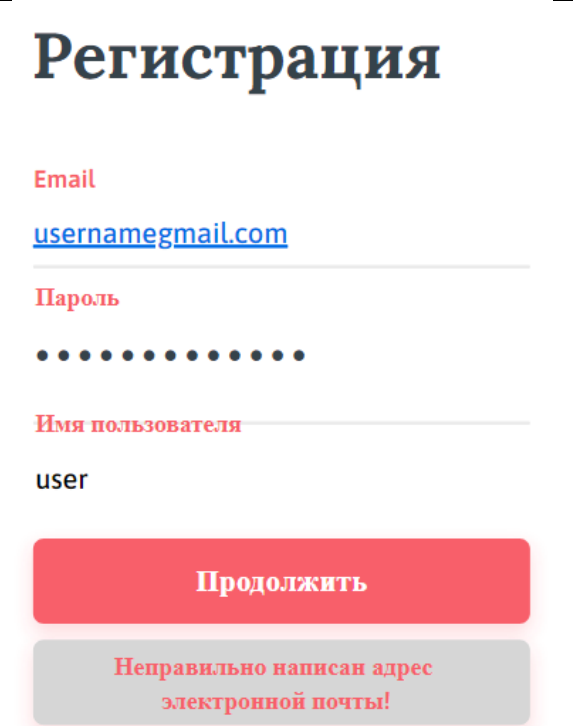
Таблиця 3.4.7 – Проектування тестових сценаріїв верифікації нефункціональних вимог

NFR ID	Test Case ID	Опис дій з ПП на рівні інтерфейсу користувача	Опис очікуваних результатів взаємодії з ПП на рівні інтерфейсу користувача
NFR1	TC1	1) В поле Email ввести значение “watchandlearn@gmail.com” 2) В поле пароль ввести значение “watchandlearn”	<p>Авторизация</p> <p>Введите свой адрес электронной почты и пароль, чтоб получить доступ ко всем функциям</p> <p>Email</p> <p>watchandlearn@gmail.com</p> <p>Пароль</p> <p>.....</p> <p>Авторизоваться</p>
NFR2	TC2	1) В поле Email ввести значение “watchandlearn@gmail.com” 2) В поле пароль ввести значение “watchandlearn”	<p>Авторизация</p> <p>Введите свой адрес электронной почты и пароль, чтоб получить доступ ко всем функциям</p> <p>Email</p> <p>watchandlearn@gmail.com</p> <p>Пароль</p> <p>.....</p> <p>Авторизоваться</p> <p>Пропустить Регистрация</p> <p>Неправильно написан адрес электронной почты!</p>

NFR3	TC1	<p>1) В поле Имя ввести значение “Владислава Мельниченко”</p> <p>2) В поле пароль ввести значение “watchandlearn”</p> <p>3) В поле почтовый адрес ввести “melnichenkovlada@wandl.com”</p>	<p>Имя</p> <p>Владислава Мельниченко</p> <p>Пароль</p> <p>•••••••••• </p> <p>Почтовый адрес</p> <p>melnichenkovlada@wandl.com</p> <p>О себе</p> <p>Информация отсутствует</p> <p>Сохранить изменения</p>
NFR4	TC3	<p>1) В поле Имя ввести значение “Владислава Мельниченко”</p> <p>2) В поле пароль ввести значение “watchandlearn”</p> <p>3) В поле почтовый адрес ввести “melnichenkovladawandl.com”</p>	<p>Имя</p> <p>Владислава Мельниченко</p> <p>Пароль</p> <p>•••••••••• </p> <p>Почтовый адрес</p> <p>melnichenkovladawandl.com</p> <p>О себе</p> <p>Информация отсутствует</p> <p>Неправильно написан адрес электронной почты!</p> <p>Сохранить изменения</p>

NFR5	TC1	<p>1) В поле названия игры ввести значение “League of Legends”</p> <p>2) В поле роли ввести значение “Support”</p> <p>3) В поле стоимости ввести от 0 до 2000</p>	 <p>Введи название игры</p> <p>League of Legends</p> <p>Выберите роль</p> <p>Support</p> <p>Введи стоимость</p> <p>От 0 до 2000</p> <p>Поиск</p> <p>Курсы</p> <p>Поиск курса</p> <p>League of Legends</p> <p>Курс для начинающего саппорта</p>
NFR6	TC4	<p>1) В поле названия игры ввести значение “League of Legends”</p> <p>2) В поле роли ввести значение “Support”</p> <p>3) В поле стоимости ввести от -10 до 2000</p>	 <p>Введи название игры</p> <p>League of Legends</p> <p>Выберите роль</p> <p>Support</p> <p>Введи стоимость</p> <p>От -10 до 2000</p> <p>Поиск</p> <p>Неправильно написан адрес электронной почты!</p>

NFR7	TC1	<p>1) В поле номер карти ввести значение “5168755900168309”</p> <p>2) В поле срок дії ввести значение “08/24”</p> <p>3) В поле CVV ввести V13</p>	
NFR8	TC4	<p>1) В поле номер карти ввести значение “5168755900168309”</p> <p>2) В поле срок дії ввести значение “08/24”</p> <p>3) В поле CVV ввести 313</p>	
NFR9	TC1	<p>1) В поле email ввести значение “<u>username@gmail.com</u>”</p> <p>2) В поле имени пользователя ввести значение “user”</p> <p>3) В поле Пароль ввести “12345678”</p>	

NFR10	TC2	1) В поле email ввести значение <u>“usernamegmail.com”</u> 2) В поле имени пользователя ввести значение “user” 3) В поле Пароль ввести “12345678”	
-------	-----	---	--

3.4.3 Створення матриці відповідності вимог до програмного продукту

Об’єднано всі вимоги та тестові сценарії в матрицю відповідності вимог у вигляді таблиці 3.4.8

Таблица 3.4.8 – Матрица відповідності вимог до програмного продукту

FR ID	NFR ID	TC ID
FR1	NFR 1	TC 1
FR1	NFR 2	TC 2
FR2	NFR 3	TC 1
FR2	NFR 4	TC 3
FR3	NFR 5	TC 1
FR3	NFR 6	TC 4
FR4	NFR 7	TC 1
FR4	NFR 8	TC 4
FR5	NFR 9	TC 1
FR5	NFR 10	TC 2

4 Конструювання програмного продукту

4.1 Особливості конструювання структур даних

Все розгортається локально.

Для зберігання інформації про користувача було створено спеціальні рядки, що зберігають інформацію і можуть змінюватися. Для цього було використано можливості **Save key-value data**.

Створені для програми рядки можна побачити на рис. 4.1, а їх виклик та зміну – на рис 4.2.

```
<string name="email_file_key">email</string>
<string name="about_file_key">About user</string>
<string name="password_file_key">password</string>
<string name="username_file_key">username</string>
<string name="game_name_file_key">game name</string>
<string name="role_file_key">role</string>
```

Рис. 4.1 – рядки для зберігання даних

```
SharedPreferences sharedPref = getActivity().getPreferences(Context.MODE_PRIVATE);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putString("email", email.getText().toString());
editor.putString("password", password.getText().toString());
editor.putString("username", username.getText().toString());
editor.apply();
```

Рис 4.2 – виклик та зміна рядків email, паролю та ім'я користувача.

Вони спільні для всіх компонентів вашої програми (наприклад, для всіх ваших дій).

Sharedpreference в основному є ключовими, пари значень з сан зберігаються в xml і доступні додатку. Ви можете використовувати його для збереження деяких значень налаштувань або значень за промовчанням або будь-якої іншої форми пар значень ключів. Таким чином, пара ключ-значення може бути збережена в одній дії і доступна в іншій дії.

Об'єкт `SharedPreferences` вказує на файл, що містить пари ключ-значення, і надає прості методи їх читання та запису. Кожен файл `SharedPreferences` керується платформою та може бути приватним або спільним.

4.2 Особливості конструювання програмних модулів

4.2.1 Конструювання програмної структури з урахуванням спеціалізованого *Framework* для *FrontEnd*-компонент архітектури

Для написання програми було використано мову програмування Java. Для створення програмного коду було обрано використовувати `Android: Fragments`.

`Fragment` (Фрагмент) - це подібність `Activity`, яку ми можемо підключати в різні частини програми. Але один `Activity` може містити кілька `fragment`. Фрагменти з'явилися у `API 11 (Android 3.0)`

Особливість `Fragment`-ів у тому, що ви можете розбити зовнішній вигляд на блоки і потім підключати їх для відображення на різних пристроях по своєму (смартфон/планшет). Також ми матимемо можливість перевикористання блоків (`Fragments`).

4.2.2 Конструювання алгоритмів методів програмних класів або процедур/функцій

Метод `Register_user` для реєстрації користувача можна побачити на рис 4.3. Він викликається шляхом натискання на кнопку «Зареєструватися» на сторінці реєстрації, після вводу користувачем його даних. Проходить перевірка на правильність вводу даних та, якщо є помилки виводиться, де саме вони сталися, а якщо помилок нема – користувача реєструє, його дані зберігаються, а на екран виводиться повідомлення про успішну реєстрацію.

```

public void onViewCreated(@NonNull View view, Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);

    EditText email = (EditText) view.findViewById(R.id.EmailAddress_input2);
    EditText password = (EditText) view.findViewById(R.id.Password_input2);
    EditText username = (EditText) view.findViewById(R.id.username_input2);
    String emailPattern = "[a-zA-Z0-9._-]+@[a-z]+\\.+[a-z]+";

    view.findViewById(R.id.registration_button2).setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if (email.getText().toString().matches(emailPattern)) {
                if (password.getText().toString().length() >= 8) {
                    if (username.getText().toString().length() >= 3) {
                        Toast myToast = Toast.makeText(getActivity(), text: "Приветствую, пользователь!", Toast.LENGTH_SHORT);
                        myToast.show();
                        SharedPreferences sharedPref = getActivity().getPreferences(Context.MODE_PRIVATE);
                        SharedPreferences.Editor editor = sharedPref.edit();
                        editor.putString("email", email.getText().toString());
                        editor.putString("password", password.getText().toString());
                        editor.putString("username", username.getText().toString());
                        editor.apply();
                        NavHostFragment.findNavController( fragment RegistrationFragment.this)
                            .navigate(R.id.action_RegistrationFragment_to_FirstFragment);
                    } else {
                        Toast myToast = Toast.makeText(getActivity(), text: "Слишком короткое имя пользователя.", Toast.LENGTH_SHORT);
                        myToast.show();
                    }
                } else {
                    Toast myToast = Toast.makeText(getActivity(), text: "Слишком короткий пароль.", Toast.LENGTH_SHORT);
                    myToast.show();
                }
            } else {
                Toast myToast = Toast.makeText(getActivity(), text: "Неправильно введен адрес электронной почты.", Toast.LENGTH_SHORT);
                myToast.show();
            }
        }
    });
}

```

Рис 4.3 – метод реєстрації

Auth_user

Метод Auth_user для авторизації користувача можна побачити на рис 4.4. Він викликається шляхом натискання на кнопку «Войти» на сторінці входу, після вводу користувачем його даних. Проходить перевірка на правильність вводу даних та, якщо є помилки виводиться, де саме вони сталися, а якщо помилок нема – користувача авторизує, а на екран виводиться повідомлення про успішну авторизацію.

```

public void onViewCreated(@NonNull View view, Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);

    EditText email = (EditText) view.findViewById(R.id.EmailAddress_input);
    EditText password = (EditText) view.findViewById(R.id.Password_input);
    String emailPattern = "[a-zA-Z0-9._-]+@[a-z]+\\.+[a-z]+";
    SharedPreferences sharedPref = getActivity().getPreferences(Context.MODE_PRIVATE);
    SharedPreferences.Editor editor = sharedPref.edit();
    String userEmail = sharedPref.getString("email", "");
    String userPassword = sharedPref.getString("password", "");

    view.findViewById(R.id.login_button).setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if (email.getText().toString().matches(emailPattern)){
                if (password.getText().toString().length() >= 8){
                    if (email.getText().toString().equals(userEmail) && password.getText().toString().equals(userPassword)) {
                        Toast myToast = Toast.makeText(getActivity(), text: "Приветствую, пользователь!", Toast.LENGTH_SHORT);
                        myToast.show();
                        NavHostFragment.findNavController( fragment: FirstFragment.this)
                            .navigate(R.id.action_FirstFragment_to_SecondFragment);
                    } else {
                        Toast myToast = Toast.makeText(getActivity(), text: "Неправильно введены данные.", Toast.LENGTH_SHORT);
                        myToast.show();
                    }
                }
            } else {
                Toast myToast = Toast.makeText(getActivity(), text: "Слишком короткий пароль.", Toast.LENGTH_SHORT);
                myToast.show();
            }
        }
    });
}

```

Рис 4.4 – метод авторизації

Edit_user

Метод Edit_user для редагування профіля користувача можна побачити на рис 4.5. Він викликається шляхом натискання на кнопку «Сохранить изменения» на сторінці профілю, після зміни користувачем його даних. Проходить перевірка на правильність вводу даних та, якщо є помилки виводиться, де саме вони сталися, а якщо помилок нема – проходить зміна даних користувача, а на екран виводиться повідомлення про успішне редагування.

```

public void onViewCreated(@NonNull View view, Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);

    SharedPreferences sharedPref = getActivity().getPreferences(Context.MODE_PRIVATE);
    SharedPreferences.Editor editor = sharedPref.edit();
    String userEmail = sharedPref.getString("email", "");
    String userPassword = sharedPref.getString("password", "");
    String userUsername = sharedPref.getString("username", "");

    EditText email = (EditText) view.findViewById(R.id.EmailAddress_input3);
    EditText password = (EditText) view.findViewById(R.id.Password_input3);
    EditText username = (EditText) view.findViewById(R.id.username_input3);
    EditText bio = (EditText) view.findViewById(R.id.about_user_input);
    String emailPattern = "[a-zA-Z0-9._-]+@[a-z]+\\.+[a-z]+";

    email.setText(userEmail);
    password.setText(userPassword);
    username.setText(userUsername);

    view.findViewById(R.id.login_button3).setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if (email.getText().toString().matches(emailPattern)) {
                editor.putString("email", email.getText().toString());
                editor.apply();
                Toast myToast = Toast.makeText(getActivity(), "Данные изменены", Toast.LENGTH_SHORT);
                myToast.show();
            } else {
                Toast myToast = Toast.makeText(getActivity(), "Неправильно введен адрес электронной почты.", Toast.LENGTH_SHORT);
                myToast.show();
            }
            if (password.getText().toString().length() != 0) {
                if (password.getText().toString().length() <= 16) {
                    if (password.getText().toString().length() >= 8) {
                        editor.putString("password", password.getText().toString());
                        editor.apply();
                        Toast myToast = Toast.makeText(getActivity(), "Данные изменены", Toast.LENGTH_SHORT);
                        myToast.show();
                    } else {
                        Toast myToast = Toast.makeText(getActivity(), "Слишком короткий пароль.(минимум 8 символов)", Toast.LENGTH_SHORT);
                        myToast.show();
                    }
                } else {
                    Toast myToast = Toast.makeText(getActivity(), "Слишком длинный пароль.(максимум 16 символов)", Toast.LENGTH_SHORT);
                    myToast.show();
                }
            }
            if (bio.getText().toString().length() != 0) {
                if (bio.getText().toString().length() <= 300) {
                    editor.putString("About user", bio.getText().toString());
                    editor.apply();
                    Toast myToast = Toast.makeText(getActivity(), "Данные изменены", Toast.LENGTH_SHORT);
                    myToast.show();
                } else {
                    Toast myToast = Toast.makeText(getActivity(), "Слишком длинная информация о пользователе.(максимум 300 символов)", Toast.LENGTH_SHORT);
                    myToast.show();
                }
            }
            if (username.getText().toString().length() >= 3) {
                if (username.getText().toString().length() <= 30) {
                    editor.putString("username", username.getText().toString());
                    editor.apply();
                    Toast myToast = Toast.makeText(getActivity(), "Данные изменены", Toast.LENGTH_SHORT);
                    myToast.show();
                } else {
                    Toast myToast = Toast.makeText(getActivity(), "Слишком длинное имя пользователя.(максимум 30 символов)", Toast.LENGTH_SHORT);
                    myToast.show();
                }
            } else {
                Toast myToast = Toast.makeText(getActivity(), "Слишком короткое имя пользователя.(минимум 3 символа)", Toast.LENGTH_SHORT);
                myToast.show();
            }
        }
    });
}

```

Рис 4.5 – метод редагування даних

Payment

Метод Payment для покупки курсу користувачем можна побачити на рис 4.6. Він викликається шляхом натискання на кнопку «Купити курс» на сторінці курсу. Проходить перевірка на правильність вводу даних та, якщо є помилки виводиться, де саме вони сталися, а якщо помилок нема – проходить оплата, а на екран виводиться повідомлення про успішну покупку.

```
public void onViewCreated(@NonNull View view, Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);

    EditText CardNumber = (EditText) view.findViewById(R.id.CardNumber);
    EditText timeNumber = (EditText) view.findViewById(R.id.timeNumber);
    EditText cvvNumber = (EditText) view.findViewById(R.id.cvvNumber);

    view.findViewById(R.id.payment_button).setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if (CardNumber.getText().toString().length()==16){
                if (timeNumber.getText().toString().length()==4){
                    if (cvvNumber.getText().toString().length()==3){
                        NavHostFragment.findNavController( fragment: PaymentFragment.this)
                            .navigate(R.id.action_PaymentFragment_to_SecondFragment);
                        Toast myToast = Toast.makeText(getActivity(), text: "Оплата прошла успешно.", Toast.LENGTH_SHORT);
                        myToast.show();
                    }
                    else{
                        Toast myToast = Toast.makeText(getActivity(), text: "Неправильно введен CVV2 код", Toast.LENGTH_SHORT);
                        myToast.show();
                    }
                }
                else{
                    Toast myToast = Toast.makeText(getActivity(), text: "Неправильно введена дата с карты", Toast.LENGTH_SHORT);
                    myToast.show();
                }
            }
            else{
                Toast myToast = Toast.makeText(getActivity(), text: "Неправильно введен номер карты", Toast.LENGTH_SHORT);
                myToast.show();
            }
        }
    });
}
```

Рис 4.6 – метод покупки курсу

4.2.3 Особливості використання спеціалізованих програмних бібліотек та API

Макетом для розміщення на екрані було обрано ConstraintLayout.

Новий макет ConstraintLayout з'явився в Android Studio 2.2 і доступний для пристроїв з версії Android 2.3.

При його використанні немає сенсу використовувати XML-подання, тільки в режимі Design, коли ви можете спонукати всі компоненти у візуальному редакторі.

При виборі компонента спостерігається картина, як на рис. 4.7

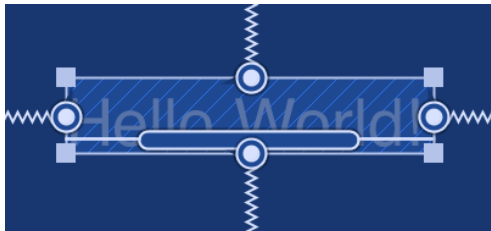


Рис. 4.7 – приклад ConstraintLayout.

Квадратні опорні точки у кутах компонента дозволяють змінювати його розміри. Круглі опорні точки по краях дозволяють керувати відступами країв екрана та інших компонентів.

5 Верифікація програмного продукту

5.1 Тестування апаратно-програмних інтерфейсів програмного продукту

Проведено тестування ПП на рівні таких інтерфейсів:

- а) інтерфейси із зовнішніми пристроями (обладнанням), наприклад, апаратні особливості комп'ютерної техніки, периферійних пристроїв;
- б) програмні інтерфейси, наприклад, версії операційних систем, версії веб навігаторів, спеціалізовані програмні бібліотеки.

Результати тестування представлено у вигляді таблиці 5.1

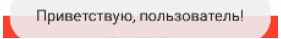
Таблиця 5.1 – Опис умов тестування

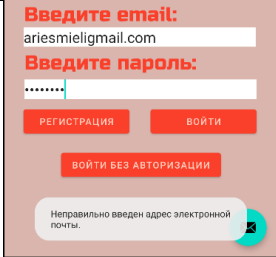
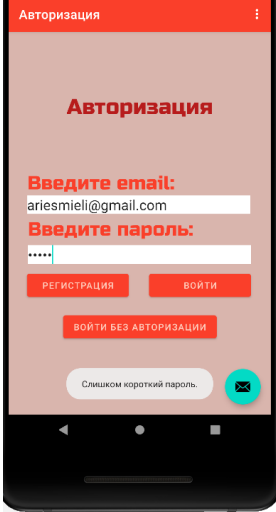
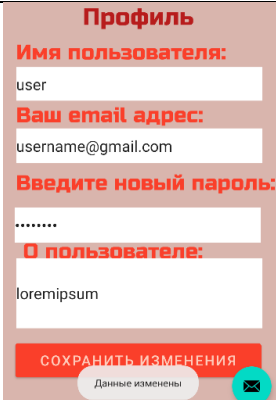
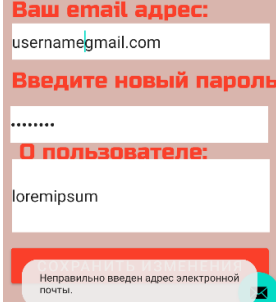
Тип умови (Hard/Soft)	Опис вимог	Опис реальних умов
Soft	Android 5.0+	Android 11

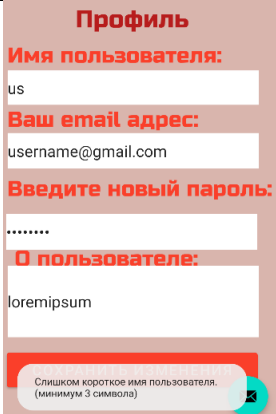
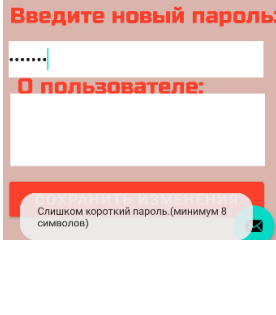

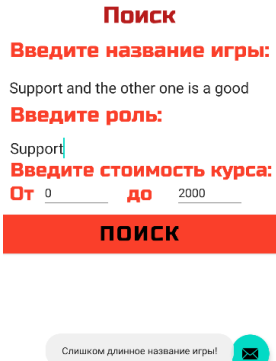
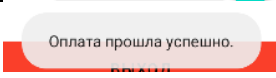
5.2 Тестування інтерфейсу користувача програмного продукту

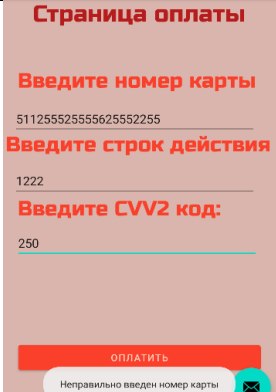
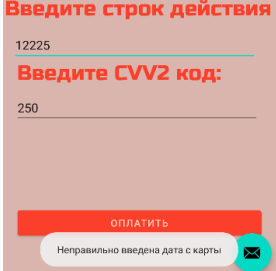
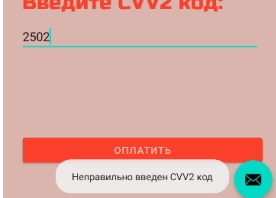
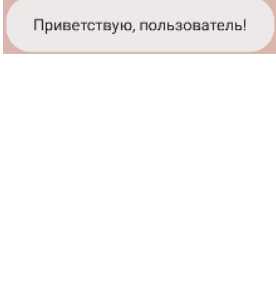
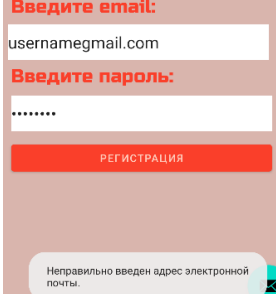
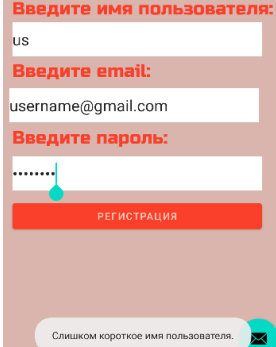
У таблиці 5.2 представлено опис та відповідні скріншоти процедури проведення функціонального тестування.

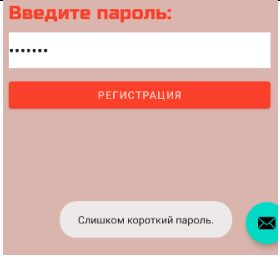
Таблиця 5.2 – Функціональне тестування.

FR ID	Test Case ID	Опис значень вхідних даних до методу (процедури, функції)	Опис очікуваних значень результату виконання методу	Опис очікуваних результатів взаємодії з ПП на рівні інтерфейсу користувача	Результат тестування (Passed / Failed/ Blocked)
FR1	TC1	Email="ariesmieli@gmail.com" password="admin123"	Користувача авторизовано, Повідомлення "Приветствую, пользователь!"		Passed

FR1	TC2	Email="ariesmiel igmail.com" password="admin 123"	Повідомлення "Неправильно написан адрес електронної пошти!"		Passed
FR1	TC3	Email="ariesmie li@gmail.com" password="admi n"	Повідомлення "Слишком короткий пароль!"		Passed
FR2	TC1	username = user bio = loremipsum email = <u>username@gmai l.com</u> password = 12345678	Данні змінено, Повідомлення "Данные изменены!"		Passed
FR2	TC2	username = user bio = loremipsum email = username@gmail.co m password = 12345678	Повідомлення "Неправильно написан адрес електронної пошти!"		Passed

FR2	TC3	username = us bio = loremipsum email = <u>username@gmail.com</u> password = 12345678	Повідомлення “Слишком короткое имя пользователя(минимум 3 символа)!”		Passed
FR2	TC4	username = user bio = loremipsum email = username@gmail.com password = 1234567	Повідомлення “Слишком короткий пароль.(миним ум 8 символов)”		Passed
FR3	TC1	Назва гри=“League of Legends” Роль=“Support” Вартість=“0 - 1000”	Виведення курсів, що підходять за фільтрами, Повідомлення “Вот курсы по вашему запросу!”		Passed
FR3	TC2	Назва гри=“League of Legends Support and the other one is a good” Роль=“Support” Вартість=“0 - 1000”	Помилка, повідомлення “Слишком длинное название игры!”		Passed
FR4	TC1	Cardnumber=“5 16875590016830 9” Date=“1222” CVV=“133”	Успішна оплата, Повідомлення “Оплата прошла успешно.”		Passed

FR4	TC2	Cardnumber="5168755900168309213123" Date="1222" CVV="133"	Помилка, повідомлення "Неправильно введен номер карты!"		Passed
FR4	TC3	Cardnumber="5168755900168309" Date="12223" CVV="133"	Помилка, Повідомлення "Неправильно введена дата с карты!"		Passed
FR4	TC4	Cardnumber="5168755900168309213123" Date="12/22" CVV="1334"	Помилка, повідомлення "Неправильно введен CVV2 код!"		Passed
FR5	TC1	username = user email = <u>username@gmail.com</u> password = 12345678	Користувача авторизовано, Повідомлення "Приветствую, пользователь!"		Passed
FR5	TC2	username = user email = username@gmail.com password = 12345678	Повідомлення "Неправильно написан адрес электронной почты!"		Passed
FR5	TC3	username = us email = <u>username@gmail.com</u> password = 12345678	Повідомлення "Слишком короткое имя пользователя!"		Passed

FR5	TC4	username = user email = <u>username@gmail.com</u> password = 1234567	Повідомлення “Слишком короткий пароль!”		Passed
-----	-----	--	--	---	--------

5.3 Тестування часу реакції програмного продукту на дії користувача

Проведено тестування часу реакції ПП на дії користувача у відповідності з таблицею результатів аналізу характеристик продуктивності. Для перевірки кожної вимоги проведено декілька тестів, щоб відносна похибка вимірювань не перевищувала 10%. Результати тестування представлено у вигляді таблиці 5.3.

Таблиця 5.3 – Тестування часу реакції ПП на дії користувача

FR (назва)	Максимальний час реакції ПП на дії користувачів, сек	Кількість проведених тестів	Результат тестування, сек (середнє значення)	Відносна похибка вимірювань, %
FR1 Авторизація користувача	0.2	3	0.19	5
FR2 Редагування профілю користувача	0.22		0.2	9
FR3 Пошук курсів	0.32		0.31	1.1
FR4 Оплата курсу	0.25		0.23	8
FR5 Реєстрація користувача	0.67		0.6	7

6 Розгортання та валідація програмного продукту

6.1 Інструкція з встановлення системного програмного забезпечення

1. Для встановлення програмного продукту на Android необхідно встановити його встановлюючий файл з розширення apk. Його можна знайти за посиланням
<https://drive.google.com/file/d/1KOOhbQp1QuADxlfSfd1al0McKMxlhZNe/view?usp=sharing>

Apk - формат файлу програми Android, використовуваний операційною системою Android та безліччю інших операційних систем на базі Android для поширення та встановлення мобільних додатків, мобільних ігор та проміжного програмного забезпечення.

2. Після встановлення його собі на пристрій, необхідно зайти у Провідник, і знайти там цей файл. Зазвичай, при скачуванні файлу з мережі інтернет його можна знайти у папці downloads(як на рисунку 6.1).

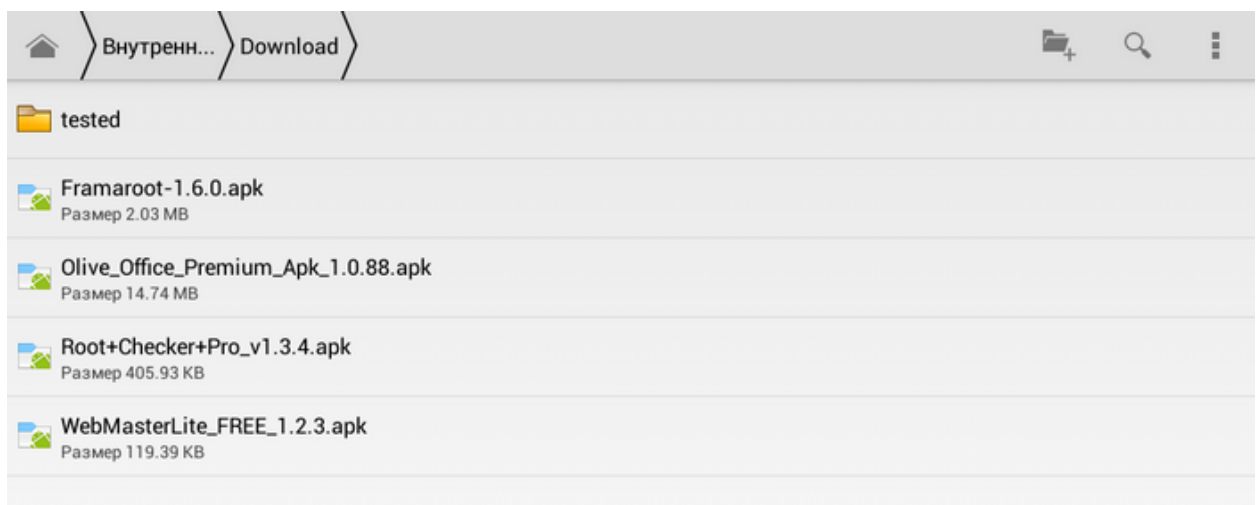


Рисунок 6.1 – папка downloads.

3. Під час запуску файлу система видасть інформаційне повідомлення в якому перерахує дозволи яких вимагає програма, що встановлюється(як на рисунку 6.2):

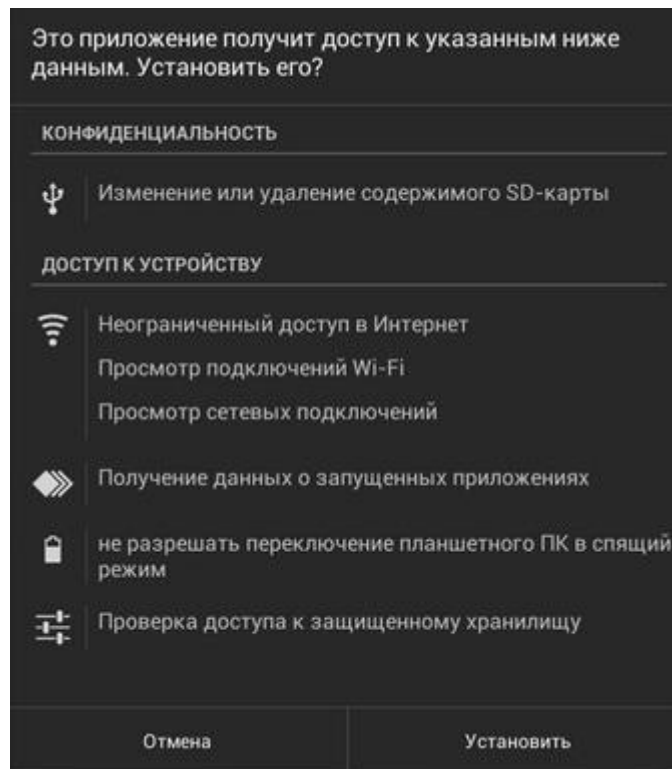


Рисунок 6.2 – запити від системи.

Після надання доступу, програма буде встановлена на ваш пристрій.

6.2 Інструкція з використання програмного продукту

Авторизація користувача

Пункт	Опис
Прецедент	Авторизувати користувача
Передумова початку виконання прецеденту	відкриття застосунку на пристрої
Актори як зацікавлені особи у виконанні прецеденту	гість, користувач
Актор-основна зацікавлена особа як ініціатор початку прецеденту	гість
Гарантія успіху	отримання доступу до повного функціоналу застосунку

Приклад основного успішного сценарію авторизації користувача

1. ПП запитує у гостя його параметри авторизації (адреса ел.пошти та пароль)
2. Гість передає ПП свої ІА-параметри (адреса ел.пошти та пароль)
3. ПП надає авторизованому користувачу доступ до прецедентів ПП.

Обмеження:

обмеження поля адреси ел. пошти – повинно містити адресу ел. пошти
обмеження паролю – довжина від 8 символів, повинен містити велику літеру та спеціальний символ

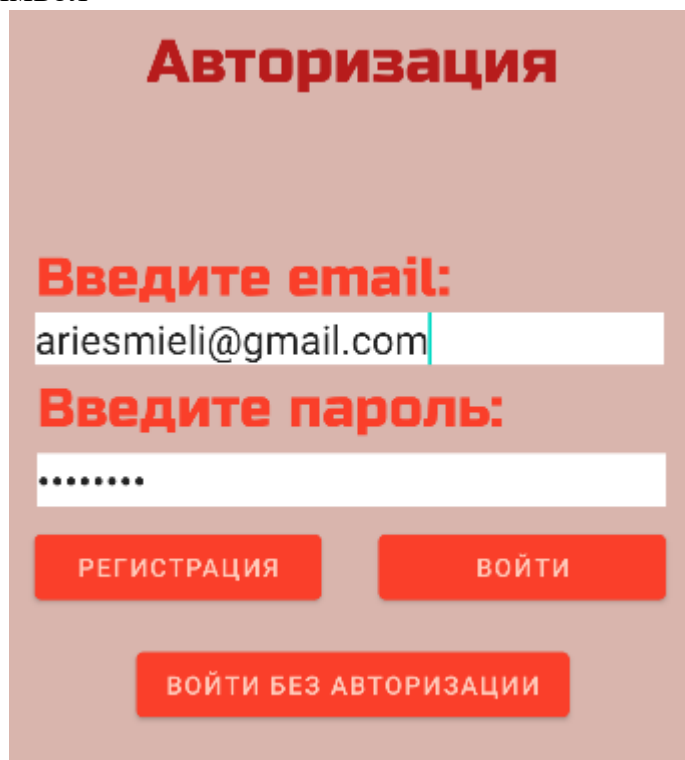


Рисунок 6.3 – Коректно введена інформація для авторизації користувача
Редагування профілю авторизованого користувача

Пункт	Опис
Прецедент	Редагування профілю авторизованого користувача
Передумова початку виконання прецеденту	успішна авторизація
Актори як зацікавлені особи у виконанні прецеденту	авторизований користувач
Актор-основна зацікавлена особа як ініціатор початку прецеденту	авторизований користувач
Гарантія успіху	оновлена інформація про користувача відповідно до його вподобань

Приклад основного успішного сценарію редагування профілю авторизованого користувача

1. ПП запитує у користувача, які параметри персональної інформації він хоче змінити (ім'я, інформація про себе (короткий текст 250 символів), пароль, адреса ел.пошти)

2. Користувач обирає параметри і передає ПП нові дані

Обмеження:

обмеження поля біо – повинно містити до 250 символів

обмеження поля адреси ел. пошти – повинно містити адресу ел. пошти

обмеження поля імені користувача – повинно містити від 0 до 30 символів

обмеження паролю – довжина від 8 символів

The screenshot shows a form titled 'Профиль' (Profile) with the following fields and values:

- Имя пользователя:** user
- Ваш email адрес:** username@gmail.com
- Введите новый пароль:**
- Пользователь:** loremipsum

At the bottom, there is a red button labeled 'СОХРАНИТЬ ИЗМЕНЕНИЯ' (Save changes) and a status message 'Данные изменены' (Data changed) with a green checkmark icon.

Рисунок 6.4 - Коректно введені дані

The screenshot shows a form titled 'Ваш email адрес:' with the following fields and values:

- Ваш email адрес:** username@gmail.com
- Введите новый пароль:**
- Пользователь:** loremipsum

At the bottom, there is a red button labeled 'СОХРАНИТЬ ИЗМЕНЕНИЯ' (Save changes) and a status message 'Неправильно введен адрес электронной почты.' (Incorrect email address entered) with a red error icon.

Рисунок 6.5 – некоректно введені дані електронної пошти та повідомлення про відповідні порушення

3. ПП перевіряє, оновлює і зберігає персональні дані у профілі користувача
Приклад основного успішного сценарію прецеденту ПП «Пошук нового курсу»

Пункт	Опис
Прецедент	Оплата нового курса
Передумова початку виконання прецеденту	Успішний запуск програми
Актори як зацікавлені особи у виконанні прецеденту	Гість, авторизований користувач

Актор-основна зацікавлена особа як ініціатор початку прецеденту	Гість
Гарантія успіху	Знаходження інформації за введеними фільтрами

1. Користувач заходить у програму та відкриває вікно пошуку без авторизації.
2. ПП запитує у користувача, які параметри він хоче використовувати для пошуку.

3. Користувач вводить необхідні для нього параметри

Обмеження:

обмеження поля назви гри – повинно містити до 30 символів

обмеження поля ролі – повинно містити до 20 символів

обмеження вартості – вартість від 0 до 2000 гривень

4. Користувач запускає функцію пошуку по курсам.

Поиск

Введите название игры:

Support and the other one is a good

Введите роль:

Support

Введите стоимость курса:

От 0 до 2000

ПОИСК

Слишком длинное название игры!

Рисунок 6.6 – коректно введені дані

назви гри, ролі та вартості курсу

Курсы

Курсы

LEAGUE OF LEGENDS SUPPORT

Вот курсы по вашему запросу.

Рисунок 6.7 – знайдені курси за запитом

5. Програма робить обробку по фільтрам та видає результат, якого чекає користувач.

Приклад основного успішного сценарію прецедента «Оплата нового курсу»

1. Користувач знаходить необхідний для нього курс через пошук використовуючи фільтри і обирає варіант покупки курсу.
2. Програма запитує у користувача номер картки та власні дані для підтвердження замовлення.

Обмеження:

Номер карти повинен мати 16 цифр

Срок дії повинен бути введений у форматі ММ/РР, та ММ від 0 до 12, РР від 21 до 35.

CVV2 code повинен мати 3 цифри

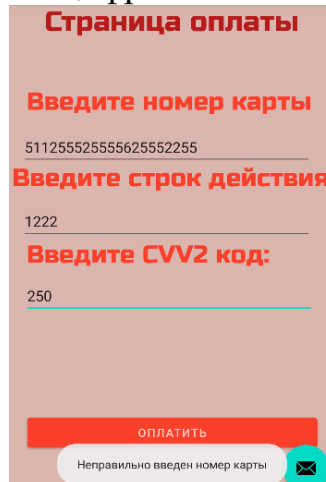


Рисунок 6.8 – пример заполнения страницы оплаты, с неправильно введенным номером карты

3. Програма отримує необхідну суму і надає користувачу інформацію про те, що курс було успішно оплачено.

6.3 Результати валідації програмного продукту

Метою створення програмного продукту було підвищення рівня знань гравця про гру, в яку він грає шляхом вивчення механік, вмінь, предметів та інших особливостей.

Протестованими гравцями були 3 студенти 1-2 курсів, віком від 18 до 19 років, які раніше дуже мало грали у League of Legends. Їм було запропоновано інтенсивний курс, що підвищив їх рівень знань у грі League of Legends.

$$\text{Рівень знань} = K/(N*A),$$

де N – кількість персонажів;

A – Abilities, тобто вміння персонажів;

K – knowledge, тобто вже знайомі вміння.

Початковий рівень знань гравців:

$$\text{Данило: } 118/(152*4)=19.4\%$$

$$\text{Ігор: } 143/(152*4)=23.5\%$$

$$\text{Аюр: } 67/(152*4)=11\%$$

Знання після проходження курсу:

Данило: $168/(152*4)=27.6\%$

Ігор: $245/(152*4)=40.2\%$

Аюр: $203/(152*4)=33.3\%$

Знання підвищились з 18 до 33%, що є досить непоганим результатом за невеликий час. Дивлячись на те, що гра постійно оновлюється, бажаним рівнем знань про гру у гравця однієї ролі є не більше 60%, що і є цифрою, до якої гравців повинен привести повний курс.

Висновки

В результаті виконання етапів курсової роботи було створено програмне забезпечення процесу покупки ігрових курсів користувачами, що бажають підвищити свій рівень знань у іграх.

Приклад використання програмного забезпечення у відповідності з інструкцією користувача представлено у відеозапису, доступним за посиланням <https://drive.google.com/file/d/14NdMb2169VzIFSJ17M37ktkwMqPDCbnt/view?usp=sharing>

Створене програмне забезпечення досягло наступної мети його споживача: Підвищення рівня знань гравця про гру.

Доказом цього є наступні факти:

- 1) Після проходження курсу гравці мали більше знань про гру.
- 2) Завдяки курсу, що гравці пройшли, їх внутрішньо ігровий ранг підвищився, що свідчить про підвищення рівня усвідомленості.

Вказані факти перетворили програмне забезпечення на програмний продукт.

В процесі створення програмного продукту виникли такі труднощі (організаційні, проблеми відсутності досвіду, знань, потрібних в різних етапах):

- 1) Відсутність знань про програмування на Android
- 2) Відсутність досвіду програмування на Java
- 3) Недостатня кількість людей, бажаючих провести тести ПП.

Через вищеописані непередбачені труднощі, а також через обмежений час на створення програмного продукту, залишаються нереалізованими такі прецеденти або їх окремі кроки роботи:

- 1) Залишення відгуків про курси
- 2) Окремі профілі для користувачів та для викладачів
- 3) Залишення відгуків про викладачів
- 4) Можливість листування
- 5) Реалізована база даних користувачів та курсів

Зазначені недоробки планується реалізувати в майбутніх курсових роботах з урахуванням тем дисциплін наступних семестрів.