МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені Тараса Шевченка ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра програмних систем і технологій

Дисципліна

«АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»

Лабораторна робота № 8

«ТЕСТУВАННЯ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»

на тему:

«Автоматизована система безготівкових електронних платежів»

Виконав:	Гоша Д.О	**Перевірив**:	Юрчук І. А.
Група	ІПЗ-23	Дата перевірки	
Форма навчання	Денна	Бали	
Спеціальність	121		
2021			

Мета: Дослідити методи тестування вимог до програмного забезпечення.

Завдання

1. Здійснити перевірку документації, створеної у лабораторній роботі 2.4, за допомогою checklist.
2. Здійснити аналіз поведінки системи за допомогою достатнього покриття системи тест-кейсами та юз-кейсами, як з позитивними так і з негативними сценаріями.

Виконання завдання:

Перевірка документації.

На основі отриманих у попередніх лабораторних роботах артефактів специфікації вимог, зокрема документацій, здійснимо їх перевірку за допомогою технології Checklist. Таким чином, будемо досліджувати вимоги за такими параметрами:

1. Повнота - документування вимоги у найбільш можливій деталізації. Здебільшого, визначається використанням поглиблених методів збору вимог (для поточної системи – інтерв'ю з усіма типами користувачів).
2. Відслідковуваність - документування вимоги з таким рівнем деталізації, що можна чітко визначити чи виконана вимога, чи ні. Здебільшого, визначається поглибленою специфікацією (для поточної системи – різні види структурних та функціональних діаграм). З
3. Важливість - документування вимоги з точки зору чіткого визначення обов'язковості слідування їй. Здебільшого, визначається поглибленою специфікацією при розгляді системи як моноліту.
4. Зрозумілість - документування вимоги з таким рівнем деталізації, що чітко зрозуміла кінцева мета виконання вимоги. Здебільшого, визначається використання методів збору вимог.
5. Однозначність - документування вимог з таким рівнем деталізації, що були ліквідовані всі неоднозначні аспекти вимог. Здебільшого, визначається використанням методів збору вимог та їх специфікацією.
6. Вимірюваність - документування вимог з таким рівнем деталізації, що

чітко можна визначити етапи виконання окремої вимоги. Здебільшого, визначається використанням методів збору вимог та їх специфікацією. Таким чином, визначимо матрицю відповідності вимог, що були наведені у документі "Технічне завдання", який був розроблений під час ЛР 1.5.

	Повнота	Відслідковуваність	Важливість	Зрозумілість	Однозначність	Вимірюваність
Вимоги до повторного використання реалізації або компонентів програми або системи (Reusability).	✓	✗	✓	✓	✓	✗
Вимоги до розширення (Extensibility)	✓	✓	✓	✓	✓	✗
Вимоги до міграції (Portability)	✓	✓	✓	✗	✓	✓
Вимоги до взаємодії між компонентами рішення, між зовнішніми компонентами, використання стандартних протоколів та технологій взаємодії (Interoperability)	✓	✓	✓	✓	✓	✓
Вимоги до підтримки системи або програми (Supportability)	✗	✓	✓	✓	✓	✓
Вимоги до модульності програми або системи (Modularity)	✓	✓	✓	✓	✓	✓
Вимоги до можливості тестування (Testability)	✓	✓	✓	✗	✓	✓

Вимоги до можливості та простоти локалізації (Localizability)	✓	✓	✓	✓	✕	✓
Доступність	✓	✓	✓	✓	✓	✕
Надійність	✓	✓	✓	✓	✕	✓
Масштабованість	✓	✓	✕	✓	✓	✓
Вимоги до зручності використання системи/додатку	✓	✓	✓	✓	✕	✓
Вимоги до продуктивності рішення	✓	✓	✓	✓	✓	✕
Обмеження	✓	✓	✓	✓	✓	✓

Аналіз поведінки системи.

Для перевірки роботи системи у відповідності до специфікованих вимог, здійснимо аналіз поведінки на основі покриття основних аспектів системи тест- та юз-кейсами. Таким чином, були отримані такі тест-кейси:

Специфікація тестового варіанту	
Назва взаємодіючих класів: Користувач	**Назва тесту:** CheckAutorization_onClick
Опис тесту: перевірка правильності встановлення значення елементу UI Login після натискання на нього та спроби авторизації користувача у системі.	
Початкові умови: браузер користувача відповів на запит, та відобразив веб сторінку застосунку.	
Очікуваний результат: користувач успішно авторизується у системі та отримає доступ до сторінки з персональною інформацією, балансом тощо.	

Отриманий результат: користувач успішно авторизувався та перейшов на головну сторінку застосунку.

Специфікація тестового варіанту	
Назва взаємодіючих класів: Користувач, Транзакція, Баланс	**Назва тесту:** CreateTransaction_onClick
Опис тесту: перевірка правильності роботи елементу, що відповідає за створення транзакції. Перевірка полів, з інформацією про транзакції та достатньої суми на балансі.	
Початкові умови: користувач авторизувався, перейшов на сторінку створення транзакції та намагається заповнити дані.	
Очікуваний результат: користувач правильно введе дані , у нього буде достатній баланс та елемент відпрацює правильно.	
Отриманий результат: користувач ввід дані, відповідно залежно від правильності та бажаної суми на балансі та у переказі додаток виведе повідомлення згідно зі станом опрацювання транзакції.	

Специфікація тестового варіанту	
Назва взаємодіючих класів: Адміністратор, БД	**Назва тесту:** InsertScreenButtonInsert_CorrectFields_OnClick
Опис тесту: перевірка того, що при натисканні на елемент UI btnInsert здійснюється запиту до серверу.	
Початкові умови: значення будь-якого обов'язкового з полів введення, які відповідають елементам UI типу OutlinedTextBox, не є пустим рядком.	
Очікуваний результат: буде здійснено запит зі значенням полів, що відповідають відповідним значенням елементів UI, та здійснений перехід до стану "InsertScreen з анімацією завантаження"	
Отриманий результат: здійснений запит зі значенням полів, що	

відповідають відповідним значенням елементів UI, та здійснений перехід до стану "InsertScreen з анімацією завантаження".

До того ж, були отримані такі юз-кейси:

Назва	Перевірка авторизації користувача
Діючі особи	Користувач
Мета	Перевірка правильності встановлення значення елементу UI Login після натискання на нього та спроби авторизації користувача у системі.
Передумова	Браузер користувача відповів на запит, та відобразив веб сторінку застосунку
Успішний сценарій: 1. Клієнт вводить логін 2. Клієнт вводить пароль 3. Натискає на кнопку, що відповідає заавторизацію 4. Серверзвязуються з БД та перевіряє дані користувача 5. Редірект на персональну сторінку	
Результат	У клієнта з’являється можливість доступцу до персональної інформації та створювати перекази

Назва	Створення транзакції
Діючі особи	Користувач
Мета	Перевірка правильності стоврення транзакції а саме: перевірка даних,

	що надійшли від користувача, обробка цих даних додатком та повернення відповіді користувачу.
Передумова	Користувач авторизован та надіслав перні дані до транзакції.
Успішний сценарій: 1. Клієнт надіслав дані, як правильно надійшли до серверу. 2. Застосунок перевірив ці дані а саме: a. Чи надіслані по захищеному каналу b. Чи у користувача достатньо коштів на балансі c. Чи користувач існує в системі та реквізити валідні 3. Застосунок успішно провів транзакції 4. Застосунок повернув влучну відповідь про стан транзакції(на будь якому етапі, у разі успіху – на останньому)	
Результат	Користувач провів успішно транзакції, застосунок вніс до БД стан транзакції та інформації про неї.

Назва	Запит до БД
Діючі особи	Адміністратор
Мета	Перевірка або редагування інформації у базі даних.
Передумова	До бази було додано певну інформацію. Вона справна та відповідає на запити.
Успішний сценарій: 1. Адміністратор БД натискає на поле введення; вводить id слова, яке	

бажає отримати. 2. Адміністратор БД натискає на кнопку здійснення запиту. 3. Відповідно до API, формується запит до серверу. 4. Сервер отримує запит на основі API; формує відповідь. 5. Сервер відправляє відповідь до застосунку. 6. Інформаційні поля заповнюються відповідно до отриманої відповіді серверу. 7. Адміністратор БД оновлює значення полів відповідно до поставленого завдання. 8. Адміністратор БД натискає на відповідний елемент UI для формування запиту. 9. Відповідно до API, формується запит до серверу. 10. Сервер отримує запит на основі API та виконує його	
Результат	Оновлюється певна інформація у БД.

Висновки

Тестування вимог дозволяє зменшити кількість допрацювань і змін. Перед початком тестування вимог необхідними є наступні умови: вимоги проаналізовані й задокументовані, тобто є щось, з чим можемо працювати. Аналітик, як роль, самостійно проводить певну оцінку та перевірку цих вимог. Під час тестування вимог відбувається наступний процес: всі зацікавлені особи підтверджують, що вимоги коректні, зрозумілі і достатні для того, щоб розпочинати роботу. А користувачі підтверджують, що вимоги відображають їхні потреби в роботі. Основним критерієм готовності вимог є те, чи містять вимоги достатньо інформації для того, щоб розпочинати розробку. І важливо відзначити, що і замовник, і користувачі, і проектна команда, якою планується робота над проектом, — всі беруть участь у тестуванні вимог і враховують різні пов’язані з тестуванням вимог аспекти.

У цій лабораторній були здобуті навички з перевірки документації, створеної у лабораторній роботі 1.4, за допомогою checklist. Здійснено аналіз поведінки системи за допомогою достатнього покриття системи тест-кейсами та юз-кейсами, як з позитивними так і з негативними сценаріями.