Міністерство освіти і науки України
Державний університет
інформаційно-комунікаційних
технологій

Золотухіна О.А, Худік Б.О.

UX/UI дизайн. Проєктування та розробка інтерфейсу користувача

Навчальний посібник

Київ - 2025

УДК 004.5

Рецензенти:

Ольга ТКАЧЕНКО
д.т.н, професор, професор кафедри
програмних систем і технологій
Київського національного університету
імені Тараса Шевченка

Олексій ШУШУРА
д.т.н, доцент, професор кафедри
цифрових технологій в енергетиці
Національного технічного університету України
«Київський політехнічний інститут імені Ігоря Сікорського»

Рекомендовано до видання
Вченою радою Державного університету
інформаційно-комунікаційних технологій
(протокол № 8 від 17 червня 2025 року)

004.5 (075.8)
З-81

Золотухіна О.А., Худік Б.О. UX/UI дизайн. Проєктування та розробка інтерфейсу користувача. Навчальний посібник. К.:ДУІКТ, 2025. – 120 с.

Посібник призначено для вивчення окремих тем дисциплін, пов’язаних з проєктуванням та розробкою інтерфейсу користувача. Посібник охоплює змістовні модулі, присвячені особливостям взаємодії людини та комп’ютерної системи, основам проєктування та розробці інтерфейсу, орієнтованого на користувача, та питанням оцінювання якості інтерфейсу з точки зору usability.

Посібник може бути використаний студентами спеціальностей галузі знань F – «Інформаційні технології» першого (бакалаврського) рівня вищої освіти при вивченні відповідних дисциплін з проєктування та розробки інтерфейсу користувача, а також усіма, хто самостійно опановує основи проєктування користувацьких інтерфейсів.

ЗМІСТ

ПЕРЕДМОВА

Сучасний цифровий світ визначається стрімким розвитком інтерфейсів взаємодії між людиною та інформаційними системами. У межах цього процесу зростає роль UX/UI-дизайну як міждисциплінарної галузі, що поєднує знання з психології, інформатики, когнітивістики, графічного дизайну та інженерії. Від ефективності, послідовності та емоційної виразності інтерфейсу користувача залежить не лише задоволення від взаємодії, а й кінцева успішність цифрового продукту.

Навчальний посібник «UX/UI дизайн. Проєктування та розробка інтерфейсу користувача» присвячено комплексному вивченню принципів та практик створення інтерфейсів, орієнтованих на користувача. Його структура відповідає логіці проєктного мислення: від розуміння особливостей людино-машинної взаємодії до побудови повноцінного, функціонального й візуально завершеного інтерфейсу з подальшою оцінкою його ефективності.

У першому розділі розглядаються базові аспекти взаємодії користувача з комп’ютерними системами, зокрема канали сприйняття інформації, вплив середовища, психофізіологічні характеристики користувачів та деякі типові патерни поведінки. Цей блок формує теоретичну основу для розуміння суті UX як людського досвіду в цифровому середовищі.

Другий розділ зосереджений на етапах проєктування інтерфейсу, орієнтованого на цільового користувача. Тут висвітлено процес визначення цілей аудиторії, аналіз конкурентів, побудову персон і карт емпатії, а також проєктування сценаріїв, що передують побудові структури інтерфейсу й створенню чорнових прототипів. Особливу увагу приділено дослідницьким підходам і документуванню рішень.

У третьому розділі розглядаються питання створення чистового інтерфейсу. Описано поняття та вимоги до високодеталізованого макета, типи сіток, роль візуальної мови, інтерактивності, а також принципи масштабованості і системності. Окремий підрозділ присвячено мікровзаємодіям – найдрібнішим одиницям взаємодії, які суттєво впливають на емоційне сприйняття інтерфейсу та якість досвіду користувача.

Четвертий розділ висвітлює поняття юзабіліті (usability) як складової якості інтерфейсу, а також пропонує огляд відомих евристик Якоба Нільсена для оцінювання зручності й ефективності цифрових продуктів.

Посібник завершується глосарієм основних термінів, добіркою рекомендованих безкоштовних онлайн-курсів з UX/UI-дизайну та переліком літератури, що може слугувати теоретичною й практичною основою для подальшого навчання.

Матеріал орієнтований на студентів спеціальностей, пов’язаних з інформаційними технологіями, графічним дизайном, а також на викладачів і початківців у сфері UX/UI-дизайну.

Сподіваємось, що цей посібник сприятиме формуванню системного мислення, розвитку дизайнерських навичок і критичного ставлення до проєктних рішень у сфері дизайну інтерфейсів.

РОЗДІЛ 1. ОСОБЛИВОСТІ ВЗАЄМОДІЇ ЛЮДИНИ ТА КОМП'ЮТЕРНОЇ СИСТЕМИ

Взаємодія людини з комп'ютером ґрунтується на фізіологічних та когнітивних можливостях користувача, його досвіді та особливостях сприйняття інформації. Основні способи взаємодії з комп'ютерними системами включають використання клавіатури, миші, монітора, динаміків, сенсорних елементів. Однак, окрім базових засобів, існують також додаткові методи, що розширюють можливості користувача, такі як голосові інтерфейси, технології розпізнавання рухів, біометричні ідентифікаційні системи тощо. Ці засоби можуть застосовуватися для розв'язання спеціалізованих завдань, а також у випадках, коли необхідно підвищити зручність користування інтерфейсом або забезпечити його доступність для людей з особливими потребами.

Взаємодія між людиною та комп'ютером здійснюється через певні канали сприйняття і передачі інформації. У сучасних системах вони все ще залишаються обмеженими, хоча розвиваються нові технології, такі як нейроінтерфейси, що передбачають інтеграцію чипів або сенсорів безпосередньо в нервову систему людини. Однак на даному етапі вони не є широко розповсюдженими і застосовуються здебільшого для вирішення специфічних завдань.

1.1 Основні канали сприйняття інформації користувачем

При проєктуванні інтерфейсів користувача необхідно враховувати, як людина сприймає інформацію та взаємодіє з комп'ютерними системами. Людина отримує інформацію через сенсорні системи, які визначають спосіб взаємодії з цифровими продуктами.

Зір є основним каналом сприйняття в контексті цифрових інтерфейсів. Близько 80% інформації людина отримує саме через зорову систему. При розробці UI-дизайну враховуються такі аспекти:

- контрастність і колірна схема – правильний вибір кольорів впливає на сприйняття і комфортність роботи з інтерфейсом, наприклад, висока контрастність покращує читабельність, а кольори можуть викликати певні емоційні реакції (червоний – попередження, зелений – підтвердження);
- розташування елементів – користувачі читають зліва направо (для мов із такою орієнтацією), що впливає на розміщення важливих елементів;
- форма і розмір об'єктів – великі кнопки привертають більше уваги, ніж дрібні;
- типографіка – вибір шрифту та його розміру впливає на комфортність читання і швидкість обробки текстової інформації;
- анімація та динамічні ефекти – рухомі елементи можуть направляти увагу користувача, але їх надмірне використання може викликати роздратування.

Звук використовується як додатковий засіб взаємодії, особливо для людей із вадами зору або в умовах, коли візуальна інформація недоступна. В контексті проєктування інтерфейсів можна виділити деякі важливі особливості звукового сприйняття:

- чутливість до частоти – людське вухо найкраще сприймає частоти в діапазоні 1–5 кГц, що відповідає звукам мовлення, занадто високі або низькі частоти можуть бути неприємними або малопомітними;
- спрямованість слуху – люди здатні визначати напрямок звуку, що використовується у стереозвуці та просторовому аудіо (наприклад, у VR);
- запам’ятовуваність звуків – впізнавані мелодії, тони або звукові сигнали можуть асоціюватися з певними діями чи брендами (наприклад, звуки повідомлень у месенджерах);

– звукове перевантаження – надлишок звуків може викликати роздратування або відволікати, тому слід обережно використовувати аудіальні ефекти.

Серед типових аудіальних сповіщень можна виділити наступні види:

– інформаційні звуки – наприклад, повідомлення про отриманий лист або завершену операцію; такі повідомлення часто мають «м’який» характер для уникнення ефекту дратування користувача;

– попереджувальні сигнали – гучніші та помітніші звуки, які сигналізують про помилки або необхідність термінової дії (наприклад, низький заряд акумулятора, системна помилка);

– звукове підтвердження – використовується для підкріплення виконаної дії (наприклад, клацання при натисканні кнопки в телефоні або звук ввімкнення пристрою);

– фонові звуки – можуть використовуватися в мультимедійних застосунках для створення атмосфери або орієнтації в середовищі.

Сучасні технології дозволяють користувачам керувати пристроями за допомогою голосу, що особливо корисно для людей із вадами зору або в ситуаціях, коли використання рук є обмеженим:

– голосові команди дозволяють запускати додатки, набирати текст, здійснювати дзвінки без використання фізичних кнопок (Siri, Google Assistant, Alexa);

– технології синтезу мовлення (Text-to-Speech, TTS) дозволяють перетворити текст у звук, що корисно для людей із вадами зору або для аудіокниг;

– технології розпізнавання мовлення (Speech-to-Text, STT) дозволяють конвертувати голос у текст, що використовується в диктуванні повідомлень, голосовому пошуку та автоматичних субтитрах.

У віртуальній та доповненій реальності, а також в ігрових застосунках використовується просторове аудіо для створення реалістичного звукового

середовища: бінауральний звук імітує природний слух людини, створюючи ефект присутності; динамічне позиціонування звуку забезпечує зміну положення звуку залежно від руху користувача.

Дотикове (тактильне) сприйняття є критично важливим в першу чергу для мобільних пристроїв і сенсорних екранів. Цей вид сприйняття базується на роботі різних видів рецепторів, що знаходяться у шкірі та реагують на різні типи впливу. Вони дають змогу розрізняти: тиск (ступінь натискання на поверхню); температуру (різницю між гарячим і холодним); текстуру (шорсткість або гладкість поверхні); вібрацію (коливальні рухи, які допомагають сприймати тактильні сигнали); біль та інші відчуття (зокрема дискомфорт або подразнення від надмірного контакту). Ці характеристики впливають на те, як людина взаємодіє з цифровими інтерфейсами через дотик.

Найпоширенішим способом тактильної взаємодії є сенсорні екрани, які дозволяють керувати пристроєм за допомогою пальців або стилуса. Основні жести включають наступні (але не обмежуються ними!):

- натискання (Tap) – одинарний або подвійний дотик до екрану для вибору об'єкта або активації функції;
- свайп (Swipe) – проведення пальцем (пальцями) у певному напрямку для прокручування або перемикання між елементами.
- щипок (Pinch-to-Zoom) – зведення або розведення пальців для масштабування зображення або тексту;
- довге натискання (Long Press/Hold) – утримування пальця (пальців) на елементі для виклику додаткових опцій (наприклад, контекстного меню);
- перетягування (Drag&Drop) – використовується для зміни розташування об'єктів у просторі.

Зважаючи на різноманітність пристроїв, які використовують сенсорні екрани, кожен виробник зазвичай додає якісь специфічні жести, за допомогою яких можна взаємодіяти з пристроєм, або деталізує вже існуючі

жести, уточнюючи контекст використання, наприклад, виконання якогось жесту одним, двома чи більшою кількістю пальцями.

Haptic-технології використовують вібрацію та зміну текстури поверхні для надання користувачеві фізичних відчуттів у відповідь на його дії:

– вібрація при натисканні кнопок створює ефект фізичної кнопки навіть на сенсорному екрані;

– імітація текстур використовується у сучасних пристроях, таких як iPhone з технологією Taptic Engine, завдяки цьому користувач може «відчувати» шорсткість або м'якість об'єкта;

– реакція на силу натискання дає змогу взаємодіяти з елементами залежно від сили натискання;

– геймерські контролери та VR-рукавички можуть відтворювати відчуття ударів, тертя або дотиків, що посилює ефект занурення у віртуальне середовище.

Одним із найпоширеніших типів фізичних інтерфейсів є клавіатури та миші, які забезпечують високоточний контроль та швидкість введення даних. Вони залишаються незамінними для професійної роботи з текстом, програмування та точних маніпуляцій, де сенсорні екрани поки що не можуть повністю замінити механічні пристрої. При цьому сучасні клавіатури можуть містити додаткові сенсорні елементи, зокрема трекпади або програмовані гарячі клавіші, що покращують користувацький досвід. Фізичні інтерфейси також включають пристрої спеціального призначення, такі як системи розпізнавання жестів, трекери руху та біометричні сенсори. Вони використовуються у медичних технологіях, промислових системах, у розумних будинках, де потрібен безконтактний або мінімально фізичний спосіб взаємодії.

Попри численні переваги дотикової взаємодії, вона має свої виклики та обмеження, які необхідно враховувати при проєктуванні інтерфейсів.

Однією з головних проблем сенсорних екранів є відсутність фізичного зворотного зв’язку. На відміну від механічних кнопок, які дають користувачеві чітке тактильне підтвердження натискання, сенсорні поверхні часто не забезпечують подібного досвіду. Це може ускладнювати використання пристрою, особливо в умовах, коли необхідно швидко реагувати на команди або виконувати дії без візуального контролю.

Ще однією складністю є випадкові дотики, які можуть спричиняти помилкові дії, особливо на смартфонах і планшетах з великими екранами. Людина може ненавмисно зачепити екран під час прокручування або утримання пристрою, що призводить до небажаних результатів. Також тактильні інтерфейси не завжди зручні для користувачів із порушеннями рухової координації або обмеженими можливостями рук, оскільки вимагають точних рухів для взаємодії з елементами інтерфейсу. Окрім цього, тактильний зворотний зв’язок (haptic feedback) також має свої обмеження. Незважаючи на розвиток технологій вібрації та імітації текстур, вони ще не здатні повною мірою передавати складні фізичні відчуття. Додатково, використання тактильного зворотного зв’язку збільшує енергоспоживання пристрою, що може негативно впливати на автономність мобільних гаджетів.

Важливим аспектом є те, що не всі користувачі однаково сприймають тактильні ефекти. Чутливість до вібрацій і механічного опору варіюється залежно від фізіологічних особливостей людини, що ускладнює стандартизацію тактильних відчуттів. Тому розробники часто додають можливість налаштування інтенсивності або повного вимкнення тактильного зворотного зв’язку, щоб зробити пристрої комфортнішими для ширшої аудиторії.

1.2 Вплив середовища на взаємодію користувача з комп'ютерною системою

Середовище, в якому користувач взаємодіє з комп’ютерною системою або програмою, відіграє вирішальну роль у зручності, ефективності та загальному користувацькому досвіді. Фактори навколишнього середовища можуть як сприяти комфортній роботі, так і створювати значні перешкоди, ускладнюючи сприйняття інформації, навігацію та виконання завдань.

Одним із ключових аспектів впливу середовища є **освітлення**. Надмірно яскраве світло або, навпаки, недостатня освітленість можуть знижувати якість зображення на екрані, спричиняючи втому очей. Наприклад, пряме сонячне світло або відбитки на екрані сенсорного пристрою можуть ускладнювати читання тексту та взаємодію з інтерфейсом. В умовах слабкого освітлення недостатній контраст між елементами інтерфейсу може зменшити їхню розпізнаваність. Саме тому дизайнери враховують різні режими роботи, пропонуючи світлі та темні теми, адаптивну яскравість і спеціальні фільтри для зменшення навантаження на зір.

Шумове середовище також може значно впливати на сприйняття інформації, особливо коли взаємодія з системою передбачає аудіальні сигнали або голосове керування. У шумному оточенні користувач може не почути звукових сповіщень, що може призвести до пропущених повідомлень або помилок у виконанні завдань. Голосові команди можуть бути неправильно розпізнані, якщо фоновий шум заважає системі аналізувати мовлення. Вплив шуму є критичним при голосовому управлінні мобільними пристроями та системами керування в автомобілях, де необхідно забезпечити точне розпізнавання голосу або незалежно від рівня шуму. Характеристики аудіальних сповіщень в системах оповіщення про

небезпеку або критичні події на підприємствах повинні забезпечувати необхідний рівень гучності для перекривання наявних шумів.

Ще одним важливим фактором є **фізичний простір**, у якому користувач працює із системою. Наприклад, в обмеженому просторі (транспорт, громадські місця) використання великих сенсорних екранів або миші з клавіатурою може бути незручним. Для таких випадків розробляються альтернативні методи взаємодії, такі як жестове керування, адаптивні інтерфейси та компактні сенсорні панелі. Водночас у просторих офісах або домашніх умовах можуть застосовуватися багатомоніторні системи, великі екрани та периферійні пристрої для підвищення продуктивності.

Кліматичні умови впливають на комфорт користувачів під час роботи з технологіями, але це стосується не тільки показників зручності використання. Температура та вологість можуть змінювати чутливість сенсорних екранів або впливати на роботу апаратних засобів. Наприклад, у холодну погоду сенсорні панелі можуть гірше реагувати на дотики, а користувачам у рукавичках доводиться використовувати спеціальні стилуси або голосові команди. В умовах високої вологості або спеки руки можуть ставати вологими, що впливає на точність дотикових жестів або призводить до небажаних спрацьовувань сенсорних екранів.

Середовище використання може створювати умови, що призводять до тимчасової чи постійної інклюзивності, наприклад:

– користувач читає повідомлення на телефоні під прямим сонячним світлом на пляжі – екран погано видно, потрібна висока контрастність і великі шрифти;

– людина намагається замовити таксі в додатку в переповненому вагоні метро – обмежений простір для взаємодії, багато перешкод для точного натискання;

– користувач використовує навігацію під час пішої прогулянки в дощову погоду – руки зайняті парасолькою, екран у краплях, складно натискати елементи;

– медсестра вводить дані в систему в лікарняному холі з гучною музикою та постійними викликами по гучномовцю – звукове середовище не дає використовувати голосове введення;

– людина намагається скористатися додатком у русі – в автобусі, що їде нерівною дорогою – нестабільне становище робить торкання екрану складним;

– у користувача вдома зникло світло, і він користується додатком на ноутбуці в темряві – яскравий інтерфейс може спричинити втомлюваність очей, важливо мати темний режим;

– на великому стадіоні під час концерту користувач намагається знайти свій ряд через мобільний додаток – дуже шумно, темно, інтернет слабкий, потрібні рішення, що підвищать якість візуального навігатору;

– людина перевіряє розклад транспорту на вулиці в сильний мороз – рукавички не дозволяють точно торкатись екрана, важливо мати підтримку фізичних кнопок або великих елементів;

– автобусні зупинки розташовані під відкритим небом, де часто йде дощ, дме вітер або світить сонце – тому інформаційні табло мають водозахищений корпус, висококонтрастні великі символи і автоматичне підсвічування вночі;

– у ЦНАПах щодня обслуговують людей різного віку, мови й досвіду користування технікою – тому інтерфейс публічних сервісів має підвищену контрастність, адаптацію для людей зі слабким зором;

– у музеях, куди приходять відвідувачі різного віку та фізичних можливостей, умови постійно варіативні – тому мультимедійні екрани

можуть мати аудіогіда, субтитри, тактильні елементи та візуальне дублювання жестової мови.

1.3 Вплив психофізіологічних характеристик користувача на взаємодію з комп'ютерною системою

Психофізіологічні характеристики людини суттєво впливають на ефективність та комфортність її взаємодії з комп’ютерними системами. Вони визначають, як швидко користувач сприймає та обробляє інформацію, наскільки точно виконує завдання, а також як адаптується до змін в інтерфейсі. Крім безпосередньо фізіологічних параметрів, таких, як зір, слух, моторика, додатково можуть враховуватись рівень уваги, швидкість реакції, когнітивні здібності, рівень стресу, емоційний стан та інші особливості та стан організму.

Стандартні тестування інтерфейсів у лабораторних умовах часто не враховують когнітивне навантаження, знижену увагу або емоційний тиск. Це означає, що фактична працездатність інтерфейсу в польових умовах (наприклад, у лікарні, на виробництві, у кризовій ситуації) може бути значно гіршою, ніж очікувалось. Системи, що не адаптуються до різних психофізіологічних станів користувачів, на практиці працюють тільки для вузької групи осіб із «ідеальними» умовами сприйняття. Це звужує аудиторію і потенційно ускладнює доступ до базових послуг для вразливих груп. Інтерфейси мають проєктуватись не лише для комфортної роботи, але й з розрахунком на компенсацію людських обмежень: зниження уваги, втоми, дезорієнтації тощо. Це не «додаткові опції», а ключова умова функціональності системи в реальному середовищі.

Ігнорування психофізіологічних характеристик користувачів у проєктуванні призводить до критичних помилок у використанні інтерфейсу навіть за відсутності очевидних фізичних обмежень. Зокрема, системи

можуть здаватися «інтуїтивно незрозумілими» не через поганий дизайн, а через те, що користувачі не здатні утримати логіку їх роботи в умовах реального навантаження.

Наприклад, **рівень концентрації уваги** безпосередньо впливає на здатність користувача помічати зміни в інтерфейсі або орієнтуватися в складній структурі меню. Людина з нестійкою або зниженою увагою (через втому, перевантаження чи особливості нервової системи) може пропустити важливі повідомлення, не помітити кнопки дії або забути про черговість виконання кроків. У такому разі *інтерфейс має підтримувати користувача за допомогою підказок, нагадувань, підсвічування активних зон або візуального маркування завершених дій*.

Швидкість реакції відіграє важливу роль особливо у тих системах, де взаємодія відбувається в реальному часі. Якщо користувач повільніше реагує на зміну інформації (наприклад, через вікові особливості або після тривалого навантаження), *інтерфейс має дозволяти гнучке налаштування часу очікування, підтвердження дій і уникати критичних сценаріїв, що вимагають миттєвого реагування без попередження*.

Когнітивні здібності, зокрема **обсяг короткочасної пам’яті та здатність до логічного мислення**, впливають на здатність користувача будувати ментальну модель системи. Якщо інтерфейс передбачає виконання багатоетапних процедур, а користувач не здатен утримувати повну послідовність дій у пам’яті, виникає *потреба у спрощенні логіки взаємодії, візуалізації прогресу, а також у можливості повернення до попередніх кроків без втрати даних*.

Емоційний стан, зокрема тривожність, дратівливість або відчуття невпевненості, може істотно знижувати продуктивність взаємодії. Людина у стані стресу має менше когнітивних ресурсів, більше помиляється і гірше розуміє інструкції. У таких випадках інтерфейс має бути максимально доброзичливим, передбачуваним і підтримуючим – наприклад, за

допомогою *спокійної візуальної стилістики, підтвердження дій, можливості скасування, адаптивних підказок та відсутності покарання за помилки*.

В окремих випадках слід враховувати **загальний психофізіологічний стан організму**, зокрема втому, біль або дію медикаментів, які можуть уповільнювати сприйняття або знижувати точність дій. Наприклад, користувач після тривалої нічної зміни або з температурою може неадекватно реагувати на сигнали системи, натискати не ті кнопки або не доводити завдання до кінця. У таких ситуаціях важливо забезпечити *мінімальне когнітивне навантаження, чітку візуальну структуру та підтримку в критичних моментах*.

Як видно з наведених прикладів, психофізіологічна інклюзивність не завжди вимагає складних рішень – у багатьох випадках достатньо передбачити можливість зменшити кількість одночасно доступної інформації, уникати нав’язливих візуальних ефектів або дозволити користувачу налаштовувати темп взаємодії.

1.4 Патерни в дизайні інтерфейсів

Патерн – це певний шаблон, який може виникати в будь-яких сферах життєдіяльності. У системному підході до проєктування інтерфейсів користувача важливу роль відіграє використання дизайнерських патернів – типових рішень для повторюваних проблем або завдань. У практиці UX/UI-дизайну такі патерни умовно поділяють на UX-патерни (патерни користувацького досвіду) та UI-патерни (патерни інтерфейсної реалізації). Обидва типи патернів служать основою для стандартизації взаємодії й забезпечення послідовного досвіду, однак суттєво відрізняються за своєю природою, ціллю та рівнем абстракції (рис.1.1).

Призначення патернів витікає безпосередньо з визначень UX та UI. Відповідно до стандарту ISO 9241-110:2006 інтерфейс користувача (UI, user interface) – це всі компоненти інтерактивної системи (програмне або апаратне забезпечення), які надають користувачу інформацію та елементи керування для виконання певних завдань за допомогою інтерактивної системи. ISO 9241-210 визначає UX (користувальницький досвід) як сприйняття та відповіді людини, що є наслідком використання та/або очікуваного використання продукту, системи або служби.

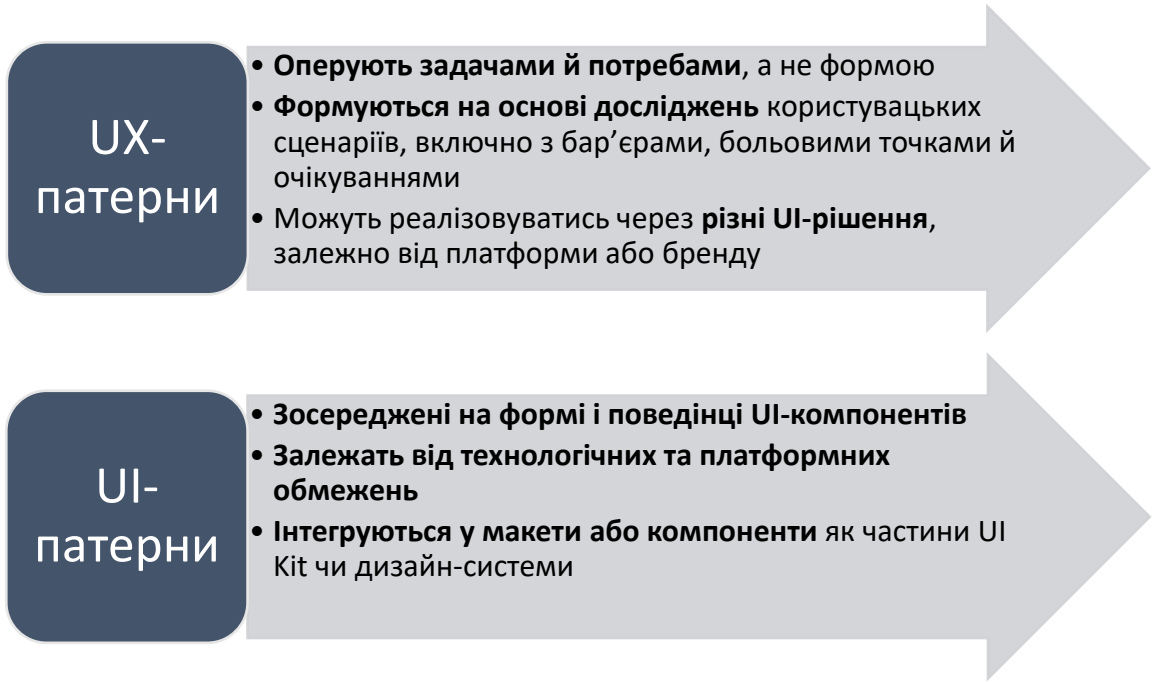


Рисунок 1.1 – Спрямованість патернів UX/UI

UX-патерни – це узагальнені поведінкові моделі, які описують, як користувач вирішує типову задачу або досягає мети у взаємодії з системою, з урахуванням його мотивації, когнітивних можливостей і контексту використання.

На відміну від технічних рішень, UX-патерни оперують сценаріями й структурними шаблонами досвіду, а не візуальними компонентами. Вони не вказують, як саме виглядає інтерфейс, але визначають що має відбутися з точки зору логіки та поведінки.

Існує надзвичайно велика кількість UX-патернів. Нижче наведено деякі поведінкові патерни, які можуть бути використані при проєктуванні взаємодії користувача:

Патерн «безпечне дослідження» передбачає створення середовища, в якому користувач може взаємодіяти з інтерфейсом без страху зробити непоправну помилку. Основна ідея полягає в тому, щоб дозволити експериментування, при цьому забезпечивши можливість легко скасувати або повернути дію.

Патерн «миттєва нагорода» ґрунтується на принципі дофамінової відповіді: позитивна реакція системи на правильну або завершену дію підсилює мотивацію користувача. До типових проявів належать анімації, звукові сигнали або схвалення, які супроводжують досягнення (наприклад, завершення реєстрації або покупки).

Патерн «розумна достатність» передбачає подання користувачеві лише тієї інформації, яка є необхідною на даному етапі виконання завдання. Це знижує когнітивне навантаження та полегшує прийняття рішень.

Патерн «зміни на півдорозі» грунтується на можливості користувача змінити своє рішення/ціль в процесі і передбачає надання зворотного зв’язку або видимих результатів дій до моменту остаточного завершення задачі. Це може бути попередній перегляд, автозбереження, оновлення результатів у реальному часі. Таким чином користувач отримує впевненість і може коригувати хід дій.

Патерн «відкладений вибір» дозволяє користувачеві пропустити або відкласти прийняття складного рішення на пізніший етап. Це підвищує ймовірність подальшої взаємодії, оскільки не створює бар’єрів. Прикладом може бути опція «Нагадати пізніше» або неперервна реєстрація, де частину даних можна ввести згодом.

Патерн «покрокова побудова» передбачає поетапне виконання складного завдання з поділом на логічні кроки. Кожен крок є самодостатнім

і зрозумілим, що підвищує загальну досяжність мети. Поширеним прикладом є багатокрокові форми або майстри налаштування

Патерн «звикання» використовує принцип поступового ознайомлення з функціональністю системи. Спочатку користувач взаємодіє лише з базовими можливостями, а потім – з розширеними. Це особливо актуально для складних професійних застосунків або інструментів із високим порогом входження.

Патерн «просторова пам’ять» передбачає сталість розташування елементів інтерфейсу для того, щоб користувач міг покладатися на просторову орієнтацію. Якщо об’єкти розміщені в одному і тому самому місці, вони швидше розпізнаються, що підвищує ефективність навігації.

Патерн «пам’ять на перспективу» допомагає користувачеві не забути про заплановану дію або задачу. Наприклад, інтерфейс може містити візуальні або текстові нагадування, статуси завдань, маркери «незавершено».

Патерн «організоване повторення» спрямований на підтримку запам’ятовування або навчання шляхом циклічного представлення інформації у зручному форматі. Може бути реалізований через поради, тьюторіали, фіксовані підказки або повторювані повідомлення.

Патерн «тільки клавіатура!» забезпечує можливість повноцінної взаємодії з інтерфейсом без використання миші. Це важливо як для користувачів з інклюзивними особливостями, так і для просунутих користувачів, які прагнуть підвищити швидкість роботи.

Патерн «поради інших людей» передбачає вбудовування соціального доказу – відгуків, рейтингів, рекомендацій інших користувачів. Це зменшує невизначеність і підвищує довіру до системи або продукту. Часто використовується в електронній комерції, сервісах бронювання, освітніх платформах.

Слід зауважити, що UX-патерни є універсальними і завжди формуються навколо конкретної потреби або мети, яку має користувач у межах взаємодії з продуктом, наприклад Наприклад, якщо мета – швидко отримати відповідь, доречний патерн «миттєва нагорода»; якщо мета – послідовне виконання складного завдання, підходить «покрокова побудова».

UI-патерни – це повторювані графічні та функціональні рішення, які описують візуальну та поведінкову реалізацію окремих елементів інтерфейсу. Вони визначають, як виглядає той чи інший компонент, де він розміщується і яку взаємодію дозволяє. UI-патерни є важливою частиною дизайн-систем, оскільки вони стандартизують зовнішній вигляд і взаємодію компонентів (детальніше дизайн-системи розглянуто в розділі 4).

UX-патерни можна розглядати як функціональні сценарії, що ставлять задачу, тоді як UI-патерни – як інструменти її реалізації. Іншими словами, UX-патерн формує очікувану модель поведінки, а UI-патерн пропонує інтерфейсне рішення, яке реалізує цю модель. У сучасних дизайн-системах (наприклад, IBM Carbon, Google Material, Apple Human Interface Guidelines) UX-патерни зазвичай описані в розділах про навігацію, доступність, інтерпретацію станів, тоді як UI-патерни представлені у вигляді бібліотек компонентів, стилістичних токенів і шаблонів макетів.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ІНТЕРФЕЙСУ, ОРІЄНТОВАНОГО НА КОРИСТУВАЧА

Як було зазначено в попередньому розділі, ключовим компонентом будь-якої комп’ютерної системи є користувач. Його потреби, цілі та контекст використання системи відіграють визначальну роль у процесі розробки інтерфейсу. Навіть у випадках, коли користувач діє без певної чітко сформульованої мети, наприклад, переглядаючи інформацію в Інтернеті для розваги, його поведінка все одно підпорядковується певним закономірностям. Відповідно, система має бути розроблена таким чином, щоб ефективно задовольняти його запити.

В цьому розділі розглядаються основні функції UX/UI-дизайнера в сучасній розробці IT-продуктів та ключові положення підходу User-Centered Design до проєктування та розробки інтерфейсів

2.1 Роль UX/UI-дизайнера в сучасній розробці IT-продуктів

Проєктування UX/UI-дизайну є важливою складовою розробки IT-продуктів. Будь-який інтерфейс створюється не для самого розробника, а для кінцевого користувача, який має власні потреби, очікування та особливості взаємодії з програмним забезпеченням. Навіть у найпростіших випадках, таких як калькулятор, слід враховувати, хто саме буде ним користуватися – школяр, бухгалтер, інженер або навіть дитина, що вивчає цифри. Для кожної групи користувачів можуть бути актуальними різні аспекти реалізації інтерфейсу.

Сучасна галузь IT рідко розмежовує посади UX- і UI-дизайнерів, оскільки фахівці, які володіють обома компетенціями, мають значно вищу затребуваність. UI-дизайнер відповідає за візуальну складову інтерфейсу та його функціональні елементи, тоді як UX-дизайнер аналізує потреби

користувача та визначає, наскільки комфортним буде досвід взаємодії з продуктом. В ідеальному випадку користувач не помічає дизайн, адже він працює інтуїтивно і не викликає труднощів чи роздратування.

Поняття «інтуїтивності» та «дружності» інтерфейсу мають конкретні критерії, які можна виміряти та оцінити. У контексті професійної діяльності UX/UI-дизайнера такі характеристики не повинні залишатися лише загальними гаслами, а мають ґрунтуватися на аналізі, тестуванні та порівнянні з іншими продуктами. UX/UI-дизайнер займається дослідженням та аналізом користувачів, поєднуючи знання з психології, фізіології, біології та інших дисциплін, що стосуються людини. Цей фахівець повинен уміти комунікувати, аналізувати інформацію, систематизувати та формалізувати отримані дані.

Основними завданнями UX/UI-дизайнера є збір та дослідження інформації про користувачів, їхні цілі, потреби, очікування, больові точки, а також середовище та обмеження використання продукту. Однак цей процес не завжди виконується дизайнером самостійно. Часто за збір та формалізацію вимог відповідає бізнес-аналітик або аналітик вимог, який систематизує дані та створює специфікацію. Важливу роль у процесі також відіграє Product Owner, який представляє інтереси замовника та забезпечує комунікацію між командами. Дизайнер може працювати з уже зібраною інформацією, яку надають аналітики або Product Owner. У великих проєктах команда UX/UI-дизайнерів не завжди має можливість безпосередньо спілкуватися зі стейкхолдерами, тому інформація передається через Product Owner, що інколи призводить до втрати або спотворення даних.

UX/UI-дизайнер досліджує не лише кінцевих користувачів, а й ширшу аудиторію, яка може взаємодіяти з аналогічними продуктами. Крім того, він може використовувати результати сторонніх досліджень, якщо вони містять релевантну інформацію. Після збору даних дизайнер

проводить їхню систематизацію, відкидаючи зайве, узагальнюючи важливе та визначаючи прогалини, які потребують додаткового аналізу.

Отримана інформація оформлюється у вигляді схем, діаграм, списків або інших структурованих моделей, що дозволяють чітко визначити, які аспекти мають значення для розробки інтерфейсу. Це включає аналіз того, які елементи впливатимуть на взаємодію користувачів із продуктом та які дизайнерські рішення будуть найбільш ефективними.

Початківці у сфері UX/UI-дизайну часто стикаються з труднощами у визначенні цінної інформації про користувачів та її впливу на розробку інтерфейсу. Процес аналізу даних вимагає навичок комунікації, критичного мислення, систематизації та відбору найбільш значущих факторів. Відпрацювання цих умінь є важливим завданням під час навчання та практичної роботи, оскільки правильна обробка інформації безпосередньо впливає на якість кінцевого продукту.

Після збору, аналізу та систематизації інформації UX/UI-дизайнер переходить до безпосереднього розроблення дизайну інтерфейсу. Цей процес базується на користувацьких та бізнес-вимогах, а також на технічних обмеженнях платформи. Важливо враховувати, що інтереси замовника та кінцевих користувачів не завжди збігаються, тому дизайнеру доводиться шукати компроміс або робити вибір на користь однієї зі сторін.

Одним із ключових аспектів проєктування є визначення показників якості інтерфейсу та методів їх досягнення. Якість дизайну оцінюється не лише з точки зору зручності використання (usability), а й через відповідність стандартам, патернам проєктування та загальним принципам взаємодії користувачів із цифровими продуктами.

На цьому етапі UX/UI-дизайнер проєктує користувацькі сценарії – моделі поведінки користувачів у системі. Вони можуть бути представлені у вигляді текстових покрокових описів, графічних діаграм або їх комбінацій.

Використання діаграм є особливо важливим у складних сценаріях, де можливі розгалуження та альтернативні шляхи виконання дій.

Далі визначається візуальне представлення взаємодії з користувачем: які елементи інтерфейсу будуть використані, як вони виглядатимуть і як забезпечуватимуть інтуїтивне користування. Досвідчені дизайнери зазвичай розробляють кілька альтернативних варіантів інтерфейсу, щоб надати замовнику можливість вибору оптимального рішення.

Наступним етапом є визначення візуального вигляду та способів взаємодії користувача з інтерфейсом. Дизайнери зазвичай розробляють кілька альтернативних варіантів, які оцінюються замовником. Один і той самий функціональний елемент може бути реалізований різними способами. Наприклад, для введення дати можливі такі варіанти:

- календар, що розгортається після натискання відповідної кнопки;
- три спадаючі списки для вибору дня, місяця та року;
- одне текстове поле з маскою введення у форматі «дд/мм/рррр»;
- голосове введення, що може бути зручним для користувачів з обмеженими можливостями;
- спеціальні пристрої, наприклад, із підтримкою шрифту Брайля.

Кожен із варіантів має свої переваги й обмеження, а вибір конкретного рішення залежить від потреб користувачів, контексту використання та технічних можливостей. UX/UI-дизайнер аналізує ці аспекти та приймає рішення, яке забезпечить найбільш ефективну взаємодію з інтерфейсом.

2.2 Визначення цілей користувача

При проєктуванні інтерфейсів, орієнтованих на користувача, визначення цілі допомагає зрозуміти, що саме користувач хоче досягти, використовуючи затосунок. Це дозволяє створювати функціонал,

орієнтований на вирішення реальних завдань, та зменшити ймовірність фрустрації користувачів (стан розчарування, почуття розпачу чи безвиході в процесі досягнення цілі чи при оцінці досягнених результатів).

Методика SMART забезпечує структуру та ясність у визначенні цілей. Ціль за SMART повинна бути:

- S - Specific - конкретною;
- M - Measurable – такою, що може бути виміряна;
- A - Achievable - досяжною;
- R - Relevant - значимою;
- T-Time bound - обмеженою в часі.

Показник Specific визначає наступне:

- *Що конкретно необхідно досягти? В чому полягає конкретність досягнень/результату?* В контексті UX-UI дизайну визначається з точки зору користувачів системи, і ні в якому разі не з точки зору розробників чи спонсорів/замовників проєкту!
- *Хто приймає участь в процесі?* В контексті UX-UI дизайну – це цільова аудиторія майбутньої системи.
- *Де саме/в якому середовищі відбувається досягнення цілі?* В контексті UX-UI дизайну слід уточнити, що це буде за застосунок – мобільний застосунок, веб-сайт, десктопний застосунок, вбудована система тощо.

Показник Measurable визначає наступне:

- *Як визначається, що ціль досягнута? Як можна виміряти досяжність цілі чи певний результат?* Необхідно вказати конкретні критерії або показники, які дозволять виміряти прогрес.
- *Що є результатами досягнення цілі?* Необхідно вказати, на основі чого можна визначити, чи ціль досягнута, чи ні.

Показник Achievable визначає наступне:

– *Чи реалістична ця ціль?* Необхідно вказати, чи можна реально досягти ціль з наявними ресурсами та в зазначений термін.
– *Які кроки потрібно зробити для досягнення цілі? Чи є технології, які дозволяють досягнути ціль?* Треба описати конкретні дії, які необхідно виконати для досягнення цілі. В контексті UX-UI дизайну можуть бути також вказані конкретні технології, що можуть бути використані в застосунку для досягнення цілі.

Показник Relevant визначає наступне:

– *Чому ця ціль важлива для користувача?* В чому полягає цінність цілі для користувача? Наскільки ціль є значимою/цінною для користувача?
– *Чи варто користувачеві взагалі витрачати час і ресурси на досягнення цієї цілі?*

Показник Time bound визначає наступне:

– *Який кінцевий термін для досягнення цілі?* Необхідно вказати конкретний термін або часові рамки, за який користувачеві бажано досягнути ціль з використанням додатку.
– *Коли починається і коли завершується виконання завдань, що дозволяють досягнути цілі?* Необхідно визначити початковий та кінцевий термін для виконання завдання (дата, або день, або місяць, або інший момент часу).

Приклад формулювання SMART-цілі.

Майбутній застосунок: веб-застосунок «Майстерня В'язання» для навчання в'язання на спицях.

Цільова аудиторія: користувачі-початківці – люди, які не мають жодного досвіду або дуже невеликий досвід в'язання на спицях.

Формулювання SMART-цілі: «Користувач-початківець навчиться базовим технікам в'язання на спицях за допомогою мобільного застосунку

«Майстерня В’язання» протягом наступних 4 тижнів, виділяючи по 30 хвилин щодня на навчання та практику з застосуванням відеоуроків та інструкцій в застосунку, щоб до кінця цього періоду зв'язати простий шарф, використовуючи три основні техніки в’язання (лицьова петля, виворітна петля, накид), що сприятиме хобі користувача-початківця та допоможе розвинути нову навичку, яка є важливою для його інтересів у створенні власних в’язаних виробів».

Розшифровка SMART-цілі:

Конкретна (Specific): Навчитися базовим технікам в’язання на спицях за допомогою мобільного застосунку «Майстерня В’язання».

Вимірювана (Measurable): Досягти рівня, коли користувач-початківець зможе повністю зв’язати простий шарф з використанням трьох основних технік в'язання (лицьова петля, виворітна петля, накид).

Досяжна (Achievable): Ціль може бути досягнена, якщо користувач-початківець буде виділяти протягом наступних 4 тижнів по 30 хвилин щодня на навчання за допомогою відеоуроків та інструкцій у застосунку, виконуючи хоча б один урок на день.

Релевантна (Relevant): Ціль сприятиме хобі користувача-початківця та допоможе розвинути нову навичку, яка є важливою для його інтересів у створенні власних в’язаних виробів.

Обмежена в часі (Time-bound): Досягнення (чи недосягнення) цілі визначається через 4 тижні за умови виконання користувачем-початківцем щоденного уроку протягом мінімум 30 хвилин.

2.3 Аналіз прямих та непрямих аналогів

Щоб виявити кращі практики, уникнути недоліків застосунків-конкурентів і краще зрозуміти очікування користувачів, використовується **аналіз програм, що є прямими та непрямими аналогами**.

Прямі аналоги: пропонують однакову або дуже схожу цінну пропозицію майбутнім користувачам застосунку. Фактично, застосунки прямі аналоги містять всі необхідні чи більшість ключових функцій, які потрібні для досягнення визначеної цілі користувачів. Їх призначення повністю або майже повністю збігається з призначенням майбутнього застосунку.

Непрямі налоги: пропонують подібну цінну пропозицію різному сегменту користувачів; або вони націлені на нашу потенційну базу користувачів, не пропонуючи точно таку ж цінну пропозицію. В цілому, вони можуть потенційно задовільнити ті самі потреби, що й застосунок, який розробляється, але такі функції не є прямим призначенням цих застосунків.

Наприклад, якщо розробляється застосунок для навчання в’язання на спицях, то в якості непрямого аналогу можна розглянути навчальну платформу Prometeus, яка містить навчальні курси, і потенційно на ній можна розмістити навчальні матеріали (в тому числі, відеоуроки), тестові завдання, форум для зв’язку з тренером (ментором) курсу чи іншими учнями курсу для опанування необхідних навичок. При цьому сама платформа не має прямого цільового призначення – «навчити в’язати на спицях»! Зверніть увагу, що непрямі аналоги, як правило, можуть задовільнити відносно невелику частину потрібних функцій, а іноді навіть взагалі одну чи дві, але таки функції є важливими чи мають таку реалізацію, що може бути використана в майбутньому продукті. Зазвичай, непрямі аналоги можуть містити велику кількість функцій, що є зайвими і надлишковими для досягнення цілей визначених користувачів.

Результати аналізу прямих та непрямих аналогів дозволяють визначити наступне:

– які UX-UI рішення добре працюють у схожих застосунках, щоб використовувати їх як основу для свого проєкту;

- які недоліки є в аналогах, яких можна уникнути у власному застосунку, забезпечуючи кращий користувацький досвід;
- непрямі аналоги можуть підказати інноваційні підходи до вирішення схожих завдань в іншому контексті;
- перелік UX-UI рішень, до яких звикли користувачі (рішення, які зустрічаються в різних аналогах та реалізовані однаково або майже однаково);
- які унікальні функції можна додати в майбутній застосунок, що виділятимуть його серед інших (в аналогах таких функцій немає, або вони реалізовані таким чином, що не забезпечують належне досягнення цілей наших користувачів).

В цілому, дослідження аналогів сприяє економії часу та ресурсів завдяки використанню перевірених підходів, водночас залишаючи простір для індивідуальних покращень.

2.4 Аналіз користувачів. Розробка профілів персон

2.4.1 Загальний алгоритм розробки персони

Персони або персонажі користувачів є одним з інструментів дизайну продукту або послуги, орієнтованого на користувача, який ґрунтується на ідеї, що створювати продукти потрібно «навколо» людей і їх цілей, а не навчати людей тому, як використовувати продукти і не робити «дизайн для всіх». Ключове завдання в даному випадку – це зрозуміти, що потрібно користувачеві через його поведінку, ставлення, потреби і цілі, проявляючи емпатію і дизайн-мислення.

Персона (персонаж) – це узагальнений, але **реалістичний** опис типового або цільового користувача продукту, тобто архетип, а не реальна

жива людина. Але персони повинні описуватися так, якби вони були справжніми людьми.

ВАЖЛИВО!

Опис персони не повинен документувати кожен аспект життя уявної людини, а повинен фокусуватися на тих характеристиках, які впливають на програму, що розробляється.

Загальний алгоритм створення персони можна описати наступними кроками:

1. **Визначення списку характеристик користувачів**, які є важливими для майбутнього застосунку. При аналізі характеристик варто чітко розуміти, на що значення тієї чи іншої характеристики може вплинути в майбутньому дизайні.

ВАЖЛИВО!

Треба уникати додавання сторонніх деталей, які не мають будь-яких наслідків для дизайну системи.

2. **Збір інформації про потенційних користувачів**. Для збору інформації можуть використовуватись інтерв'ю, опитування, фокус-групи, анкетування тощо, які проводяться з реальними людьми. Однак, за відсутності можливості спілкування з реальними потенційними користувачами, створюють прото-персони. ***Прото-персона*** представляє собою образ користувача, ще не підтверджений практичними дослідженнями, і ґрунтується тільки на припущеннях команди про цілі, задачі та характеристики потенційних користувачів.

3. **Групування користувачів в кластери за схожими ознаками користувачів.** Якщо кілька кластерів здаються занадто схожими, необхідно об’єднати їх разом або усунути будь-які характеристики, які здаються менш важливими для цілей системи.

4. **Узагальнення характеристик користувачів кожного кластеру** та формування опису «типового» для кластеру користувача (персони).

5. **Деталізація опису персони.** Деталі додаються, щоб зробити персону більш реалістичною, правдоподібною і незабутньою. Варто долучити емпатію, щоб отримати «живий» опис персони. В якості відповідного інструменту використовується карта емпатії.

ВАЖЛИВО!

Кінцева мета полягає в створенні правдоподібної і живої персони.

Для кращого сприйняття персоні треба придумати ім’я, яке буде максимально відповідати образу користувача (порівняйте ваше перше враження від імен користувачів «Марічка», «Ізольда Едуардівна» та «Марфа Іванівна»).

Також важливо підібрати (або згенерувати засобами ШІ, наприклад https://create.microsoft.com/uk-ua/features/ai-image-generator*) відповідне зображення, яке дозволить ще краще уявити персону користувача (порівняйте зображення на рис.2.1).*

КАТЕГОРИЧНО НЕ РЕКОМЕНДУЄТЬСЯ:

– *використовувати імена та фото відомих осіб (актори, бізнесмени, політики тощо);*

– *використовувати імена та зображення персонажів кіно, серіалів, мультфільмів, героїв книг, коміксів тощо.*

Рисунок 2.1 – Приклади зображень користувачів, генеровані ШІ (генеровано з використанням https://designer.microsoft.com/image-creator)

Такий підхід може ввести команду в оману та додати майбутній персоні характеристик, якими вона насправді не володіє. Також не рекомендується придумувати прото-персонам імена, які є «фантастичними», придуманими «заради сміху», «щоб було всім цікавіше та веселіше». З одного боку такі імена дозволяють команді легко запам’ятати персону. З іншого боку, як і імена відомих людей, персонажів творів, фільмів, вони можуть внести певні непорозуміння в трактування образу персони. Наприклад, навряд чи ***типового*** представника цільової аудиторії сайту ЦНАП можуть звати Ельдрех Круанзель (ім’я та прізвище генеровано Chat GPT). Звісно, що може так трапитись, що знайдеться реальна людина з таким іменем, але мова про те, що це повинно бути таке ім’я, що дозволяє легко уособити цю персону з визначеною групою майбутніх користувачів застосунку, а не просто «цікаво звучить».

2.4.2 Типові питання для дослідження цільової аудиторії та виявлення характеристик користувачів

Типові питання для дослідження цільової аудиторії та виявлення характеристик користувачів включають наступний перелік:

– **Для досягнення яких цілей та вирішення яких проблем майбутні користувачі зможуть використовувати застосунок? Чи взагалі він їм потрібен?** Для формування відповідей на ці питання можна використовувати раніше розглянутий формат постановки цілей по SMART, але іноді можна обійтись спрощеним варіантом. Наприклад, розглянемо застосунок для онлайн-запису до перукаря: мета користувача – зручно забронювати час без дзвінків і зайвих уточнень; проблема – часто не може додзвонитись, змушений чекати відповідь у месенджерах; чи потрібен застосунок – так, оскільки він економить час і дозволяє самостійно обрати вільні слоти; без застосунку користувач або дзвонить, або пише в Instagram, часто отримує відповідь із затримкою.

– **Наскільки часто буде виникати потреба у використанні майбутньої системи? За яких обставин (або з яких причин) майбутні користувачі будуть мати цю потребу, яка мотивація цього використання?** Щоб правильно спроєктувати майбутній застосунок, важливо зрозуміти як часто і чому користувач взагалі звертатиметься до нього. Це дозволяє визначити, чи має система бути постійно доступною (як щоденний інструмент), чи достатньо періодичної взаємодії. Також важливо з’ясувати, яка подія або обставина спонукає людину відкрити застосунок, і що її мотивує: зручність, обов’язок, вигода, тиск, звичка тощо. Наприклад, в системі електронної черги до лікаря частота використання застосунку періодична (раз на кілька тижнів або місяців), обставини використання виникають, коли користувач відчуває проблему зі здоров’ям або планує плановий огляд, його мотивація включає уникнення черг, економію часу, бажання швидко знайти вільного лікаря, тож система має бути доступною «за вимогою» та не потребувати частого оновлення. Другий приклад – CRM-система для менеджера з продажів, яка відноситься до так званих застосунків «робочого використання». Таким чином, частота її використання – багаторазово протягом дня. Обставини, які вимагають

використання системи, - це робочі дзвінки, листи або угоди, які потребують внесення даних, перевірки статусу клієнта, запису нотаток тощо. Мотивацією користувача є безпосередньо робочий процес, контроль з боку керівництва, бажання не втратити клієнта або угоду.

– **Яким чином зараз користувачі вирішують свої задачі (без використання майбутнього застосунку)? Чи використовують вони якісь інші застосунки? Якщо так, то як вони їх оцінюють, що вважають важливим у функціоналі та дизайні цих застосунків?** *Зверніть увагу, що раніше проведений аналіз аналогів може стати в нагоді для виявлення цих характеристик.*

– **Якими є демографічні, психофізіологічні, соціальні та інші характеристики користувачів (вік, фізичний/психологічний стан, рівень освіти, професійна діяльність, соціальна активність, матеріальний стан, культурні особливості, мова спілкування та інші)? Чи всі з них впливають на взаємодію з майбутнім застосунком? Яким чином ці характеристики можуть вплинути? Наскільки різняться групи користувачів за цими характеристиками? Чи є підгрупи користувачів з особливими потребами?**

– **Які ключові слова/словосполучення/фрази описують, що користувач робить в «реальному житті»?** Варто визначити не тільки фрази в контексті предметної галузі майбутнього застосунку, а й визначити стилістику мови користувача, його «термінологію», що дозволить команді краще «відчути» цільову аудиторію. Наприклад, кур’єр служби доставки не скаже: «Я завершив доставку товару». Він швидше використовує такі фрази, як «відвіз посилку», «здав точку», «відмітив доставку», «все, закрив маршрут», або «не було на місці – поставив статус». Його мова проста, ділова, часто скорочена до ключових слів, із мінімумом формальностей. Це дозволяє зрозуміти, що у застосунку йому зручніше бачити короткі

підказки типу «точка здана», «відмітити відсутність», а не «Ваше завдання виконано успішно».

– **Який рівень досвіду мають користувачі при використанні подібних застосунків?** При аналізі цього питання за наявності фокус-груп з реальними користувачами можна використовувати чек-листи, які дозволять спростити процеси збору необхідної інформації. Приклад такого чек-листа наведено в таблиці 2.1.

Таблиця 2.1 – Приклад опитувальника для визначення досвіду користувачів в користуванні застосунками-аналогами

№	Питання	Відповідь / Позначка
1	Чи користувався користувач раніше подібними застосунками чи сервісами? Якими саме?	
2	Як часто користується подібними системами у щоденній діяльності?	☐ Щодня ☐ Раз на тиждень ☐ За потреби
3	Чи має досвід з аналогічними/конкурентними рішеннями? Що подобалося / не подобалося?	
4	Чи виникали труднощі з попередніми системами? Які саме?	
5	Наскільки комфортно орієнтується в інтерфейсах цифрових продуктів (меню, фільтри, повідомлення)?	☐ Дуже впевнено ☐ Помірно ☐ Невпевнено
6	Чи потребує навчання або допомоги при освоєнні нового застосунку?	☐ Так ☐ Частково ☐ Ні
7	Який рівень використання функцій системи?	☐ Лише базові ☐ Частково розширені ☐ Повноцінно працює з усіма функціями

– **Які поведінкові патерни притаманні користувачеві?** *(питання патернів розглянуто в розділі 1).*

– **Чи є якісь підкатегорії користувачів, які мають особливі запити/потреби?** Мова тут не тільки про користувачів, що потребують інклюзивної взаємодії (наприклад, люди, що мають дальтонізм), але й про нетипові способи вирішення завдань чи сценарії поведінки. Наприклад, застосунок – це мобільна платформа для бронювання столиків у ресторанах і основна група користувачів, це звичайні гості, які планують вечерю. Нетиповою підкатегорією користувачів можна вважати фуд-блогерів або гастрооглядачів. Їх «особливі запити»: вибір столика біля вікна або з привабливим інтер'єром для фото/зйомки; доступ до інформації про спеціальні дегустаційні меню або «нетипові» позиції; можливість залишити розширений відгук із фото та оцінками за різними критеріями; важливість швидкого обслуговування для зйомок (наприклад, щоб їжа не втратила вигляд); інтеграція з соцмережами або розширений профіль (як «професійний гість»). Вони не є основною цільовою аудиторією, але їхня поведінка та потреби можуть сильно відрізнятися - вони можуть використовувати систему не за стандартним сценарієм, наприклад, бронювати одразу кілька столиків на день в різні години не для безпосереднього споживання їжі, а для зйомок.

– **В яких умовах передбачається використовувати систему (які характеристики оточуючого середовища)? Чи важлива фізична сторона робочого середовища (освітлення, шум, робочий простір, температура, наявність комп'ютерів, телефонів, кількість персоналу і т.п.); місце роботи користувача і ступінь його мобільності (офіс/квартира/магазин/…, стаціонарно/з пересуваннями і т.д.); питання ергономіки та умов використання системи (задіяний зір/слух, робота ведеться сидячи/стоячи, чи визначені режими роботи, тривалість змін і т.д.)?** Наприклад, у випадку з касовим застосунком для

супермаркету, користувач (касир) працює у середовищі з постійним шумом, обмеженим робочим простором, великою кількістю клієнтів, стоячи, протягом тривалих змін. Тут важлива ергономіка, простота навігації, великі кнопки, мінімальна кількість кліків і наявність апаратних засобів (сканери, термінали тощо). Інший приклад – застосунок для логістичного персоналу, який працює на складі. Користувач може пересуватись між стелажами з планшетом або смартфоном у руках, часто одягнений у рукавички, працює при слабкому освітленні чи в холодних умовах. У такому випадку потрібен контрастний інтерфейс, великі елементи управління, мінімальна залежність від введення тексту та можливість голосових команд.

– **Які програмно-апаратні засоби доступні для використання користувачеві (види та характеристики пристроїв, платформи, характеристики технічних каналів зв'язку, потреба в допоміжному програмному забезпечення, необхідному для виконання задач користувача в системі та ін.)?** Наприклад, офісні працівники зазвичай використовують стаціонарні або портативні комп’ютери з настільними браузерами, сучасними операційними системами (Windows, macOS) і стабільним провідним або Wi-Fi з’єднанням. У такому середовищі система може мати розширений функціонал, великі екрани та складну навігацію. Натомість кур’єри чи працівники складу здебільшого взаємодіють із системою через смартфони або промислові планшети на Android, у середовищах з нестабільним мобільним інтернетом (3G/4G). Тут критичною є оптимізація для мобільного інтерфейсу, офлайн-доступ, синхронізація даних та мінімальне навантаження на канал зв’язку. Часто також потрібне допоміжне програмне/апаратне забезпечення, наприклад, сканер штрихкодів або геолокаційні модулі. Ще один приклад – користувачі освітніх платформ, які можуть працювати з різноманітних пристроїв: ноутбуки, планшети, телефони з різними ОС, часто – з обмеженим обсягом пам’яті та старими браузерами. У такому випадку важливо забезпечити

кросплатформену підтримку, адаптивний дизайн, сумісність із браузерами попередніх версій і, за потреби, зовнішні плагіни або програми, як-от Zoom, Google Meet чи інтерактивні дошки.

Список питань може бути уточнений виходячи з особливостей конкретного проєкту.

2.4.2 Розробка карти емпатії

Візуалізувати потреби цільової аудиторії проєкту допомагає **карта емпатії** користувачів (User Empathy Map).

Карта емпатії є ключовим інструментом для дизайну інтерфейсів, які відповідають реальним очікуванням і досвіду користувачів. Цей підхід є розробкою компанії XPLANE і призначений для дослідження цільової аудиторії. Карта емпатії може бути складена як для реально існуючого застосунку, так і для того, який тільки проектується.

Емпатія - це психологічний термін, який означає здатність не просто розуміти емоції, почуття, потреби та досвід інших людей, а «ставити себе на їхнє місце». У контексті дизайну інтерфейсів емпатія означає здатність дослідника або дизайнера глибоко зрозуміти цільових користувачів, їхні проблеми, мотивації та очікування.

Для представлення результатів дослідження користувачів рисують графічну схему, в центрі якої розташовується фото, скетч чи інше зображення користувача, якого обрано як персону з цільової аудиторії, а навкруги в певних блоках розміщуються фрази, цитати, описи, що дозволяють створити уявлення про цю персону.

Структура карти емпатії може мати різні форми. В класичному варіанті навкруги зображення користувача в карті емпатії розміщуються 4 блоки, що описують взаємодію користувача з оточуючим світом в контексті задачі, яку передбачається вирішити за допомогою майбутнього

застосунку: думаю і відчуваю (Think and Feel), бачу (See), чую (Hear), кажу і роблю (Say and Do), а також 2 узагальнюючих блоки: больові точки (Pain), успіхи і прагнення (Gain) (рис.2.2). Блоки больових точок та успіхів і прагнень формуються як в контексті майбутньої системи так і загалом, і заповнюються після того, як описані основні 4 розділи, що відображають взаємодію та відчуття.

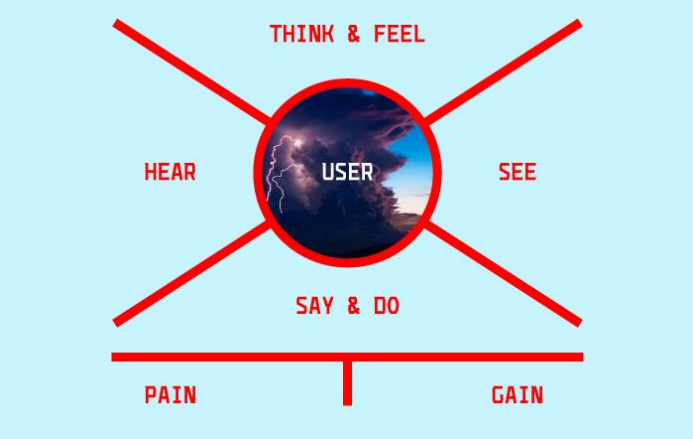


Рисунок 2.2 - Приклад структури карти емпатії

Вміст блоків зазвичай включає нижченаведені аспекти.

1. **Думаю і відчуваю (Think and Feel):** глибинні переконання; нерозказані емоції (розчарування, тиск, страх, відраза, надія); сумніви.

Приклад:

«Я боюся, що якщо натисну не те – все зітреться.»

«Ненавиджу, коли щось працює не з першого разу.»

В застосунку варто зробити підтвердження дій, автозбереження, дружні повідомлення про помилки в стилістиці та термінології мови користувача.

2. **Бачу (See):** які інтерфейси користувач бачить щодня (конкуренти, банківські застосунки, месенджери тощо); робоче/домашнє оточення

(монітор/планшет/старий смартфон/гучне кафе тощо), хто/що впливає: реклама, соцмережі, колеги, родина, друзі тощо.

Приклад:

«У мене дешевий Android і я не бачу дрібний шрифт.»

В застосунку варто використати великий шрифт за замовчуванням, адаптивний дизайн, достатній контраст.

3. **Чую (Hear):** що кажуть колеги/друзі/родичі, звідки поступає аудіальна інформація: відгуки, TikTok, техпідтримка, інфлюенсери тощо, що з почутого є інформаційним шумом і інші фактори, пов'язані зі звуковим каналом.

Приклад:

«Колега казала, що це все одно не працює без інструкцій.»

В застосунок варто додати мікронавчання в інтерфейсі, інфо-підказки.

4. **Кажу і роблю (Say and Do):** що говорить користувач під час використання, які реальні дії робить (скролить без зупину, не читає інструкції, замикається на одній кнопці і т.д.), типова поведінка користувача (пробує натискати навмання, звертається в чат, закриває вкладку і т.д.).

Приклад:

«Ну де тут кнопка «Далі»?» (хоч вона є, але непримітна)

В застосунку варто зробити таку кнопку великою, кольоровою, з підписом «Далі» або «Продовжити».

5. **Больові точки (Pain)** (як в контексті майбутньої системи так і загалом, заповнюються після того, як заповнені розділи 1-4)**:** з якими труднощами стикається користувач зараз, без застосунку; які процеси він вважає незручними, повільними, ризикованими; що його найбільше дратує в аналогах або ручних процедурах. Інформація з цього блоку

використовується для оптимізації найболючіших для користувача операцій та усуненні дрібних дратівливих факторів.

Приклад:

«Зараз мені треба дзвонити менеджеру або писати в месенджер, щоб дізнатися статус заявки. Це займає багато часу і мене ігнорують.»

Інтерфейс має містити автоматичне оновлення статусу, доступне користувачу в один клік.

«Я часто помиляюсь у формах, бо не розумію, що саме від мене хочуть.»

У проєктуванні передбачаємо чіткі підказки, приклади введення, валідацію в процесі введення даних чи безпосередньо після введення.

6. **Успіхи і прагнення (Gain)** (як в контексті майбутньої системи так і загалом, заповнюються після того, як заповнені розділи 1-4)**:** який ідеальний досвід користувач хоче мати, що буде вважати вдалим результатом, які очікує вигоди (наприклад, економія часу, легкість, надійність, статус, автоматизація тощо).

Приклад:

«Було б добре, якби я міг усе зробити за 2 хвилини, поки стою в черзі.»

«Мені просто треба знати, що все зареєстровано і я не забув нічого важливого.»

«Я хочу бачити, що відбувається з моєю заявкою, але без зайвих дзвінків і уточнень.»

ВАЖЛИВО!

Для підсилення емпатії інформація, що вноситься в карту емпатії, повинна бути живою та емоційною. Доречними будуть цитати з життя персони.

Приклад карти емпатії для користувача-початківця Лесі, що хоче навчитись в’язати спицями наведено на рис.2.3. Леся є студенткою, вона захоплюється всім, що пов'язано з рукоділлям, але має досвід лише в'язання гачком. Хотіла б опанувати в'язання на спицях, в тому числі і тому, що це хобі дозволить внести індивідуальні оригінальні нотки в її гардероб. Фото для даної карти емпатії згенеровано з використанням сервісу https://designer.microsoft.com/image-creator. Зверніть увагу, що карта емпатії містить велику кількість цитат (текст в лапках), що визначають думки персони, її промови, аудіальні сигнали від знайомих, родичів та з медіа. Розділи больових точок, успіхів та прагнень сформовані у вигляді списку ключових моментів, які є важливими для персони в контексті подальшої розробки інтерфейсу майбутнього застосунку.

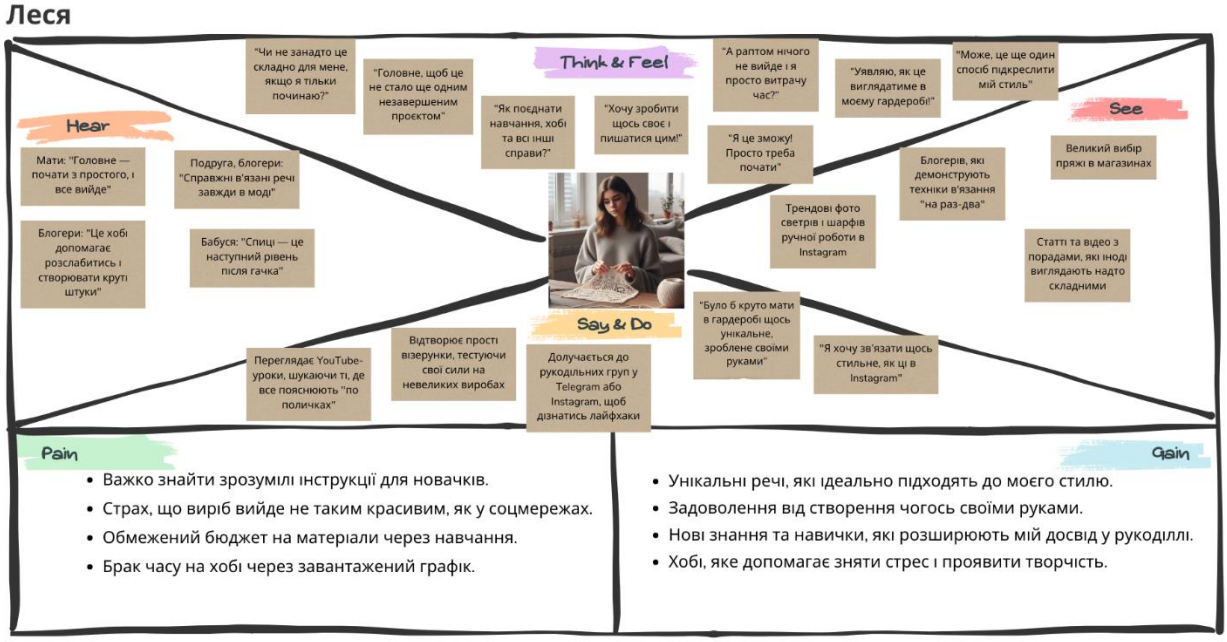


Рисунок 2.3 – Приклад карти емпатії користувача-початківця Лесі («студентка»)

При створенні карти емпатії варто використовувати готові шаблони, які є в великій кількості в мережі Інтернет. Однак, там дуже часто зустрічаються інші, ***модифіковані варіанти карти емпатії***, які можуть

мати іншу структуру (наприклад, як на рис.2.4) і в більшості орієнтовані на виявлення характеристик користувача, як потенційного покупця (результати маркетингових досліджень).

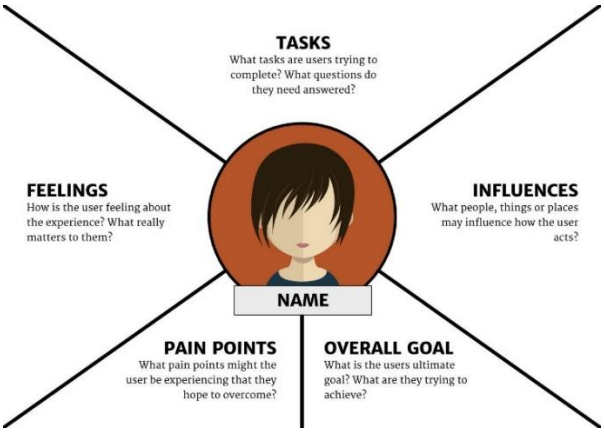


Рисунок 2.4 – Приклад структури модифікованої карти емпатії

Вміст блоків модифікованої карти емпатії може бути використано для отримання відповідей на наступні питання:

- Завдання. Які завдання користувачі намагаються вирішити? На які питання вони повинні відповісти? Слід зауважити, що завдання - це не «функції системи», а реальні завдання з життя користувача, які він намагається вирішити – незалежно від того, чи є для цього застосунок, чи ні. Для формулювання завдань можна використовувати формати, схожі з користувацькими історіями (User Stories), які є одним із інструментів роботи з користувацькими вимогами в швидких методологіях розробки.

Приклад шаблону для формулювання завдань:

[Користувач] хоче/прагне/має потребу у [конкретна дія або результат], щоб [мотивація, емоційна або практична вигода].

Приклади формулювання завдань для учителя початкової школи:

- Учитель хоче швидко повідомити батьків про домашнє завдання, щоб уникнути зайвих питань і зберегти спокій після уроків.
- Учителю потрібно виставити оцінки одразу після уроку, щоб не повертатися до цього ввечері або наступного дня.
- Учитель прагне мати підтвердження, що інформація дійшла до батьків, щоб уникнути скарг на «недостатню комунікацію» і не виправдовуватись.
- Учителю потрібно мати можливість швидко відмітити відсутність учня, щоб не втрачати час на паперові журнали та не допускати плутанини.

- **Почуття. Як людина ставиться до нового досвіду? На які питання чекає відповіді?** Блок «Почуття» в цьому контексті замінює або доповнює класичний блок «Think and Feel», але фокусується саме на ставленні до нового, невідомого, змінного. Це опис емоційної реакції користувача на новий досвід, інтерфейс, систему або зміну звичного процесу. Тут важливо дослідити наступні питання:

- Які емоції виникають у людини під час першого контакту з системою?
- Як вона ставиться до змін і цифрових інструментів?
- Які внутрішні питання чи тривоги виникають у неї, хоча вона може їх і не озвучувати?

Типові питання, які «живуть у голові» користувача, але не завжди озвучуються:

- «А це точно безпечно?»
- «А я зможу розібратись без допомоги?»
- «А чи воно взагалі мені потрібне?»
- «Скільки це займе часу?»
- «А якщо я зроблю щось не так – що станеться?»
- «Хто побачить мої дії?»

- «А раптом дані зникнуть?»

Приклад опису блоку «Почуття» для офісного менеджера, який має користуватись новим застосунком для внутрішньої звітності:

- Відчуває роздратування: «Знову щось нове, коли вже звик до Excel.»
- Сумнівається, чи нова система дійсно спростить життя.
- Чекає відповіді на питання: «А це не займе більше часу, ніж стара система?», «Чи будуть збережені старі звіти?».

- **Вплив. Як люди, речі або місця, можуть вплинути на поведінку користувача?** Блок «Вплив» - один із найцінніших, коли ми хочемо зрозуміти не лише людину, але й середовище, яке формує її вибір, звички, бар'єри чи мотивації. І саме тут UX-дизайнер має побачити непрямі чинники, які часто неочевидні, але дуже дієві. Цей блок описує зовнішні фактори, які можуть впливати на поведінку, рішення чи мотивацію користувача. Сюди належать:

  – люди: колеги, керівники, друзі, родичі, експерти тощо;
  – речі/пристрої: смартфони, комп'ютери, сканери, блокноти, паперові журнали тощо;
  – місця/середовище: транспорт, шумне кафе, лікарня, виробництво, віддалена робота тощо.

Приклад опису блоку «Вплив» для учителя початкових класів:

Люди: колеги можуть сказати: «Я працюю тільки з папером – так простіше і звичніше». Загальне небажання колективу переходити на нові технології може гальмувати прийняття застосунку в окремих вчителів («ефект натовпу»).

Речі: звичка до паперового журналу, щоденника, дошки, фізична прив'язаність до формату.

Місце/середовище: школа з поганим Wi-Fi, старі комп'ютери, потреба використовувати власні гаджети для роботи з застосунком. Через

це використання онлайн-застосунку може викликати фрустрацію або упередження.

- Больові точки. З яким болем справляється користувач? Чи хоче її подолати? Блок «Больові точки» у модифікованій карті емпатії є аналогом класичного блоку Pain, але з глибшим фокусом на внутрішнє сприйняття та готовність діяти. Це не просто «що незручно», а який дискомфорт користувач реально відчуває, наскільки він усвідомлений, і чи є в нього мотивація позбутися цього болю. Фокус: не лише «в чому біль», а й «наскільки вона болить». Два користувачі можуть мати однакову проблему, але: один відчуває її як справжній бар'єр і хоче рішення; інший – сприймає як норму, не має мотивації змінювати ситуацію.

Приклад опису блоку «Больові точки» для учителя в державній школі:

- Біль: постійна потреба вручну дублювати інформацію в різні канали (журнал, чат для батьків, Viber).
- Сприйняття: «Так було завжди, вже звик.»
- Готовність змінювати спосіб вирішення задач: низька, якщо новий застосунок не економить час буквально «з перших хвилин».

- Цілі. Яка кінцева мета користувача? Чого він намагається домогтися? Блок «Цілі» у модифікованій карті емпатії – це ключ до розуміння кінцевого сенсу взаємодії користувача з вашим рішенням. На відміну від блоку «Завдання», який описує, що саме користувач робить, блок «Цілі» відповідає на питання – «Навіщо він це робить? Який результат для нього справді важливий?». Ціль – це те, заради чого взагалі виникає взаємодія з системою. Часто користувачі цього прямо не формулюють, але дизайнер має інтерпретувати її «між рядками». Типові види цілей:

– практична мета – завершити справу, заощадити час, уникнути помилок;

– емоційна мета – відчувати спокій, впевненість, контроль;

– соціальна мета – виглядати компетентно, уникнути критики, бути «на рівні».

Приклад блоку «Цілі» для учителя початкових класів:

- Кінцева мета: підтримувати порядок у документах і зменшити хаос у щоденних комунікаціях з батьками.
- Важливо: мати впевненість, що все враховано й зафіксовано.
- Соціальний аспект: не виглядати непрофесійно перед батьками або адміністрацією.

Приклад блоку «Цілі» для кур'єра служби доставки:

- Кінцева мета: швидко завершити маршрут без проблем і скарг.
- Емоційна мета: не нервувати через дрібниці, не пояснювати клієнтам або менеджеру, «чому щось не так».
- Практична мета: здати звіт без додаткових дзвінків.

ВАЖЛИВО!

При виборі шаблону карти емпатії варто використовувати такий, що надає максимально повну інформацію про персону в контексті саме UX-UI дизайну, оскільки «маркетингові» шаблони дуже часто зосереджені зовсім на іншому – користувач і його цілі/задачі розглядаються як відправна точка для отримання прибутку компанії, а при виборі характеристик користувача в першу чергу беруться до уваги такі, що дозволять побудувати успішну стратегію продажів, і значно меншої мірою – задовольнити потреби користувача. Такий підхід не відповідає людиноорієнтованим принципам розробки інтерфейсів.

2.4.3 Картка персони (персона)

Узагальнену інформацію про персону зводять у вигляді документу, що містить опис характеристик користувача у структурованому вигляді.

Така «картка» персони може бути створена як в звичайному текстовому документі чи таблиці, або з використанням шаблонів в спеціальних редакторах, наприклад https://www.canva.com/templates/s/user-persona/, https://www.figma.com/community/tag/user%20persona/files, https://miro.com/templates/ux/ та інші.

ВАЖЛИВО!

Наявні в мережі шаблони дуже часто мають маркетингову спрямованість і містять набори характеристик, які не знадобляться для побудови інтерфейсу (рис.2.5). В цілому на проєктах картка персони може включати не тільки ту інформацію, що потрібна для UX-UI дизайну, але в варто в першу чергу зосередитись саме на тих характеристиках, що можуть бути використані при проєктуванні та розробці інтерфейсу.

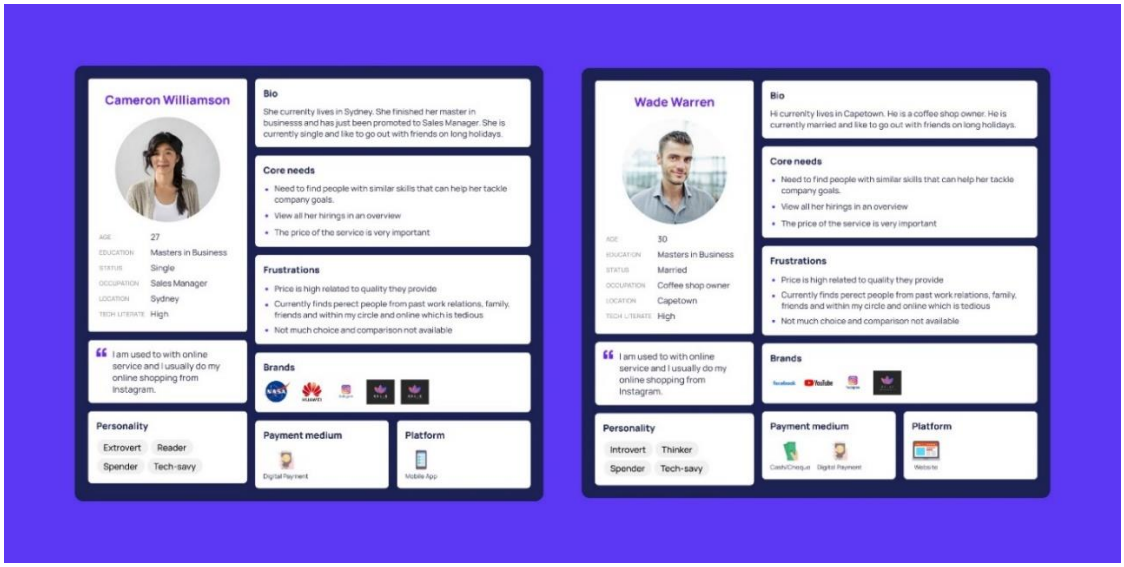


Рисунок 2.5 – Приклади персон, створених за програмними шаблонами. Шаблони мають орієнтацію на маркетинг, а не на UX-UI дизайн

Створення картки персони не повинно займати багато часу. Для пришвидшення процесу розробки картки персони можна обійтись простим табличним шаблоном, наприклад, як на рис. 2.6. Тут представлено картку

персони для користувача-початківця Лесі (студентка), зроблену безпосередньо в документі Word.

<table>
<tr><td>  </td><td>Леся
Користувач-початківець (студентка)</td><td colspan="4">Я студентка. Захоплююсь всім, що пов'язано з рукоділлям, але у в’язанні маю досвід лише з гачком. Мені подобаються в’язані спицями речі, але в Інтернеті стільки всього, що мені важко вибрати. До того ж, не завжди є зрозумілі інструкції для новачків ще й безкоштовно. Оскільки я ще вчусь, то на хобі не завжди залишається час. Я дуже хочу навчитись в’язати унікальні речі, які б ідеально підходили до мого стилю. Було б класно сумістити гачок та спиці.</td></tr>
<tr><td colspan="2">Вік: 20 років</td><td>Стать: жінка</td><td colspan="2">Зайнятість: студентка</td><td>Мова спілкування: українська</td></tr>
<tr><td colspan="3">Рівень навичок в’язання на спицях: низький</td><td colspan="3">Суміжні навички: в’язання гачком</td></tr>
<tr><td colspan="2">ЦІЛІ
Швидко навчиться базовим технікам в’язання на спицях, щоб мати можливість зв’язати прості та складні вироби.
•</td><td colspan="4">ОЧІКУВАННЯ
• Хочеться мати унікальні в’язані речі, які ідеально підходять до моєго стилю.
• Я отримую задоволення від створення чогось своїми руками, сподіваюсь, що в’язання на спицях не стане виключенням.
• Нові знання та навички розширюють мій досвід у рукоділлі.
В’язання - це хобі, яке допоможе мені зняти стрес і проявити творчість.</td></tr>
<tr><td colspan="2">Готовність виділяти час на навчання: середня</td><td colspan="2">Зацікавленість в кінцевому результаті: висока</td><td colspan="2">Фінансові можливості: низькі</td></tr>
<tr><td>Платформи: macOS, iOS</td><td colspan="5">Соціальні мережі, месенджери, медіа: Дуже часто: Доволі часто: Дуже рідко:</td></tr>
</table>

Рисунок 2.6 – Приклад шаблону картки персони у форматі таблиці Word

На рис.2.7 наведено приклад картки персони, створеної з використанням шаблону https://www.canva.com/templates/EAFupsPXUJ4-beige-and-orange-professional-marketing-buyer-persona-a4-document/, який було модифіковано відповідно до завдань розробки інтерфейсу.

Рисунок 2.7 – Приклад картки персони, створеної на основі шаблону Canva

ВАЖЛИВО!

Не існує єдиного «правильного» шаблону для створення картки персони! При виборі шаблону завжди треба керуватись тим, які саме характеристики персони ми хочемо відобразити в картці і як розмістити їх в картці так, щоб це було наочно, зрозуміло і цікаво не тільки тому, хто розробив цю картку, а й іншим учасникам команди розробки, оскільки така картка в реальних проєктах використовується всією командою, а не тільки безпосередньо UX-UI дизайнером.

На рис.2.8 та 2.9 наведено приклади карток персон, створених для інших застосунків та з використанням інших шаблонів.

Рисунок 2.8 – Приклад картки персони з використанням табличного формату. Персона – «Новачок-кінолог», застосунок для планування та відстеження прогресу собак під час тренувань

Рисунок 2.9 – Приклад картки персони з використанням шаблону. Персона – «Досвідчений веган», застосунок для веганів

При створенні картки персони варто дотримуватись певних правил:

– картка повинна вміщуватись на одну сторінку А4 (орієнтація сторінки не має значення) чи екранну сторінку без необхідності масштабування зображення для того, щоб можна було прочитати наявну інформацію;

– в картку включається та інформація, яка знадобиться в першу чергу для подальшого проєктування інтерфейсу користувача, інша інформація подається для забезпечення реалістичності персони (наприклад, ім’я робить персону більш особистісною та легкою для обговорення в команді і дозволяє команді говорити про персону як про реальну людину);

– не можна включати в картку персони контактну інформацію, якщо розробляється прото-персона (пам’ятаємо, що у випадку прото-персони маємо справу з придуманим персонажем), і, навпаки, якщо дослідження проводиться на реальних представниках цільової аудиторії (наприклад, при розробці застосунку для задач підприємства ми описуємо реального типового користувача-співробітника), то варто додати контактні дані для зв’язку з цією людиною;

– дуже бажано, щоб картки персон мали однаковий список та формат представлення характеристик – це дозволяє легко побачити схожість та відмінності різних персон та груп користувачів (порівняйте картки різних персон однієї групи «Користувач-початківець», підгрупи «Студентка», «Пенсіонерка» на рис.2.7 та 2.10).

Приклади деяких характеристик, які можна включити в картку персони з точки зору розробки UX-UI, наведено далі:

Вік – може бути використано, якщо є групи користувачів, що мають певні граничні вікові значення (люди похилого віку, маленькі діти): для людей похилого віку як правило можуть спостерігатись проблеми з зором (типовим порушенням є далекозорість), тому можна запропонувати в UI можливість збільшення розміру шрифтів без значних змін в макетах

екранних форм; для маленьких дітей характерним є невміння читати, тому в застосунках може бути використано технології голосового інтерфейсу (VUI), реалізовано інтерактивні анімовані підказки та інші способи взаємодії, що не вимагають навичок читання. Для інших ситуацій характеристика віку може використовуватись для додавання більшої реалістичності персони.

Рисунок 2.10 – Приклад картки персони підгрупи користувачів «Пенсіонерка» групи «Користувачі-початківці» застосунку для навчання в’язання на спицях

Мова спілкування – характеристика визначає параметри локалізації

майбутнього застосунку (мова, часовий пояс, формат дати, часу тощо). Слід зауважити, що навіть якщо користувачі розмовляють декількома мовами, це зовсім не означає потребу в багатомовному інтерфейсі. Вибір локалізації повинен бути обґрунтований потребами користувачів.

Пристрої, платформи – визначають, яких стандартів в дизайні варто буде дотримуватись при розробці UX-UI. Якщо типові користувачі використовують для вирішення своїх завдань різні пристрої/платформи, то при створенні макетів екранних форм треба реалізувати потрібну кількість версій макетів з урахуванням стандартів кожної групи пристроїв та платформи.

Соціальні мережі, месенджери, медіа – важливо визначити, не тільки якими саме середовищами користуються персони, але й наскільки часто, чи комфортно їм це використання. Ця інформація може бути використана в дизайні для різних задач. Наприклад, можна передбачити можливість входу в додаток через акаунт Facebook з правами зареєстрованого користувача, можна додати в GUI можливість поділитись інформацією на сторінці Facebook, переслати інформацію у Viber чи Telegram тощо. Конкретний спосіб врахування інформації про соціальні мережі, месенджери та медіа залежить від цілей та задач користувача.

Мотивація – характеристика може відображати як рівень мотивації майбутнього користувача при виконанні ним певних завдань, для яких він планує використовувати застосунок, так і текстовий опис причин, що мотивують його виконувати ці завдання і використовувати при цьому деякий застосунок. Ця інформація в подальшому враховується при розробці сценаріїв взаємодії з застосунком та при визначенні поведінкових патернів, притаманних даній групі користувачів.

Очікування – блок визначає, що користувач сподівається отримати незалежно від існування застосунку та для чого саме планується використати в результаті взаємодії з застосунком. Дані щодо очікувань

користувача можуть бути сформовані на основі карти емпатії (блок Gain, «Цілі»).

ВАЖЛИВО!

При описі очікувань треба уникати абстрактних формулювань типу «простота використання», «швидкий доступ до функцій», оскільки вони не відповідають характеристикам SMART і в подальшому оцінка їх досягнення сильно ускладняється.

Проблеми – визначають основні бар'єри, труднощі та «болі», які заважають користувачу досягати своїх цілей. Список проблем в картці персони може бути сформований на основі даних карти емпатії (блок Pain, «Больові точки»).

Рівень доходів (фінансові можливості) – характеристику варто вказувати, як правило, у тому випадку, ***коли в застосунку передбачено використання монетизації***. Можна розглянути варіант безкоштовного використання застосунку за умови монетизації за рахунок зовнішньої вбудованої реклами (можливі реалізації в UI – спливаючий рекламний банер, що перекриває основний екран; рекламні повідомлення в статусному рядку; рекламні банери фіксованих розмірів в зоні основного контенту) або використовувати персоналізацію інтерфейсу на основі платних підписок.

ВАЖЛИВО!

Не існує єдиного «правильного» фіксованого списку характеристик, які включаються в картку персони! При виборі характеристик завжди треба керуватись тим, як вплине та чи інша характеристика на UX-UI, або наскільки вона дозволяє «оживити» персону, зробити її реалістичною та запам'ятованою для команди.

2.5 Розробка сценаріїв користувача

Сценарій користувача – це правдоподібна детальна історія про те, як користувач вирішує певну задачу за допомогою програми, що розробляється. Сценарій містить послідовність кроків, від початку виконання деякої задачі до досягнення визначеної користувачем деякої фінальної точки, що визначає успішне чи неуспішне вирішення задачі.

Кожна програма створюється для вирішення певних завдань, які допомагають користувачам досягти їх цілей. Таким чином, в ***рамках одного програмного продукту реалізується багато сценаріїв користувача***, виконання яких приведе користувача до певної цілі або які стосуються цієї цілі опосередковано (рис.2.11).

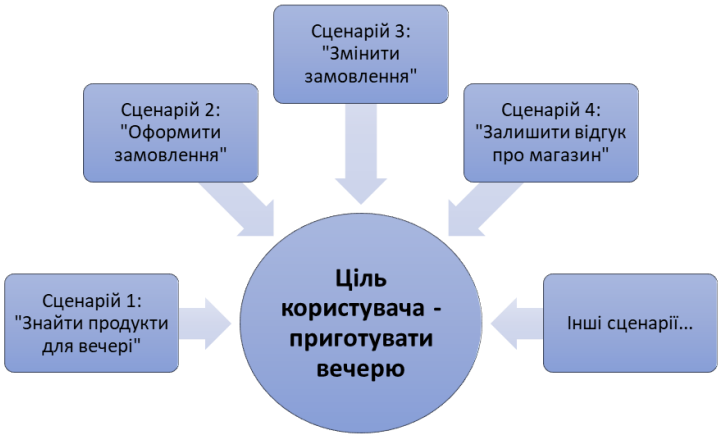


Рисунок 2.11 – Зв’язок сценаріїв з метою користувача

Важливим є те, що різні групи користувачів, а також різні персони в межах кожної групи, можуть мати свої специфічні сценарії взаємодії для досягнення власних мікроцілей. Наприклад, сценарій «Оформити замовлення» для одної персони може включати дії, які пов’язані з оформленням кур’єрської доставки продуктів, оскільки людина замовляє їжу у великій кількості (персона святкує день народження,

організовує корпоратив, має нагодувати велику сім'ю тощо), а іншій персоні зручніше заїхати за замовленням самостійно і не прив'язуватись до графіку служби доставки, оскільки вона з якихось причин не має можливості спланувати час доставки. Таким чином, навіть одна і та сама задача може мати різні сценарії для різних персон.

Щоб написати хороший сценарій варто виділити в ньому наступні компоненти:

Контекст: Опишіть ситуацію, яка може викликати необхідність використовувати додаток, що розробляється.

Цілі: Чого хоче досягти користувач? (незалежно від того, чи буде він використовувати додаток, чи ні)

Проблеми: Що користувач намагається вирішити та з якими перешкодами стикається? (описуємо не проблеми майбутнього додатку, а проблеми людини при виконанні задач, що приводять до цілі!).

Мотиви: Для чого користувачу потрібно те, що буде пропонувати додаток?

Особисті якості користувача (використовуємо інформацію з картки персони, з карти емпатії): Забезпечують передісторію для більш детальної картини.

Звички, захоплення та/або переконання користувача (беремо з картки персони, з карти емпатії): Додають додатковий контекст.

Персональні та демографічні дані (беремо з картки персони): Відповідна інформація про ім'я, місце проживання, дохід, професію, сімейний стан і т.д. (якщо вона наявна). Додають додатковий контекст та дозволяють «персоналізувати» сценарій.

Приклад користувацького сценарію «Оренда автомобіля для ділової поїздки»:

Ірина, топ-менеджер великої торгівельної компанії, їде у ділову поїздку в Закарпаття. В рамках поїздки вона буде мати вільний час, який

хотіла в витратити на походи по горам. Щоб мати певну свободу пересування, компанія виділяє Ірині гроші на оренду автомобіля. Зважаючи на те, що у вільний час Ірина хоче поїхати в гори, а в окремих місцях дороги можуть бути поганими, вона хоче орендувати автомобіль, придатний для таких подорожей. ***В пошуках авто Ірина звертається до нашого сайту.*** *Вона не знається на автомобілях такого типу, тому їй треба порівняти декілька моделей, почитати відгуки від інших клієнтів. В інших поїздках вона вже орендувала автомобілі, тому трохи розбирається в типових умовах оренди.* ***Коли вона знайде на нашому сайті підходящий автомобіль, вона його орендує в рамках виділеного бюджету.***

Ключові компоненти сценарію:

Контекст: Оренда автомобілю для ділової поїздки.

Цілі: Ірина хоче отримати надійний автомобіль для їзди бездоріжжям і в межах бюджету її компанії.

Проблеми: Ірина не знайома с такими типами автомобілів.

Мотиви: Похід у гори під час відрядження та безпечне прибуття на маршрути.

Особисті якості користувача: Цілеспрямована, любить активний відпочинок, завжди готова до нових відкриттів, але при цьому стежить за безпекою («мама повинна повернутись до дітей після активного відпочинку неушкодженою»).

Звички, захоплення та/або переконання користувача: Пішохідні прогулянки, альпінізм.

Персональні та демографічні дані: Ірина, 35 років, заміжня, має 2 дітей, топ-менеджер великої торгівельної компанії, має високий рівень доходів.

Приклад користувацького сценарію «Замовлення святкового кейтерингу для ювілею»:

Олена готується до святкування свого 60-річного ювілею. Подія

особлива: вперше за багато років збирається уся велика родина – діти, онуки, хрещеники, племінники, подруги. Урочистість заплановано провести в будинку за містом, з гарно накритим столом, музикою, фотоальбомом і теплими тостами.

Олена прагне створити домашню атмосферу, але водночас все має бути «на рівні» - вона хоче, щоб кожен гість почувався бажаним, а стіл виглядав щедро, різноманітно й смачно. Оскільки готувати на 30+ людей самотужки складно й виснажливо, вона вирішує замовити частину страв у кейтеринговому сервісі, а дещо приготувати разом із донькою.

Їй важливо знайти щось близьке до домашньої кухні, з класичною подачею: запечене м’ясо, салати «як у ресторані», соління, канапки, тістечка. Водночас – без «екзотики» та «фьюжну», бо серед гостей є старші люди, діти, кілька алергіків.

Олена переглядає сайт нашої платформи з кейтеринговими рішеннями для подій. Їй зручно працювати з планшетом, але вона не любить, коли треба довго шукати або розбиратись. Вона хоче побачити готові пропозиції у форматі «меню на 10, 15 або 30 осіб», з фотографіями, цінами, загальним переліком страв. Їй важливо, щоб їжа:

- мала знайомі назви,
- подавалась у святковому оформленні,
- була зрозуміла за складом (щоб уникнути неприємних несподіванок – наприклад, гострих соусів або грибів, які не всі їдять).

Також для неї важлива точність у часі: гості збиратимуться о 15:00 – тому вона очікує доставку заздалегідь, з можливістю попереднього нагрівання чи подачі. Вона турбується, що в останній момент щось піде не так, тому звертає увагу на відгуки інших замовників, шукає фото «з життя», уточнює телефон підтримки. Коли вона визначиться з меню та умовами доставки, то оформить замовлення.

Ключові компоненти сценарію:

Контекст: Замовлення святкового кейтерингу для ювілею в заміському будинку на 30+ гостей.

Цілі: Забезпечити щедрий, красивий і зручний стіл без виснаження власними зусиллями.

Проблеми: Страх «щось не те обрати», сумнів у нових форматах їжі, обмежений час, чутливість до деталей оформлення та подачі.

Мотиви: Хоче бути «хорошою господинею», але не втомитись до свята, подарувати близьким радісний спогад.

Особисті якості: Турботлива, відповідальна, емоційна, обережна в ухваленні рішень, схильна довіряти практичному досвіду, а не «модним» трендам.

Звички, захоплення, переконання: Любить «класику» в кулінарії, не довіряє всьому, що «занадто нове»; вірить, що справжнє свято – це тепло, простота і увага до кожного гостя.

Персональні та демографічні дані: Олена, 59 років, пенсіонерка, мешкає в передмісті Києва, має двох дорослих дітей і трьох онуків, звикла користуватись планшетом та Viber, середній рівень доходу, цінує сервіс, якому можна зателефонувати й уточнити все «по-людськи».

2.6 Графічне представлення сценаріїв

Для деталізації послідовності кроків, що дозволяють виконати задачу, використовують User Flow.

User Flow (UX flow, user flow charts, interaction flows, activity flows, user interface flows, navigation flows, user flow diagrams, task flow diagrams) – це графічна схема (блок-схема), яка відображає шлях, який проходить користувач (порядок дій) для досягнення певної мети при використанні програмного продукту.

Для побудови User Flow тадиційно в використовують графічні елементи, які схожі з елементами блок-схеми (рис.2.12). Приклад User Flow, оформленого в такому стилі, наведено на рис. 2.13.

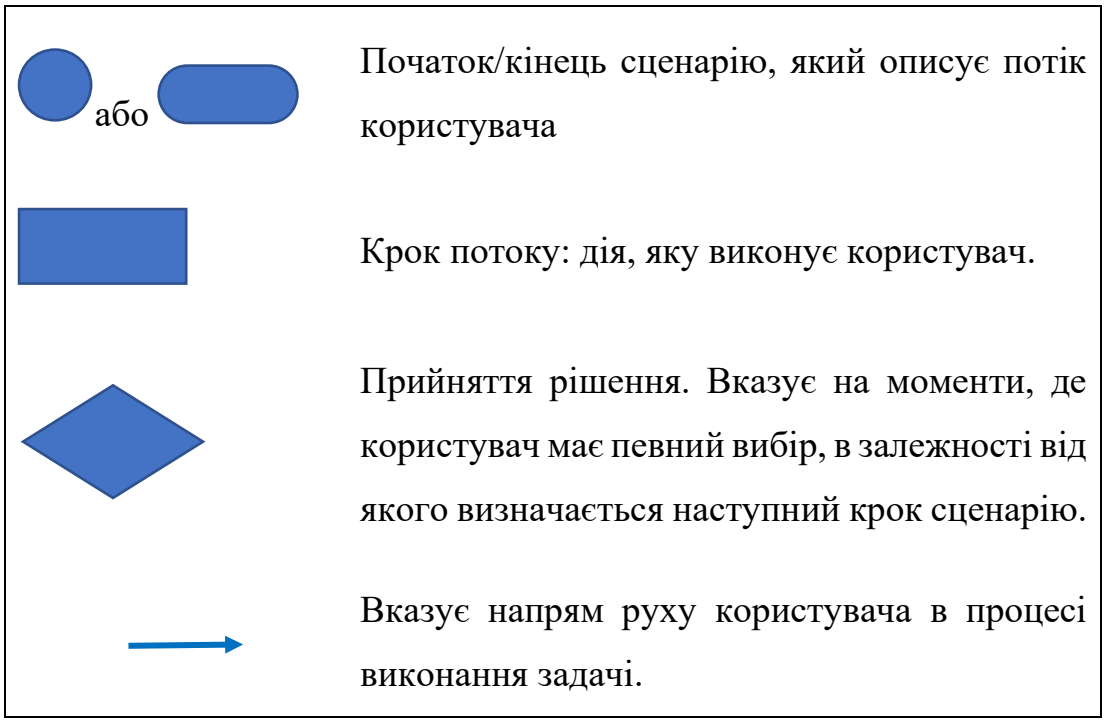


Рисунок 2.12 – Графічні елементи User Flow

Щодо використання кольорів чи стилів оформлення, то немає певного стандарту – варто керуватись здоровим глуздом та естетичністю готової схеми.

Одним із часткових випадків User Flow є Task Flow. **Task Flow Diagram** – це **лінійна схема**, що представляє собою графічне зображення процесу виконання **конкретного завдання** користувачем. Воно показує послідовність дій, необхідних для досягнення певної мети, без деталізації варіантів чи розгалужень. Task Flow завжди орієнтований на конкретну задачу, наприклад, «оренда автомобіля-позашляховика», «замовлення їжі на ювілей». User Flow – це більш загальна діаграма, яка описує, як користувач може взаємодіяти з усією системою для досягнення своєї цілі.

User Flow при цьому може включати кілька завдань або містити декілька сценаріїв поведінки різних користувачів для вирішення одного завдання. Task Flow є складовою частиною User Flow. При побудові Task Flow можна обмежитись лише блоками Початок/кінець та блоком «Крок потоку» (блок дії), оскільки розгалужень не передбачено. Але, якщо в якомусь блоці користувач приймає певне рішення, то цей крок потоку може бути представлено блоком «Прийняття рішення» без розгалуження. Вважається, що користувач прийняв в цьому блоці одне конкретне рішення і альтернатива не розглядається. При використанні таких блоків рекомендується для наочності показати, за якої умови стався перехід до наступного кроку.

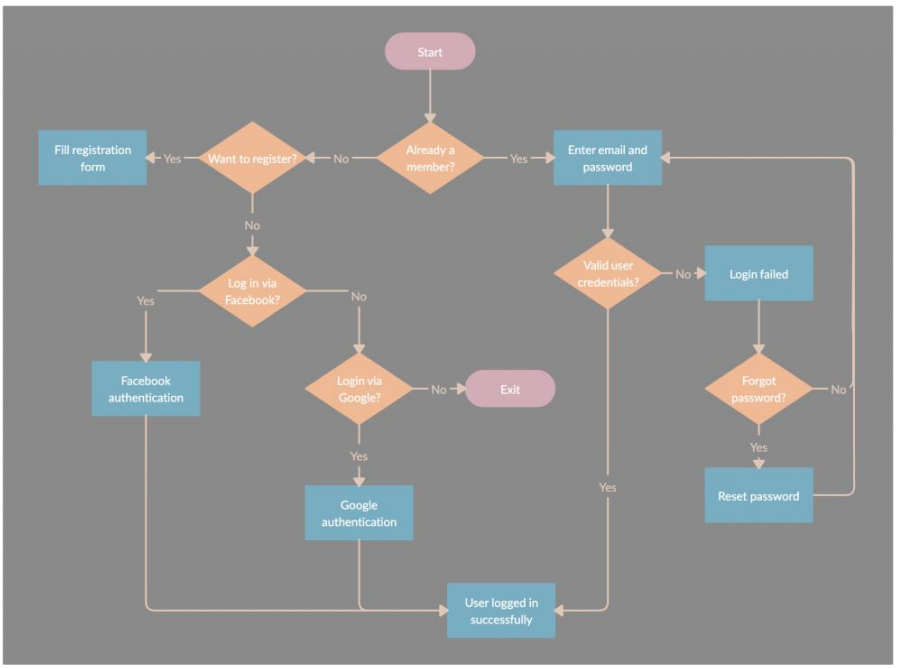


Рисунок 2.13 – Приклад оформлення User Flow

Розглянемо варіант оформлення Task Flow на рис. 2.14 для сценарію «Замовлення святкового кейтерингу для ювілею», обговореного в попередньому пункті. В цьому варіанті навмисно зроблено декілька помилок, які на перший погляд можуть здатись незначними, але в

подальшому можуть викликати певні проблеми при розробці User Flow та макетуванні інтерфейсу.

Рисунок 2.14 - Приклад Task Flow для сценарію «Замовлення святкового кейтерингу для ювілею». Варіант з помилками

Блоки на рис.2.14, вказані блакитним кольором (1, 2, 5, 10), містять операції, які є чіткими та представляють атомарну дію. Таким блокам легко поставити в подальшому у відповідність конкретні елементи інтерфейсу, наприклад, форми/сторінки, кнопки, пункти меню, фішки, плитки тощо.

Що стосується блоків, позначених сірим кольором (3, 4, 7, 8), то тут маємо певну неоднозначність, оскільки, кожен із цих кроків об’єднує кілька дій або підзадач, що ускладнює моделювання сценаріїв і може заплутати команду на етапі проектування UI або розробки логіки (таблиця 2.2).

Блоки на рис.2.14, позначені жовтим кольором насправді містять варіанти виконання, які можуть призвести до гілок в User Flow або умов у логіці, хоча Task Flow передбачає лінійність процесу (табл.2.3).

Таблиця 2.2 – Аналіз неатомарних блоків Task Flow на рис. 2.14

Крок	**Реальний список дій**	**Варіанти реалізації в інтерфейсі**
Крок 3: «Ознайомитися з описом та складом меню»	Перегляд загального опису (ціна, фото, тип події). Ознайомлення зі складом (інгредієнти, ваги, алергени).	Це можуть бути різні блоки інтерфейсу, або навіть різні екранні сторінки.
Крок 4: «Перевірити умови доставки та терміни»	Користувач може перевіряти: – зону доставки; – час очікування; – додаткову оплату; – політику скасування – інші умови.	В інтерфейсі традиційно ці дії розділюються, бо вони часто візуалізуються або реалізуються в різних місцях (банер, підменю, FAQ тощо).
Крок 7: «Вибрати дату та час доставки»	Формально – це два поля або кроки: – вибір дати (календар); – вибір часу (таймпікер або часовий слот).	У деяких продуктах ці дії відділяють на два етапи (наприклад, у формі зворотного виклику чи доставки).
Крок 8: «Ввести контактні та адресні дані»	Формально – це два блоки введення або кроки: – контактні дані (наприклад, ПІБ, телефон, email); – адреса доставки (наприклад, місто, вулиця, будинок, коментар).	Ці блоки часто мають окрему валідацію, різні поля й навіть логіку автозаповнення (наприклад, через Google Maps API для адреси).

Таблиця 2.3 – Аналіз помилкових блоків розгалуженняTask Flow на рис. 2.14

Крок	**Розгалуження**	**Варіанти реалізації в інтерфейсі**
Крок 6: «Вказати кількість гостей або наборів»	Тут є дві можливості: – користувач вводить кількість персон; – користувач обирає фіксовану кількість наборів (наприклад, «Меню на 10 осіб» × 3).	Це дві різні моделі обрахунку вартості й логіки заповнення, які вимагатимуть різного UX-рішення.
Крок 9: «Обрати спосіб оплати»	Типові варіанти розгалуження: – онлайн-оплата; – оплата при отриманні; – безготівковий рахунок (для корпоративних клієнтів).	Кожен варіант тягне за собою інший набір наступних дій, або навіть інтеграцію з платіжними системами.

Скоригований варіант Task Flow наведено на рис.2.15.

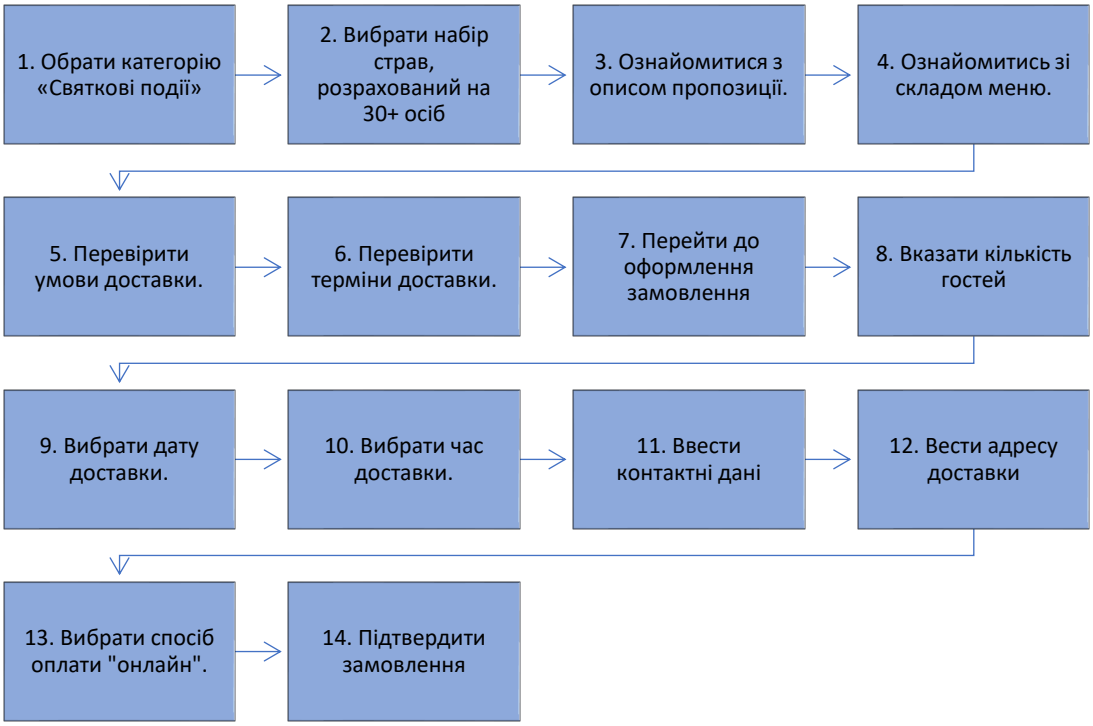


Рисунок 2.14 - Приклад Task Flow для сценарію «Замовлення святкового кейтерингу для ювілею». Скоригований варіант

2.7 Визначення структури інтерфейсу

Структура інтерфейсу застосунку представляє собою графоподібну структуру, вершинами якої є назви екранних форм, сторінок, спливаючих повідомлень чи інших елементів застосунку, а дуги/ребра описують переходи між ними (рис.2.15). Для веб-застосунків така структура може бути відображена у вигляді мапи сайту.

Як правило, навігація в застосунку починається з головної (домашньої) сторінки, але можуть бути виключення. В окремих випадках взаємодія з застосунком може починатись з деякої стартової сторінки, яка не пов’язана безпосередньо з виконанням задач користувача. Вміст такої

стартової сторінки може відрізнятись в залежності від передбачених способів взаємодії з застосунком.

Рисунок 2.15 – Ідея загальної структури мапи навігації

Умовно стартові сторінки можна поділити на 4 типи:

– ***Тип 1 – екран-заставка (Splash Screen), відображається під час завантаження застосунку*** (як правило, зображення, логотип і поточна версія програмного забезпечення). Екрани-заставки зазвичай використовуються особливо великими програмами для сповіщення користувача про те, що програма завантажується. Вони дають відгук про те, що йде ***тривалий процес***. Іноді індикатор прогресу на екрані-заставці вказує на прогрес завантаження. Екран-заставка зникає, коли з’являється головне вікно програми.

– ***Тип 2 - відображається лише при першому вході в застосунок***, наприклад, використовується для збору даних про користувача шляхом процедури реєстрації (*при цьому при подальших відкриттях застосунку додаток може і не потребувати авторизації, якщо інформація, що надає*

про себе користувач, не має конфіденційного характеру чи не потребує окремих обмежень на доступ!), для ознайомлення з політиками та правилами використання застосунку тощо.

– ***Тип 3 – відображається при кожному вході в застосунок і має тільки привітально-ознайомчий характер*** (виводиться дуже невелика кількість інформації про застосунок або «порада дня», або рекламна інформація, або інформація про актуальні акції тощо).

– ***Тип 4 – відображається при кожному вході в застосунок або при іншій події (наприклад, при кожному оновленні версії програми) для забезпечення авторизації в застосунку.***

ВАЖЛИВО!

В одному й тому ж застосунку можуть використовуватись всі чотири типи стартових сторінок!

Вершини в структурі інтерфейсу застосунку можуть бути позначені будь-яким способом, в тому числі, з використанням піктограм (рис.2.16).

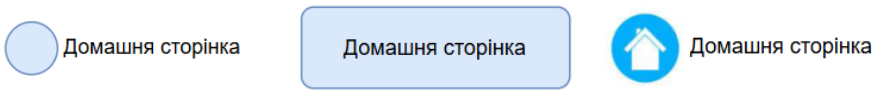


Рисунок 2.16 – Приклади позначення вершин

На відміну від класичних графів, при описі структури інтерфейсу застосунку можуть використовуватись одночасно дуги і ребра, а також додаткові кастомізовані типи зв’язків, які визначають специфічну взаємодію сторінок (рис.2.17). Наприклад, ребро може визначати перехід від однієї сторінки до другої і назад, а дуга – перехід тільки в одному напрямку без можливості повернення на попередню сторінку (наприклад,

при переході з екрану-заставки на головну ми не маємо можливості повернутись назад, або після онлайн-оплати замовлення ми не можемо повернутись на сторінку його редагування). Спеціальними лініями також можна позначити спливаючі повідомлення, модальні вікна та інше (на рис.2.17 пунктирною лінією з колом на початку лінії показано виклик спливаючого вікна). Лінії зв'язку або окремі елементи можуть мати підписи, що позначають певні ситуації/умови, при яких виникає потреба у вказаній взаємодії (на рис.2.17 – перехід на сторінку реєстрації відбувається тільки при першому запуску, в усіх інших випадках сторінка реєстрації не відкривається).

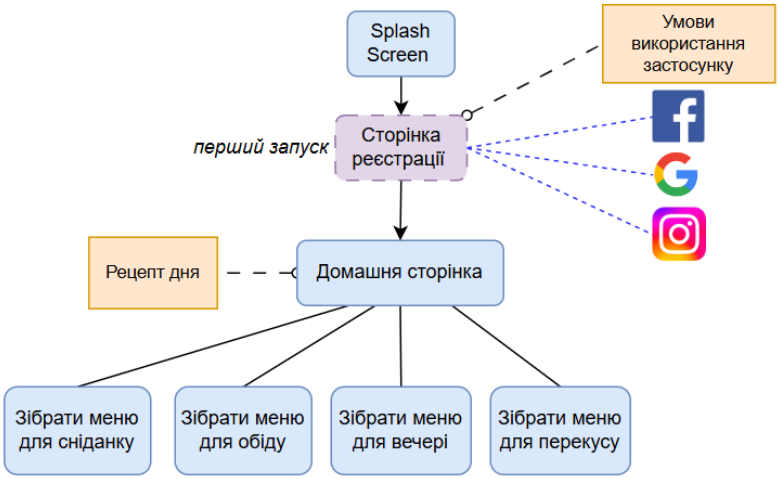


Рисунок 2.17 – Приклади різних позначень при описі структури інтерфейсу застосунку

ВАЖЛИВО!

Не існує стандартної системи позначень (кольори, типи ліній тощо). головне правило – використовувати однакову систему для позначень елементів та взаємодій однакового типу.

При розробці структури інтерфейсу застосунку варто враховувати довжину ланцюгів в побудованій схемі. Занадто велика глибина дерева навігації (рис. 2.18) означає, що користувачеві доведеться дуже довго добиратись до екранів, що відповідають певним задачам (пам’ятаємо правило 3 кліків , яке передбачає, що будь-яка важлива для користувача функція повинна бути доступна не більше ніж за 3 кліки. **Але!** Якщо для реалізації користувацької задачі потрібно більше 3 кроків і на кожному кроці користувачеві чітко зрозуміло, що саме він робить і до якого результат це призведе, то кількість кроків може бути значно більшою, ніж 3).

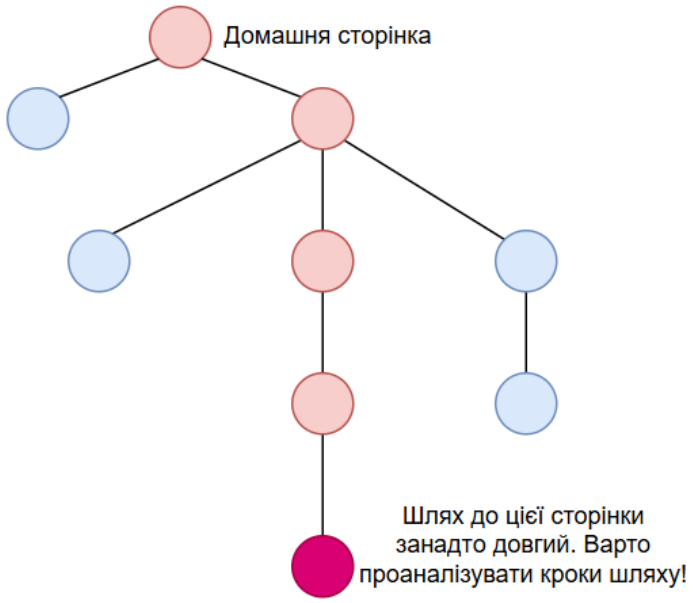


Рисунок 2.18 – Приклад ланцюгу, який потребує додаткового аналізу

Дуже широке дерево навігації (рис.2.19) демонструє велику кількість варіантів для вибору на одній сторінці, що може негативно позначитись на взаємодії.

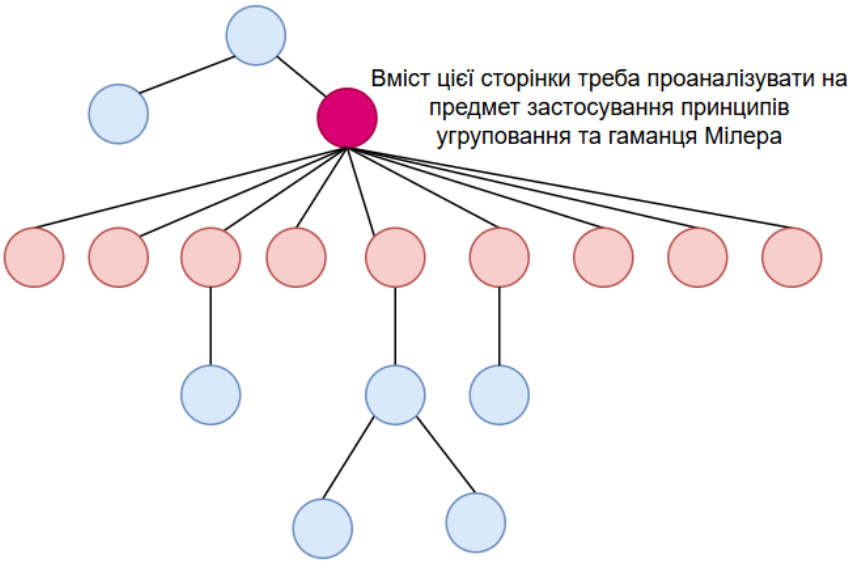


Рисунок 2.19 – Приклад набору зв’язків сторінки, який потребує додаткового аналізу

2.8 Аналіз альтернатив при проєктуванні інтерфейсу

Кожна з визначених структурою інтерфейсу сторінок може бути реалізована різними способами – як з використанням різних елементів інтерфейсу, так і з реалізацією різних схем взаємодії. Перед безпосереднім створенням макету інтерфейсу треба визначитись, який саме варіант буде використовуватись командою. Для цього можна використовувати різні підходи, але всі вони передбачають обговорення цих варіантів в команді та/або з безпосередніми замовниками чи реальним/потенційними користувачами. Спосіб структурування пропозицій ніяк не регламентується, тому може використовуватись текстова форма, таблиця, графічні схеми, презентації тощо. В таблиці 2.4 наведено фрагменту опису сторінок застосунку для підбору рецепту. З наведених варіантів реалізації інтерфейсу в подальшому повинен бути вибраний один, на основі якого буде створюватись макет (в таблиці виділено для прикладу зеленим кольором).

Таблиця 2.4 – Приклад фрагменту опису сторінок застосунку для підбору рецепту

Вміст сторінки	Можливі варіанти реалізації UI
Splash Screen (стартова 1) *(відображається при кожному вході в застосунок)*	
Коротка інформація про застосунок	1. Анімований логотип застосунку. 2. Логотип застосунку + Progress Bar. 3. Назва застосунку + Progress Bar.
Сторінка реєстрації (стартова 2) *(відображається лише при першому вході в застосунок, авторизація в застосунку не передбачена)*	
Привітання для користувача	1. Назва застосунку (текстовий підпис) + коротка текстова інформація про застосунок (текстовий підпис) + тематичне фото або піктограма + вітальна фраза (текстовий підпис). 2. Логотип застосунку (рисунок) + вітальна фраза (текстовий підпис). 3. Назва застосунку + вітальна фраза (тільки текстові підписи).
Засоби реєстрації в застосунку через інші акаунти (Google, Facebook, Instagram)	1. Вертикальний список з кнопок. Кожна кнопка містить логотип та назву застосунку для реєстрації. 2. Горизонтальний список з кнопок (*або фішок, або плиток*). Кожна кнопка містить тільки логотип застосунку для реєстрації.
Інформація для ознайомлення з політиками та правилами використання застосунку	Чекбокс для підтвердження ознайомлення з політиками та правилами використання застосунку + гіперпосилання на сторінку з детальним описом політик та правил використання застосунку.
Головна	
Засоби навігації для переходу на інші сторінки.	1. Плитки з рисунками, що асоціюються у користувача з відповідними діями в програмі. 2. Плитки з піктограмами та текстовими підписами. 3. Бургер-меню (**може використовуватись спільно з меню, представленим плитками, з повтором тих самих функцій і з додатковими другорядними функціями*). 4. Вертикальний список гіперпосилань з назвами пунктів меню (з піктограмами або без них). 5. Статусний рядок з піктограмами.
Випадково обраний з бази рецепт дня	* *Якщо для навігації на головній сторінці використовуються плитки чи вертикальний список гіперпосилань, то рецепт відображається нижче після елементів навігації.* 1. Скорочена картка рецепту, що вміщує: – фото страви, – назву страви (текстовий підпис), – де може бути використано: сніданок/обід/вечеря/перекус (варіанти

Вміст сторінки	Можливі варіанти реалізації UI
	реалізації: текстовий підпис, фішки, набір піктограм). – складність приготування (зірочки за шкалою в 5 градацій або дійсне число за шкалою від 1 до 10), – гіперпосилання на детальний опис рецепту. 2. Детальна картка рецепту, що вміщує: – фото страви, – назву страви (текстовий підпис), – складність приготування (зірочки за шкалою в 5 градацій або дійсне число за шкалою від 1 до 10), – список інгредієнтів (текст або текстовий список+піктограма, або фішки з текстовими підписами, або фішки з піктограмами+текстовими підписами), – текстово-графічний контент з описом кроків приготування (по кожному пункту: текст + 1 фото).
…	
Зібрати меню для вечері	
…	
Зібрати меню для обіду	
…	
…	

Отже, при розробці прототипів інтерфейсу дуже часто ми можемо мати декілька варіантів реалізації, а розробка макету для кожного з варіантів може займати багато часу та ресурсів команди. Крім того, при демонстрації замовникам та користувачам інтерактивного прототипу інтерфейсу, в них може скластись враження, що це вже працююча програма (іноді дуже важко пояснити стейкхолдерам, що якщо щось виглядає, як кнопка, натискається, як кнопка та дозволяє перейти на сторінку, що відповідає цій кнопці, - це ще не готовий функціонал а тільки демо-версія з заздалегідь підготовленими варіантами результатів).

2.9 Створення чорнового прототипу інтерфейсу

Щоб не витрачати час на детальну прорисовку кожного рішення та мати можливість пояснити зовнішнім стейкхолдерам чи членам команди концепт дизайну інтерфейсу, для чорнового прототипування використовується підхід на основі Wireframe.

Wireframe (також low-fidelity prototype, lo-fi) представляє собою образі низької точності відображення дизайну екрану. У Wireframe акцент робиться не на візуальну складову (кольори, шрифти, зовнішній вигляд елементів, конкретний контент тощо), а на розташування елементів, структуру та зміст екрану.

Wireframe має чітко показувати:

- основні групи контенту (що?);
- структуру інформації (де?);
- базову візуалізацію взаємодії між інтерфейсом та користувачем (як?).

Серед ключових особливостей Wireframe варто відмітити наступні:

- неповне представлення про кінцевий дизайн – в прототипі можуть бути реалізовані не всі види взаємодій (наприклад, відсутні мікровзаємодії);
- простий зовнішній вигляд – традиційно чорно-біло-сіра гама, синій колір для посилань, іноді може навіть використовуватись стиль «дитячого малюнку» (рис.2.20).

Типова система спрощень при створенні Wireframe представлена в таблиці 2.4. Щодо інших елементів UI, то вони можуть виглядати так само, як і в прототипі високої точності, однак стиль їх відображення та кольорова гама не повинні відрізнятись від інших елементів Wireframe.

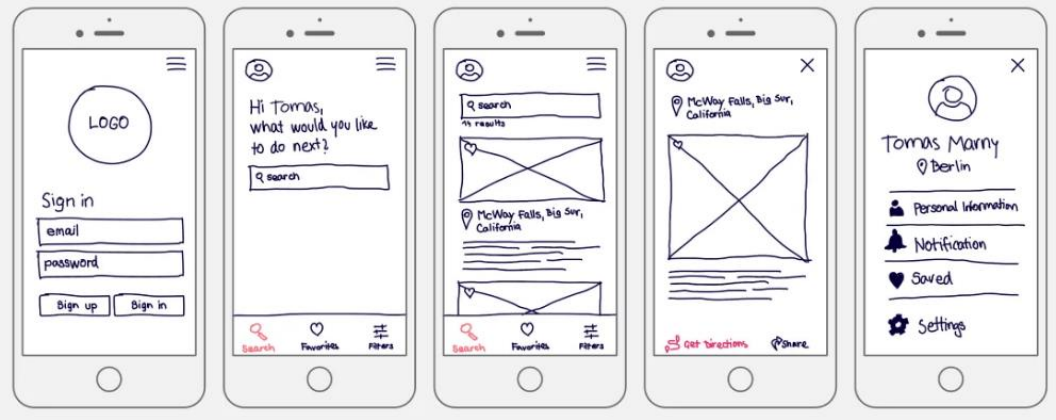
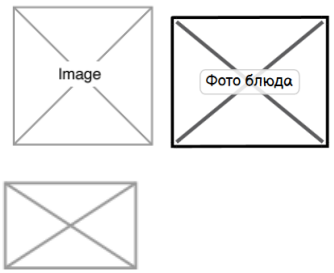
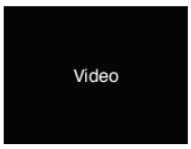
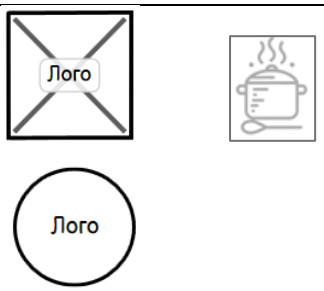


Рисунок 2.20 – Приклад Wireframes

Таблиця 2.4 – Приклади спрощених позначень елементів при створенні Wireframes

Елемент	**Графічне позначення**	**Примітка**
Зображення		Замість реального зображення відображається квадратна чи прямокутна перехрещена навхрест рамка. Всередині образу зображення може бути підпис, що визначає сенс зображення.
Відео		Замість відео-лементу відображається замальований квадрат чи прямокутник. Всередині образу може бути підпис, що визначає сенс відео, назву відеороліку тощо.
Логотип		Для представлення логотипів може бути використана така сама перехрещена рамка, що й для зображень, але з підписом «Лого» (“Logo”) всередині. Замість рамки може використовуватись коло. В деяких випадках ситуаціях Wireframe може бути містити зображення логотипів в сірому кольорі або логотипи низької точності.

Елемент	Графічне позначення	Примітка
Текст	**Lorem ipsum** "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."	Важливість або ієрархія інформації на сторінці створюється за допомогою заголовків, найчастіше напівжирного або тексту більшого розміру. Текст у Wireframe представляється або фактичною копією, або текстом-заповнювачем, таким як Lorem ipsum (https://www.lipsum.com/).
Посилання	Посилання	Посилання представлені найчастіше синім підкресленим текстом. Посилання також можуть мати інший колір, який відповідає певному напрямку візуального дизайну.

Дуже часто на основі Wireframes роблять **Wire flow** – це поєднання елементів блок-схеми та Wireframes, коли замість елементів блок-схеми використовують Wireframes. Акцент в дизайні сторінок в такому випадку робиться на елементах навігації, інші елементи можуть бути взагалі позначені символічно (рис.2.21).

Таке представлення доцільно використовувати при розробці мобільних застосунків за рахунок відносно невеликих розмірів Wireframes (рис. 2.22), однак, для сторінок великого розміру такий підхід не завжди дозволяє повноцінно представити макети сторінок (рис.2.23).

Для створення Wireframes можна використовувати будь-які застосунки, що дозволяють працювати з графічними зображеннями, але зручніше використовувати такі, що мають готові шаблони елементів UI для чорнового прототипування інтерфейсів, наприклад:

- https://balsamiq.com/wireframes/,
- https://wireframepro.mockflow.com/,
- https://miro.com/,

– https://www.figma.com/
– та інші.

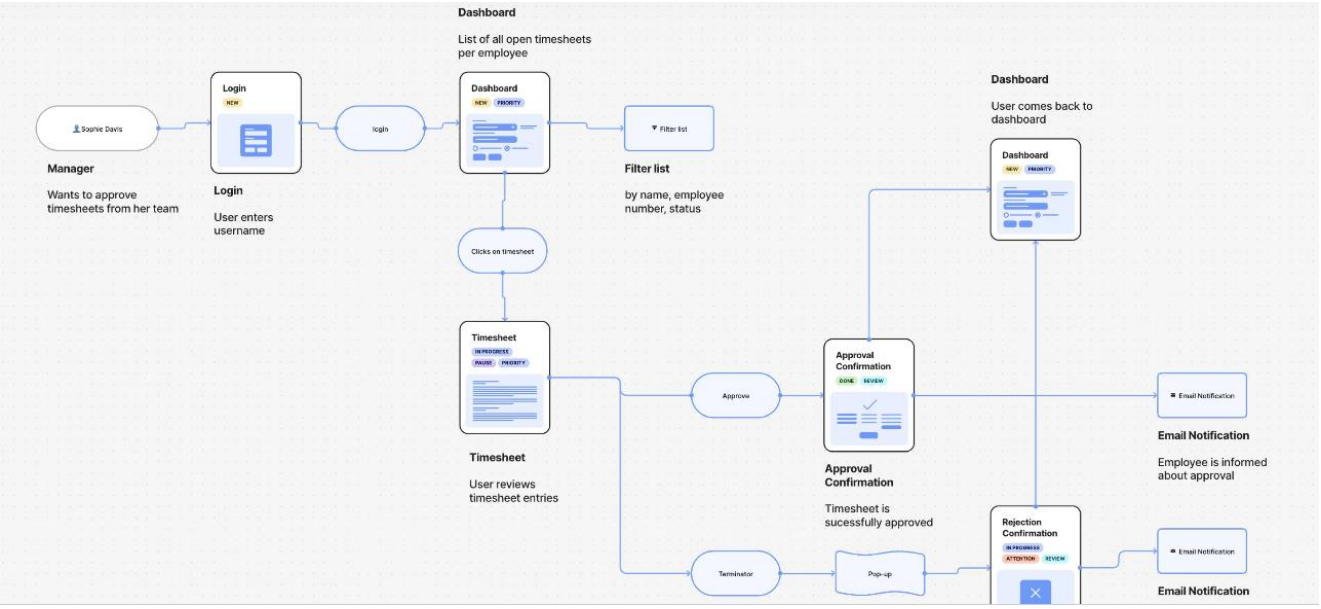


Рисунок 2.21 – Ілюстрація потоку користувача в мобільному застосунку з використанням Wireframes – вміст сторінок надмірно спрощено, акцент на навігацію (джерело https://www.flowmapp.com/features/wireflows-comprehensive-guide)

Перевагою використання спеціалізованих застосунків є можливість створювати інтерактивні прототипи, в яких властивості та поведінка «чорнових» елементів цілком відповідає реальним елементам UI. Крім того, в таких застосунках є можливість пов’язати Wireframes між собою, імітуючи таким чином поведінку користувача в системі. Також в спеціалізованих застосунках як правило присутні «контейнери сторінок» реальних пристроїв, наприклад смартфонів, планшетів тощо, що значно спрощує процес розміщення елементів, оскільки дизайнеру не потрібно рисувати додатково сам пристрій, для якого створюється макет.

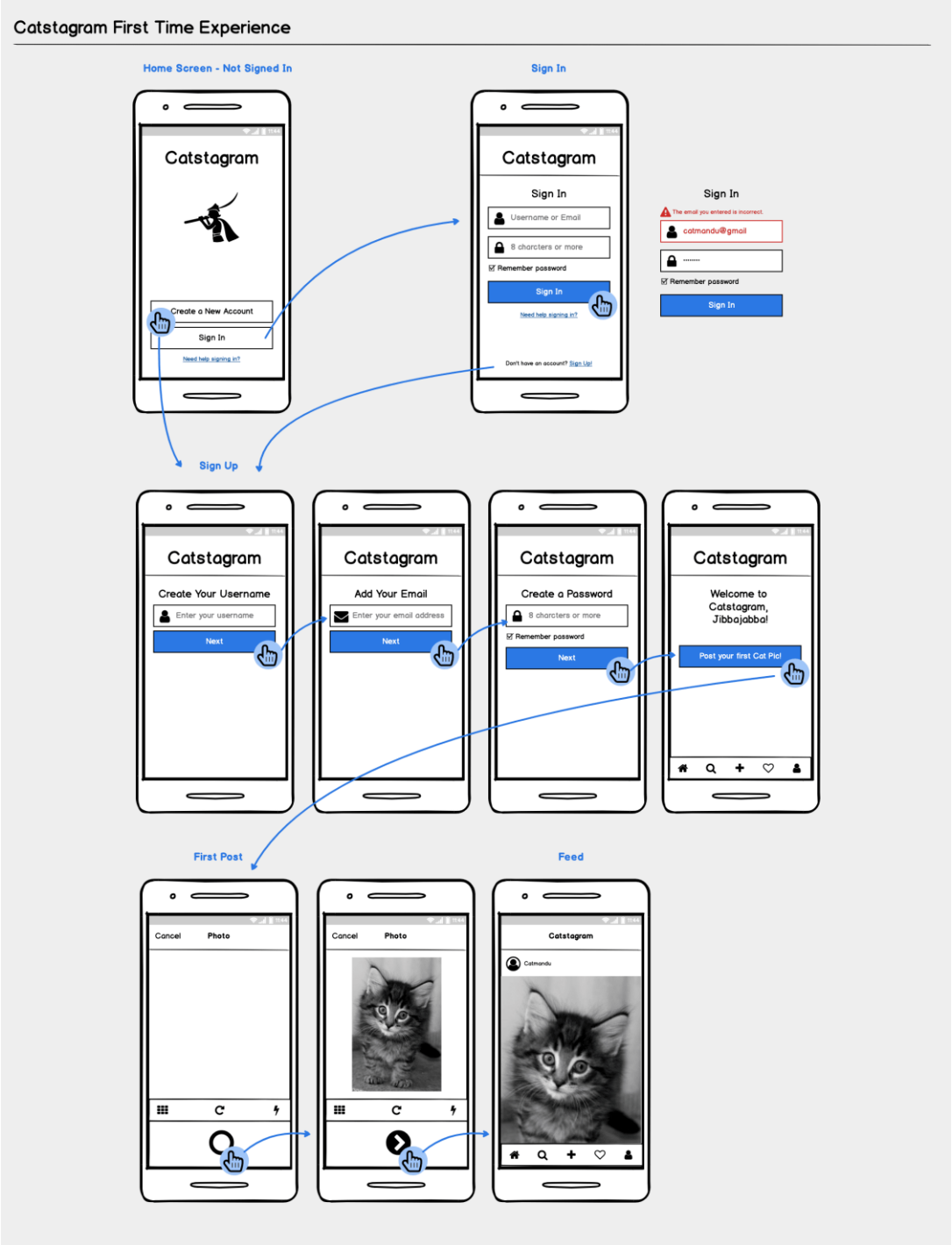


Рисунок 2.22 – Ілюстрація потоку користувача в мобільному застосунку з використанням Wireframes – вміст сторінок деталізовано (джерело https://balsamiq.com/learn/articles/wireflows/)

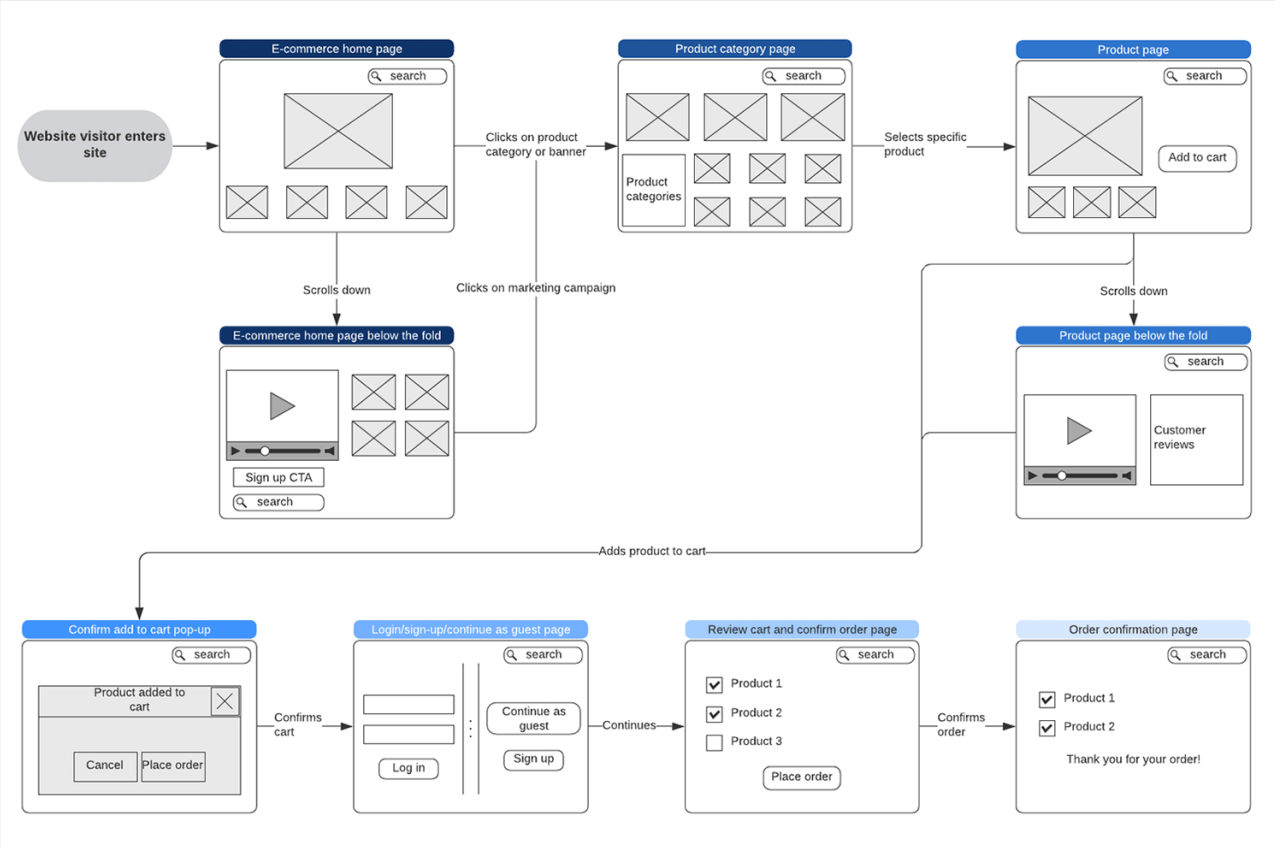


Рисунок 2.23 - Ілюстрація потоку користувача в веб-застосунку з використанням Wireframes – розміри сторінок не витримано, всі елементи позначено умовно, продемонстрована лише ідея вмісту сторінок (джерело https://www.justinmind.com/blog/user-flow/)

РОЗДІЛ 3. РОЗРОБКА ІНТЕРФЕЙСУ

3.1 Чистовий інтерфейс

3.1.1 Поняття та призначення чистового інтерфейсу

Після етапу розробки Wireframes команда дизайнерів переходить до створення чистової версії інтерфейсу користувача – так званого високодеталізованого макету (high-fidelity mockup). Цей етап передбачає втілення візуального стилю продукту, остаточний добір UI-компонентів, типографіки, кольорової гами, іконографії та просторових відношень між елементами. Якщо Wireframes відповідають на запитання «що і де буде розміщено?», то чистовий інтерфейс – це відповідь на запитання «як саме це виглядатиме і як сприйматиметься?»

Під «чистовим» інтерфейсом (high-fidelity interface) у сфері UX/UI-дизайну розуміють остаточно опрацьований, візуально сформований варіант інтерфейсу, який включає усі ключові графічні й типографічні рішення, компоненти взаємодії та відображає повну логіку користувацької поведінки на рівні візуального представлення. У межах цього етапу визначається остаточна реалізація принципів, закладених у каркасних схемах: уточнюється функціональне навантаження елементів, уточнюються просторові взаємозв’язки між ними, застосовується кольорова палітра, підбираються шрифти, іконографіка та стилістика ілюстрацій. Чистовий інтерфейс, на відміну від каркасу, не обмежується схематичним зображенням логічної структури. Він містить усі ознаки візуальної завершеності й максимально наближений до того вигляду, який побачить кінцевий користувач. Разом із тим, макет все ще залишається статичним – його головна функція полягає в демонстрації зовнішнього вигляду

інтерфейсу та підготовці до наступного етапу – інтерактивного прототипування або передачі в розробку.

Варто чітко відрізняти макет (mockup) як статичне візуальне представлення інтерфейсу та сам інтерфейс у ширшому сенсі – як поєднання візуальних, функціональних і поведінкових патернів. Хоча макети на цьому етапі можуть мати дуже високу точність (до пікселя), вони все ще не є «живим» інтерфейсом, а радше його проєкцією. Водночас саме вони визначають базу для подальшої інтеграції в розробку та взаємодії з програмною частиною.

3.1.2 Вимоги до чистового інтерфейсу

Робота над чистовим інтерфейсом повинна відповідати таким вимогам:

– візуальна завершеність – застосування фірмових кольорів, шрифтів, компонентів інтерфейсу (UI-елементів), графічних стилів відповідно до брендбуку або дизайн-системи;

– функціональна узгодженість – кожен елемент повинен мати чітко визначену функцію, логічне розташування та інформативне візуальне оформлення;

– консистентність – повторюваність патернів, стилів та візуальних рішень на всіх екранах і в різних сценаріях взаємодії;

– доступність (accessibility) – дотримання принципів доступного дизайну, зокрема контрастність тексту, масштабованість елементів, урахування сенсорних і когнітивних особливостей користувачів;

– відображення усіх станів елементів – наявність прикладів вигляду інтерактивних компонентів у різних станах: активному, неактивному, наведеному, вибраному, з помилкою тощо.

3.1.3 Сітки в дизайні інтерфейсів

Поняття сітки в дизайні інтерфейсів посідає ключове місце як організаційний інструмент, що дозволяє створювати впорядковану, логічну та візуально узгоджену структуру простору. Основна мета сітки – забезпечення впорядкованості, узгодженості та ритмічності у візуальному представленні, а також полегшення адаптації дизайну до різних екранів.

В основі будь-якої сітки лежить принцип регулярного поділу простору на зони, які можуть бути вертикальними, горизонтальними або комбінованими. Незалежно від конкретної конфігурації, сітка виконує подвійну функцію: структурну (забезпечення впорядкування) та семантичну (формування ієрархії значень і взаємозв'язків). Її наявність допомагає дизайнеру не лише дотримуватися композиційної логіки, а й швидше приймати рішення щодо вирівнювання, розмірів і відступів, що особливо важливо у проєктах, де застосовується системний підхід до візуального оформлення.

Сітки можна використовувати, розділивши веб-сторінку на певну кількість стовпців і рядків, де ви розміщуєте елементи. Кожен елемент можна розмістити в цих стовпцях і рядках. Стовпці та рядки сітки встановлюються властивостями CSS, які визначають їхню ширину, висоту та інтервал.

У практиці дизайну виокремлюють кілька типів сіток, кожна з яких має власне функціональне призначення та історичне походження. Залежно від складності, призначення та контексту використання, обирається відповідна конфігурація сітки. Найбільш поширеними є такі типи:

- манускриптна сітка**,**
- колонкова сітка**,**
- модульна сітка**,**
- сітка на основі базової лінії,

– ієрархічна сітка**.**

Манускриптна сітка є найпростішим типом сітки. Вона побудована на єдиному великому прямокутному полі, що визначає межі розміщення основного текстового або візуального блоку. Такий формат особливо поширений у традиційних макетах книг, постерів або сторінок з переважанням одного типу контенту (рис.3.1).

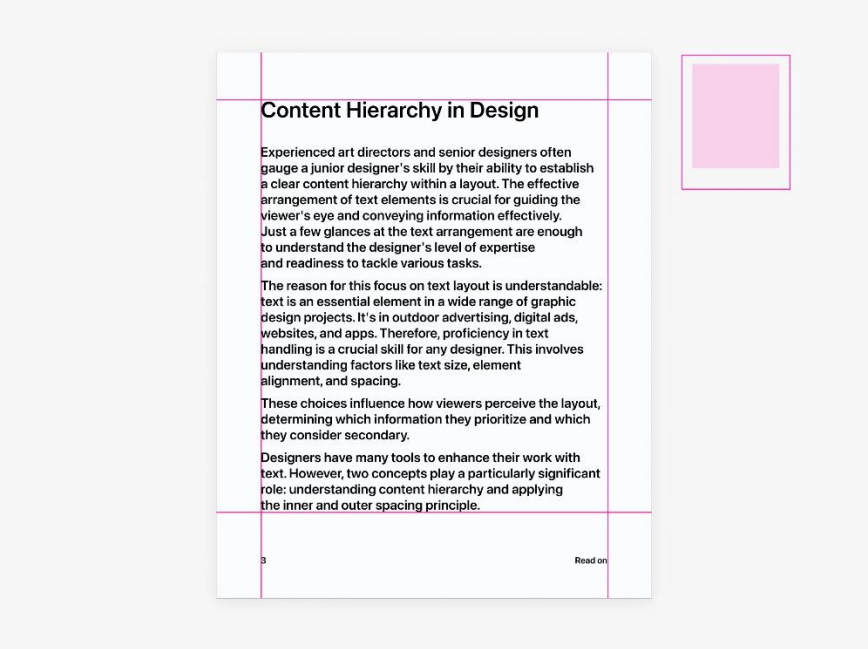


Рисунок 3.1 – Приклад манускриптної сітки
(джерело https://cases.media/en/article/grids-in-websites)

У веб- та інтерфейсному дизайні манускриптна сітка використовується для створення одноколонкових макетів, наприклад, у мобільних застосунках або лендінгах, де основна увага зосереджена на одному змістовому потоці. Вона забезпечує високий рівень читабельності, але обмежує дизайнерську гнучкість у складних структурах.

Колонкова сітка ґрунтується на поділі простору на вертикальні колонки з визначеними інтервалами між ними (так званими «канавками» або gutters). Вона є основною організаційною структурою в більшості

сучасних веб-інтерфейсів, що зумовлено її здатністю адаптувати контент до різних розмірів екранів та утримувати композиційну рівновагу.

Рисунок 3.2 – Приклад колонкової сітки
(джерело https://cases.media/en/article/grids-in-websites)

Колонкова сітка може бути симетричною або асиметричною, мати фіксовану або відсоткову ширину колонок. Найпоширенішим варіантом у вебдизайні є 12-колонкова сітка (рис.3.3), яка забезпечує гнучке компонування елементів у різних пропорціях (наприклад, 6+6, 4+8, 3+9 тощо).

Застосування колонкової сітки сприяє побудові масштабованої та адаптивної структури, в якій легко поєднувати текст, зображення та інтерактивні компоненти.

Модульна сітка є розвитком колонкової, з тією відмінністю, що до вертикальних колонок додаються горизонтальні ритмічні лінії (рис.3.4). В результаті утворюється мережа прямокутників (модулів), які створюють основу для більш складних та щільно структурованих макетів.

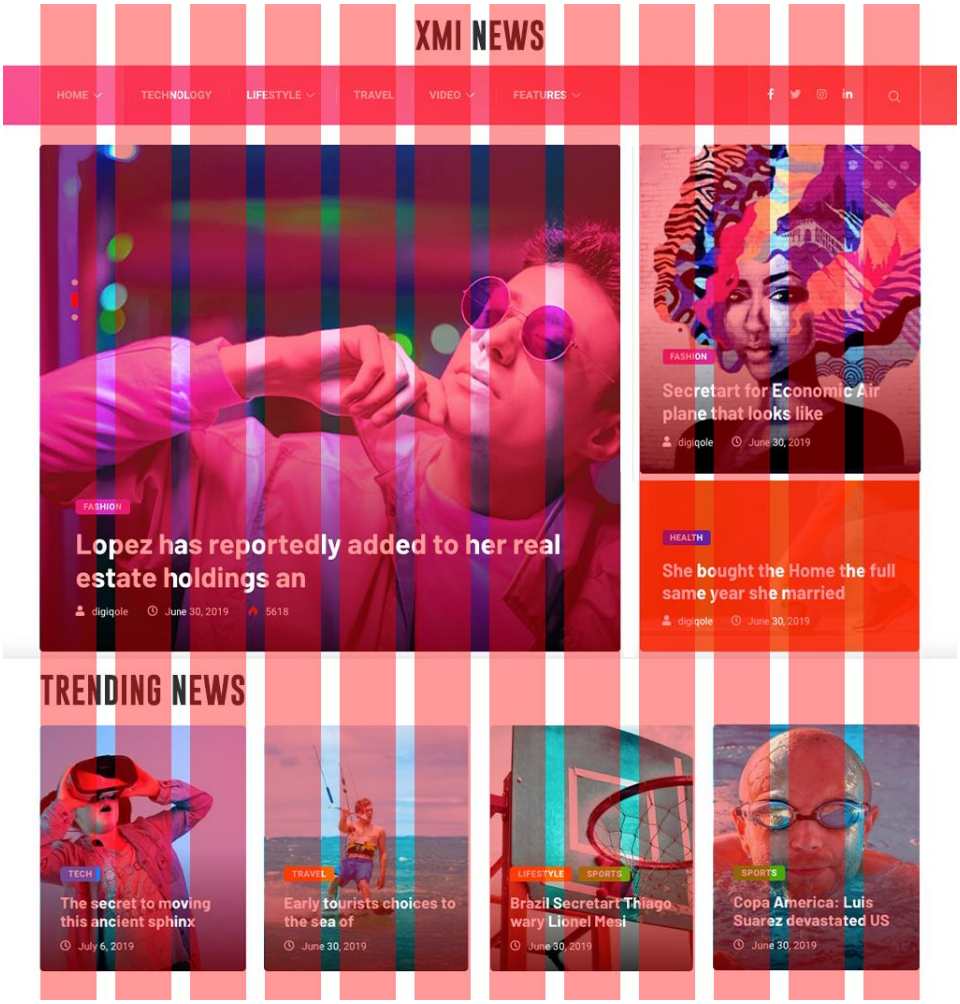


Рисунок 3.3 – Приклад 12-колонкової сітки

(джерело https://www.ramotion.com/blog/website-grid-design/)

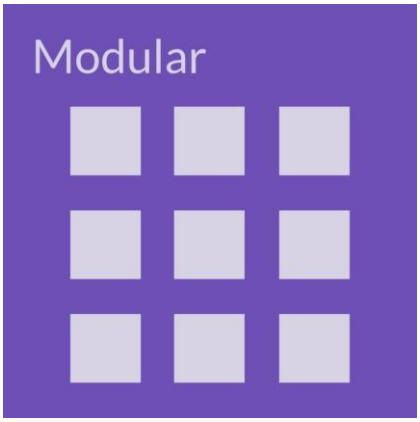


Рисунок 3.4 – Структура модульної сітки

Цей тип сітки особливо ефективний у тих випадках, коли потрібно розміщувати багато однотипних елементів, наприклад, у каталогах, галереях, дашбордах або адміністративних панелях. Модульна сітка дозволяє встановити чіткі межі між елементами та забезпечити рівномірне візуальне наповнення простору (рис. 3.5).

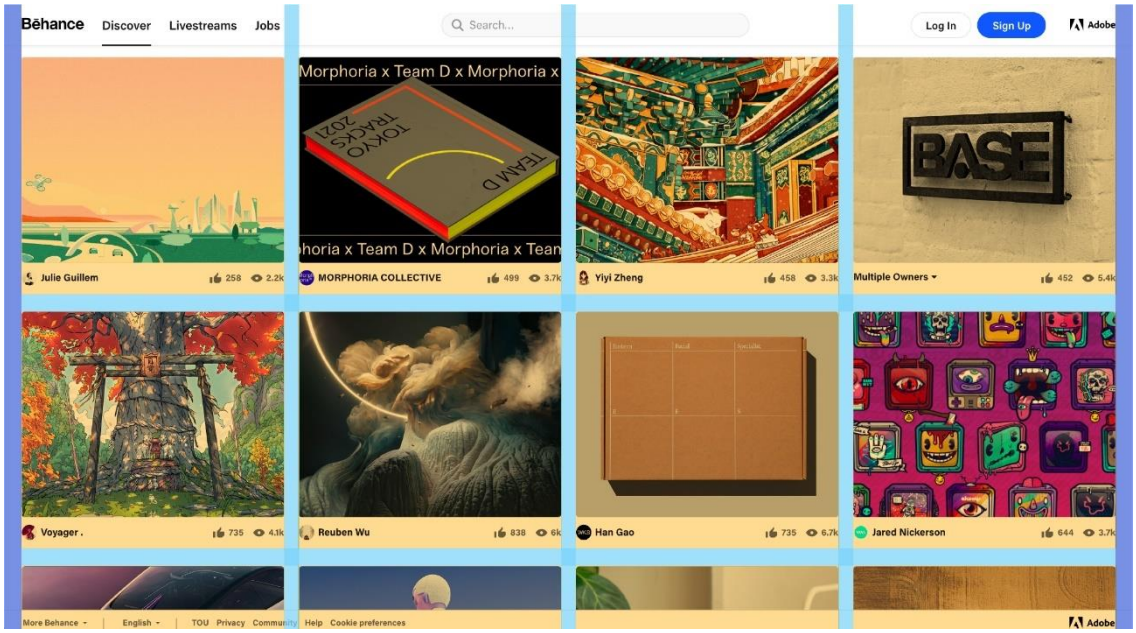


Рисунок 3.5 – Приклад модульної сітки

(джерело https://www.behance.net/)

Сітка на основі базової лінії побудована з урахуванням розміщення текстових елементів таким чином, щоб усі рядки тексту узгоджувалися за вертикальною лінією – базовою лінією шрифту. Це особливо важливо для текстових макетів, де велике значення має збереження вертикального ритму та вирівнювання між різними типами блоків (рис 3.6).

Застосування базової сітки сприяє високому рівню типографічної впорядкованості, забезпечуючи злагоджене читання й естетичну гармонію у випадках зі складною типографікою (наприклад, журнальні статті, освітні платформи, інформаційні ресурси) (рис.3.7).

Рисунок 3.6 – Структура сітки на основі базової лінії

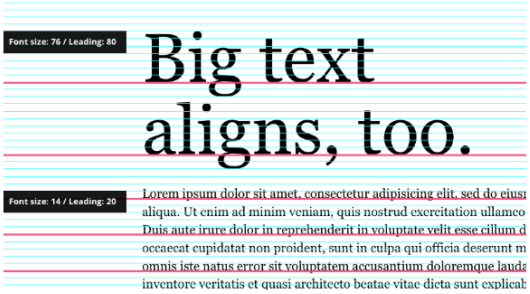


Рисунок 3.7 – Приклад розміщення елементів за сіткою на основі базової лінії (джерело https://www.uxdesigninstitute.com/blog/how-to-use-grids-in-web-design/)

Ієрархічна сітка не має чіткої регулярної структури, як у попередніх типів. Вона будується на основі індивідуального змісту, логіки користувацького потоку та контексту презентації, що дозволяє більш вільно розміщувати елементи відповідно до їхньої ваги та функції (рис.3.8).

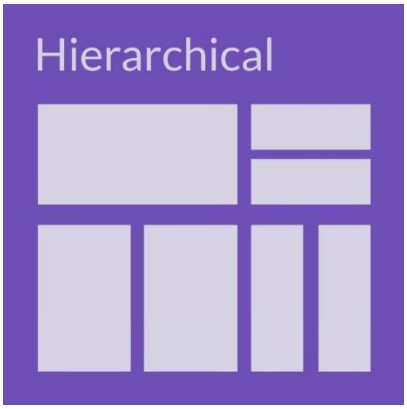


Рисунок 3.8 – Приклад структури ієрархічної сітки

Такий тип сітки часто застосовується в дизайнерських рішеннях, орієнтованих на емоційний вплив, креативні інтерфейси, промоційні сайти або експериментальні макети. Ієрархічна сітка дає змогу зберегти баланс між візуальною виразністю та інформаційною структурою, хоча й вимагає високого рівня професійної підготовки від дизайнера (рис.3.9).

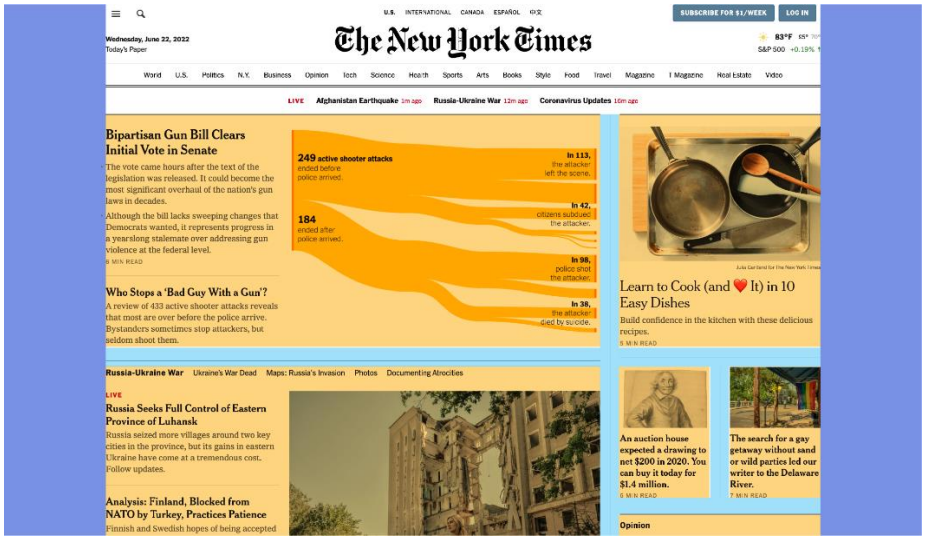


Рисунок 3.9 – Приклад розміщення елементів на основі ієрархічної сітки (джерело https://www.nytimes.com/)

Сітка як інструмент композиції виконує функцію опорної структури, на якій базується вся логіка візуального проєктування інтерфейсу. Обираючи тип сітки, дизайнер має виходити з особливостей контенту, цілей взаємодії та рівня складності продукту. Розуміння властивостей різних типів сіток та вміння їх адаптувати до конкретного завдання є невід'ємною складовою професійної компетентності фахівця у сфері UX/UI-дизайну.

3.1.4 Формування візуальної мови інтерфейсу

У межах цього етапу визначаються або впроваджуються такі візуальні компоненти:

– кольорова палітра – основні, допоміжні, статусні кольори (наприклад, для повідомлень про помилки, підтвердження, інформацію);

– типографіка – добір гарнітур шрифтів, розмірів, міжрядкового інтервалу, стилів заголовків і текстових блоків;

– іконографіка та ілюстративні елементи – зображення, піктограми, декоративні компоненти відповідно до загальної стилістики продукту;

– компоненти інтерфейсу – інтерактивні елементи, які повинні мати єдині розміри, поведінку та стилі відповідно до системи компонентів (див. в пункті «Масштабованість і системність»).

3.1.5 Інтерактивність і сценарії використання

Незважаючи на статичний характер чистового макету, на цьому етапі необхідно враховувати сценарну поведінку користувача: переходи між екранами, стани компонентів, можливі дії, які потребуватимуть зворотного зв’язку з боку системи. У макеті або в супровідному файлі дизайнер фіксує візуальні прояви цих сценаріїв – наприклад, як виглядає форма до й після заповнення, повідомлення про помилку, завантаження, підтвердження дії тощо.

Приклад 1: Поле введення у формі.

У макеті варто передбачити не лише нейтральний стан текстового поля, а й такі стани:

- Focus – поле активне: підсвічена рамка, курсор у полі, можливо, підказка;
- Filled – поле заповнене коректними даними;
- Error – введено некоректні дані (наприклад, неправильний формат електронної адреси): поле підсвічується червоним, з’являється повідомлення з описом помилки;
- Disabled – поле недоступне для редагування (наприклад, попередньо заповнене даними з профілю).

У Figma або іншій дизайн-системі ці стани можуть бути винесені окремо у вигляді візуального набору варіантів одного компонента, що забезпечує консистентність під час розробки.

Приклад 2: Кнопка.

Кнопка як базовий елемент взаємодії повинна мати щонайменше такі стани:

- Default – базовий вигляд: фоновий колір, текст, іконка (якщо передбачено);
- Hover – змінений колір фону або обвідки при наведенні;
- Pressed – реакція на натискання: затемнення, зсув, анімація;
- Disabled – сірий вигляд, кнопка неактивна;
- Loading – замість тексту з’являється індикатор (spinner), який сигналізує про очікування.

Наприклад, після натискання кнопки «Надіслати форму», кнопка змінюється на loading state, а після успішного надсилання – на «Готово» із зеленою піктограмою.

Приклад 3: Навігаційні переходи.

У чистовому макеті варто передбачити візуальні маркери переходу між екранами або станами, зокрема:

- виділення активної вкладки в меню (наприклад, підсвічення розділу «Профіль»);
- сторінка, що відкривається після дії, наприклад, підтвердження реєстрації або оновлення замовлення;
- згортання/розгортання блоків (акордеон, фільтри), які змінюють свій вигляд під час взаємодії.

Такі переходи можна відтворити як набір статичних станів в одному макеті, або реалізувати у вигляді мікропрототипу з базовою інтерактивністю.

Приклад 4: Спадаючі списки, модальні вікна.

Усі компоненти, які відкриваються або з’являються після взаємодії, повинні бути:

- спроєктовані окремо (наприклад, drop-down-меню з опціями),
- узгоджені з основним макетом за стилем і розмірами,
- проілюстровані у різних варіантах (наприклад, список заповнений/порожній).

Так, у разі проєктування інтерфейсу покупки квитка варто продемонструвати:

– список міст із автозаповненням;

– модальне вікно з попередженням про зміни умов оплати;

– повідомлення про успішну оплату.

Таким чином, візуалізація сценарної поведінки користувача в межах чистового інтерфейсу є необхідною умовою створення передбачуваного, доступного й ефективного продукту. Макети повинні демонструвати не лише статичні екрани, а й реакції інтерфейсу на ключові події. Такий підхід дозволяє забезпечити логічність дизайну, підтримку з боку розробки та відповідність очікуванням користувача.

3.1.6 Масштабованість і системність. UI Kit та дизайн-системи

У процесі проєктування складних цифрових продуктів – зокрема вебплатформ, мобільних застосунків, багатофункціональних сервісів – надзвичайно важливо забезпечити масштабованість інтерфейсу. Ідеться не лише про технічну адаптацію до різних пристроїв або роздільних здатностей екранів, а про здатність інтерфейсу зберігати цілісність, узгодженість і керованість у міру його розширення, оновлення чи модифікації.

Масштабованість у дизайні означає, що:

- інтерфейс можна безболісно доповнювати новими елементами (екранами, мовами, функціями);
- зміни стилю або логіки поведінки легко застосовуються до всієї системи;
- продукт зберігає єдиний візуальний і функціональний стиль навіть при зростанні команди або частоти релізів.

Приклад:

Платформа електронної комерції, яка спочатку має 5 типів екранів, через рік отримує:

- розділ для постачальників (новий тип ролі користувача),
- інструмент аналітики,
- темну тему.

Якщо продукт побудовано на основі системи, додавання цих нових модулів не призводить до стилістичних або функціональних суперечностей.

Системний підхід до дизайну передбачає не ізольоване створення окремих екранів, а розробку уніфікованої дизайн-системи – структурованого набору елементів, що регламентують візуальні й функціональні рішення продукту (табл. 3.1).

Дизайн-система (Design System) – це набір узгоджених стилів, UI-компонентів, гайдлайнів, шаблонів та принципів, які використовуються для побудови інтерфейсу. На відміну від бібліотеки макетів, яка фіксує візуальні стани, дизайн-система є динамічним інструментом, що підтримує розширення проєкту, забезпечує єдність і дозволяє залученим командам працювати узгоджено.

Таблиця 3.1 – Зведена інформацію по UI Kit та дизайн-системам (на основі матеріалів https://ux.pub/sajaden/ui-kit-design-system-guidelines-riznitsia-2onf)

	UI Kit	**Дизайн Система**
Що це таке	Збірка готових графічних елементів (кнопки,	Комплексне рішення, яке об'єднує не лише графічні

	UI Kit	**Дизайн Система**
	форми, іконки, типографіка тощо), які можна використовувати для швидкого створення інтерфейсів.	елементи, але й стандарти, принципи та код, спрямоване на підтримку консистентності, ефективності та якості дизайну в довготривалих проєктах.
Використання	Зазвичай застосовується для невеликих проєктів або для швидкого прототипування, коли необхідно заощадити час на розробці дизайну.	Для середніх та великих проєктів, які потребують уніфікації дизайну та коду в багатьох командах і продуктах.
Складові	– Компоненти інтерфейсу (кнопки, поля вводу, селектори тощо) – Типографіка – Іконки – Зразки кольорів	*Компоненти:* Повторювані UI-компоненти, які часто супроводжуються кодом (React, Vue, Angular тощо), що забезпечує їхню інтеграцію в продукт. *Дизайн-токени:* Стандарти кольорів, шрифтів, розмірів, відступів тощо, які використовуються в компонентах.
Документація	Часто відсутня або мінімальна.	Повний опис принципів дизайну, використання компонентів, стилів тощо.
Гайдлайни	Може не мати чітких правил та рекомендацій використання.	Рекомендації та стандарти щодо використання компонентів та стилів.
Інструменти	UI Kit може бути частиною дизайн-системи чи розроблятись окремо під проєкт, на продаж чи для інших потреб.	Інтеграція з інструментами розробки та дизайну (Figma, Sketch, Storybook, GitHub).

	UI Kit	**Дизайн Система**
Приклади використання	Прототипи, невеликі веб-сайти, мобільні застосунки, MVP.	Великі веб-сайти, мобільні застосунки, корпоративні продукти, екосистеми продуктів.
Корисні посилання	Free UI Kits for Seamless Designs - Figma Курс "UI Kit"	Три найкращі дизайн-системи Figma у 2023 році для UI/UX дизайнерів

Приклад:

У великій онлайн-платформі з розділами: «Профіль», «Замовлення», «Підтримка», «Платежі» – у всіх модулях використовуються ті самі:

- стилі заголовків (наприклад: H1 – 32 px, Inter Bold; H2 – 24 px),
- кольори статусів (успішна дія – зелений #2ECC71, помилка – червоний #E74C3C),
- кнопки (первинна – синя, вторинна – біла з сірою обвідкою),
- поля введення (один компонент для всієї форми в UI Kit).

Це дозволяє не створювати нові рішення для кожного підрозділу, а масштабувати єдині патерни, економлячи ресурси й підвищуючи якість.

Системний підхід до побудови інтерфейсу передбачає:

- узгодженість стилю та логіки дій на різних екранах (однакове поводження кнопок, однакова поведінка повідомлень, повторювані шаблони розмітки);
- повторне використання компонентів (наприклад, один шаблон картки товару використовується у пошуку, обраних і в історії замовлень);
- підтримку адаптивності через продуману grid-систему та модульні блоки;
- можливість внесення змін централізовано, що особливо важливо у великих продуктах.

В якості практичних інструментів для забезпечення системності використовують:

- UI Kit – набір стандартних візуальних компонентів (кнопки, поля, чекбокси) з варіантами станів;
- компонентну бібліотеку у Figma або аналогічному інструменті зі зв’язками між шаблонами й варіантами;
- токени дизайну (design tokens) – системні змінні для кольорів, шрифтів, відступів, які застосовуються програмно;
- документацію (Style Guide, Component Reference) для передачі дизайнерських рішень розробникам.

3.2 Мікровзаємодії

3.2.1 Поняття та види мікровзаємодій

Крім безпосередніх переходів на іншу сторінку в застосунках можливі так звані мікровзаємодії, які призводять до зміни зовнішнього вигляду елементів сторінки, шо вимагає розробки додаткового дизайну цих елементів чи сторінки в цілому (рис.3.10).

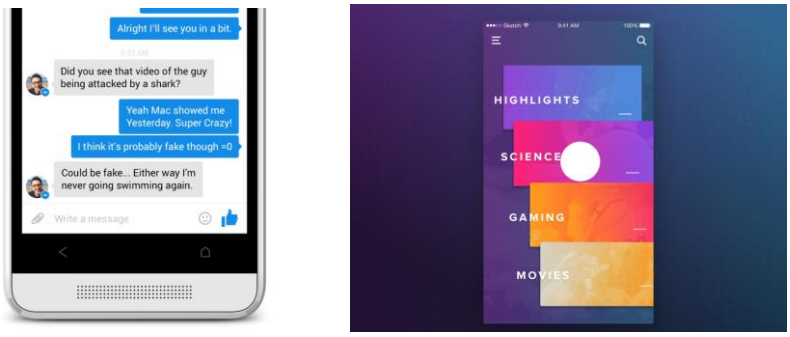


Рисунок 3.10 – Приклади мікровзаємодій

Мікровзаємодія – це невеликий (іноді зовсім крихітний) функціональний елемент, що виконує одну конкретну дію. На відміну від

звичайної сторінки/форми, мікровзаємодія часто не передбачає переходу на іншу сторінку, а змінює щось на поточній (але є і виключення!).

Типові ситуації для використання мікровзаємодії:

– Відображення статусу та прогресу системи (в тому числі, анімовані представлення) (рис.3.11).

Рис.3.11 – Приклади мікровзаємодії для відображення стану системи

– Заклик до дії (поява CTA-елементу або зворотній зв’язок на CTA-елемент – зміна кольору/вмісту елементу, поява смайліка тощо) (рис.3.12).

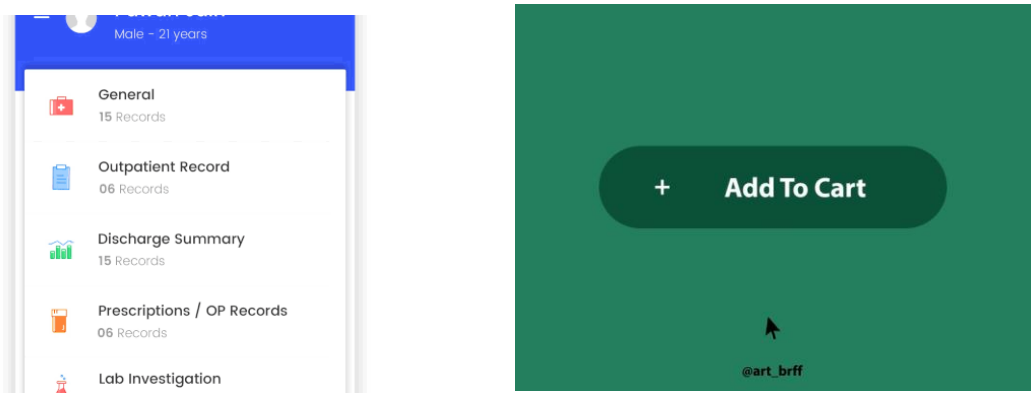


Рисунок 3.12 – Приклади мікровзаємодій заклику до дії

– Візуалізація введення даних (рис.3.13).
– Запобігання помилкам (рис.3.13).

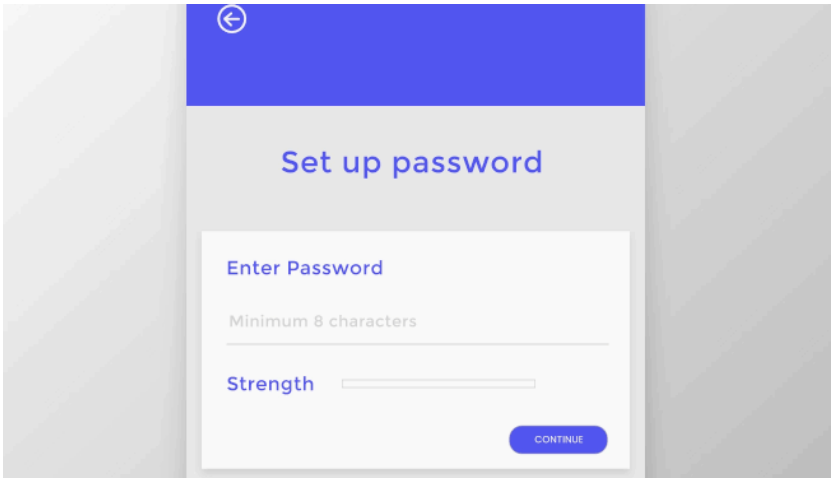


Рисунок 3.13 – Приклад мікровзаємодії візуалізації введення даних та запобігання помилкам

– Гумор (рис.3.14)

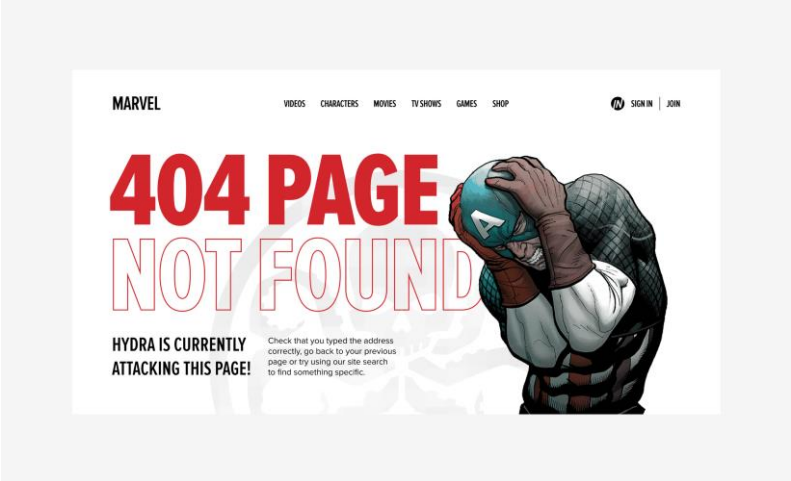


Рисунок 3.14 – Приклад мікровзаємодії що реалізує іронічне повідомлення при переході на неіснуючу сторінку.

– Підказки в реальному часі (рис.3.15).
– Пожвавлення режиму очікування.
– Відміна дії.
– Гейміфікація.
– Інше.

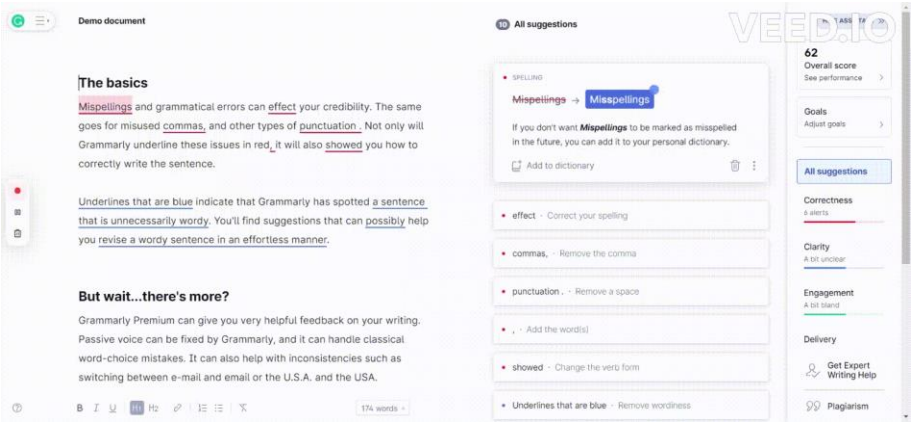


Рисунок 3.15 – Приклад мікровзаємодії,
що реалізує підказку в реальному часі

3.2.2 Компоненти мікровзаємодії

При проєктуванні мікровзаємодії необхідно чітко визначити її компоненти. До них відносяться тригери, правила, зворотній зв’язок, цикли та режими.

Тригери. Ініціюють мікровзаємодію. Тригери можуть бути ініційовані користувачем чи системою. У першому випадку тригер ініційований якоюсь користувальницькою дією. У ситуації системного тригера програмне забезпечення виявляє виконання певних умов та ініціює дію.

Правила. Визначають, що відбувається, коли спрацьовує тригер, щоб результат відповідав очікуванням користувачів. Наприклад, коли користувач натискає на іконку, можуть бути різні дії. Натискання може запустити анімацію, вивести користувача із системи або запустити іншу специфічну функцію.

Зворотній зв'язок. Дозволяє користувачам дізнатися, що відбувається. Все, що користувач бачить, чує або відчуває під час мікровзаємодії, є зворотним зв'язком. Зворотній зв'язок підтверджує користувачеві, що система розпізнала його дії.

Зворотний зв’язок може бути наданий у кілька способів:

- *Візуально*: іконки, анімація чи зміна кольору;
- *Аудіально*: звуки або сповіщення;
- *Тактильно*: вібрації або фізичні реакції.

Наприклад, iPhone вібрує, щоб показати, що користувач перевів його в беззвучний режим.

Цикли та режими. Визначають мета-правила мікровзаємодії. Цикли визначають, як довго триває мікровзаємодія, зокрема чи повторюється вона чи змінюється з часом. Наприклад, більшість продуктів показують онбординг лише за першому вході у систему. Існуючі користувачі, знайомі з продуктом, не знаходять у цьому жодної реальної користі. Режими змінюють те, як зазвичай функціонує мікровзаємодія, наприклад, зміна розташування в погодній програмі або встановлення телефону на режим «Не турбувати».

3.2.3 Наукове пояснення мікровзаємодій

У науковому дискурсі мікровзаємодії розглядаються як частина поведінкових патернів інтерфейсу, що поєднують функціональність, анімацію та емоційне реагування. Їх аналіз вимагає міждисциплінарного підходу – з урахуванням когнітивної психології, ергономіки, теорії дизайну інтерфейсів та принципів доступності. З погляду когнітивної науки, мікровзаємодії сприяють створенню ефективного зворотного зв’язку, формують очікування, забезпечують відчуття завершеності та задоволення. Їх ефективність обумовлюється не лише візуальним оформленням чи технічною реалізацією, а й тим, наскільки точно вони відповідають природним схемам обробки інформації в мозку користувача.

Розглянуті нижче шість когнітивних та нейропсихологічних принципів пояснюють, чому мікровзаємодії мають такий значний вплив на

користувацький досвід, і як їх використання здатне підвищити ефективність, привабливість і впізнаваність цифрового продукту.

1. Схильність мозку до зворотного зв'язку.

Люди за своєю природою схильні до пошуку зворотного зв'язку. З дитинства ми вчимося, взаємодіючи з навколишнім середовищем і отримуючи відповіді. Цей процес допомагає нашому мозку зрозуміти причини й наслідки наших дій. Коли користувач виконує дію, наприклад, натискає кнопку, що призводить до реакції, це створює цикл дії та зворотного зв'язку. Це надає не лише впевненість, але й відчуття контролю.

2. Принцип задоволення та дофамінові винагороди.

Система винагород мозку, особливо вивільнення дофаміну, відіграє важливу роль у керуванні нашою поведінкою. Приємний досвід або навіть його очікування може викликати сплеск дофаміну. Гарна мікровзаємодія, наприклад, грайлива анімація після завершення дії, може слугувати невеликою винагородою. Це може спонукати користувачів більше взаємодіяти з вашим інтерфейсом і шукати такі мікромоменти.

3. Ефект Зейгарнік.

Згідно з ефектом Зейгарнік, люди краще запам'ятовують незавершені завдання. Наш мозок схильний прагнути до завершення. Надаючи зворотний зв'язок про хід або завершення завдання – наприклад, панель завантаження або анімацію чек-листа – мікровзаємодії задовольняють це когнітивне бажання, пропонуючи розумову розв'язку і задоволення.

4. Прагнення до послідовності.

Наш мозок постійно робить прогнози на основі попереднього досвіду. Коли інтерфейс працює послідовно, підтверджуючи наші прогнози, він зменшує когнітивне навантаження і підвищує задоволеність користувачів. Повторювані послідовні мікровзаємодії, наприклад, однакова анімація для певної дії, допомагають визначити очікування користувача, роблячи досвід більш інтуїтивно зрозумілим.

5. Естетично-практичний ефект.

Дослідження NN/g демонструють, що користувачі сприймають естетично привабливий дизайн як зручніший, навіть якщо він не є функціонально кращим. Це означає, що добре продумана мікровзаємодія не тільки додає функціональної цінності, але й підвищує загальну естетичну привабливість інтерфейсу. А вона, в свою чергу, може покращити сприйняття зручності використання та завоювати довіру користувачів.

6. Посилення уваги.

Наш мозок налаштований на те, щоб помічати й запам’ятовувати характерні деталі, особливо ті, що виділяються. Унікальна мікровзаємодія може привернути увагу користувача, підкресливши важливі особливості або спрямувавши її у потрібному напрямі.

РОЗДІЛ 4. ОЦІНКА ЯКОСТІ ІНТЕРФЕЙСУ

4.1 Поняття та показники Usability

Usability зазвичай визначається як міра того, наскільки ефективно, результативно та задовільно користувач може досягати поставлених цілей при використанні певного продукту чи системи. Одне з класичних визначень подано в стандарті ISO 9241-11, згідно з яким usability – це: «Ступінь, до якого продукт може бути використаний певними користувачами для досягнення визначених цілей з ефективністю, результативністю і задоволенням у певному контексті використання».

Таким чином, поняття usability охоплює не лише технічні параметри інтерфейсу, а й враховує специфіку користувача, його завдання, а також середовище, в якому відбувається взаємодія.

Usability є міждисциплінарним поняттям, яке застосовується в галузях програмної інженерії, дизайну, психології, когнітивної науки та ергономіки. У дизайні інтерфейсів воно безпосередньо пов’язане з досвідом користувача (User Experience, UX) і слугує інструментом оптимізації системи під реальні потреби людей.

Контекст застосування також суттєво впливає на інтерпретацію usability. Для складних професійних систем (наприклад, медичного обладнання чи авіаційного ПЗ) пріоритетами є точність, надійність і мінімізація помилок. Для мобільних застосунків – швидкість виконання завдань, простота інтерфейсу та приємність у користуванні.

У структурі usability зазвичай виокремлюють кілька ключових компонентів:

– ефективність (Effectiveness) – здатність користувача досягати поставлених цілей з використанням системи;

– продуктивність (Efficiency) – кількість ресурсів (час, зусилля), необхідних для досягнення цілей;

– задоволеність (Satisfaction) – суб'єктивне враження користувача від процесу взаємодії з продуктом;

– легкість у навчанні (Learnability) – наскільки швидко новий користувач може освоїти систему;

– запам'ятовуваність (Memorability) – здатність користувача відновити навички після перерви в роботі з продуктом;

– частота та тяжкість помилок (Errors) – кількість допущених помилок, їх серйозність і легкість виправлення.

Ці характеристики можуть варіюватися залежно від типу системи та контексту її використання, однак в комплексі вони забезпечують повноцінну оцінку юзабіліті.

Показники usability – це кількісні або якісні метрики, що дають змогу оцінити рівень зручності використання системи. Основними серед них є:

– час на виконання завдання: середній час, який користувач витрачає на виконання конкретного завдання;

– кількість правильних/успішних дій: частка користувачів, які змогли досягти цілі без помилок;

– частота помилок: кількість неправильних дій або помилок, допущених під час виконання завдань;

– кількість звернень до допомоги: кількість випадків, коли користувач звертався до довідки, підказок або зовнішньої допомоги;

– рівень задоволеності: вимірюється через опитування або шкали, такі як SUS (System Usability Scale);

– показник відмови (Drop-off rate): частка користувачів, які припинили виконання завдання або покинули інтерфейс.

Ці показники можуть бути отримані як під час лабораторних тестувань (usability testing), так і через аналітичні інструменти в реальному середовищі (наприклад, веб-аналітика).

Існує кілька основних підходів до оцінки usability:

– тестування з користувачами – залучення реальних або потенційних користувачів до виконання типових завдань, дослідник фіксує поведінку, час виконання завдань, помилки та коментарі;

– евристичний аналіз – експерти з юзабіліті аналізують інтерфейс на відповідність встановленим евристикам (наприклад, евристики Нільсена);

– опитування та анкетування – збирання суб'єктивних оцінок користувачів, зокрема за шкалою SUS, QUIS або CSUQ;

– аналіз логів і поведінкових даних – автоматичний збір статистики з вебінтерфейсів або програмних систем.

Комбінація кількох методів дозволяє отримати повнішу картину якості взаємодії користувача із системою.

4.2 Евристики usability Якоба Нільсена

У 1990-х роках Якоб Нільсен, консультант з веб-юзабіліті та партнер у Nielsen Norman Group, і Рольф Моліч, експерт з юзабіліті, розробили список із десяти інструкцій щодо дизайну інтерфейсу користувача. Вони називаються евристичними, тому що це широкі емпіричні правила, а не конкретні вказівки щодо зручності використання дизайну взаємодії. Зараз ці евристики так само актуальні, як і тоді. Деякі з найвпливовіших компаній, як-от Apple і Google, застосовують ці евристики у своїх продуктах.

Нижче наведено перелік евристик, роз'яснення до них та приклади деяких UX-UI рішень, що демонструють відповідність цим евристичним критеріям.

ВАЖЛИВО!

Наведені приклади UX-UI, що демонструють застосування евристик, ні в якому разі не є вичерпними! В кожному конкретному застосунку можуть бути застосовані інші реалізації, які визначаються специфікою застосунку.

1. Видимість стану системи. Система повинна завжди інформувати користувача про те, що відбувається, за допомогою відповідного зворотного зв’язку в розумний проміжок часу.

Приклади UX-UI рішень:

- індикатор завантаження під час завантаження сторінки або виконання дії;
- прогрес-бар із часом, що залишився, щоб користувач знав, коли завершиться процес;
- анімація «завантаження» при переході між сторінками;
- виділення кольором, жирним шрифтом чи іншим способом поточного елементу меню/сторінки;
- позначка кількості товарів в кошику біля піктограми кошику;
- різний вигляд піктограм для обраних/необраних (виділених/не виділених) елементів – наприклад, пустий/не пустий кошик, пустий/не пустий список улюбленого тощо.

2. Відповідність між системою та реальним світом. Система повинна «говорити» мовою користувача, використовуючи знайомі слова, фрази, зображення та концепції, а не технічний жаргон.

Приклади UX-UI рішень:

- використання реальних, знайомих користувачеві, схематичних зображень для позначення об’єктів програми, наприклад, для позначення контейнера видалених об’єктів використовується картинка сміттєвого кошика, кошик в інтернет-магазині має вигляд реальної валізки в супермаркеті або сумки, додавання в улюблене позначається значком сердечка, операція додавання до порівняння має вигляд ваг, можливість дзвінка оператору позначена трубкою телефону тощо;

- фільтри в інтернет-магазинах мають назви, що відповідають конкретним реальним характеристикам товарів («Розмір», «Колір», «Бренд» тощо), а не називаються технічними термінами, які можуть бути незрозумілими користувачам;
- використання символів, зрозумілих у певній культурі (наприклад, $ для позначення валюти, піктограми соціальних мереж та месенджерів для позначення переходу у відповідний застосунок тощо);
- графічне зображення товару (наприклад, фото меблів замість текстового опису).

3. Довільне керування і свобода дій користувача. Користувачі повинні мати можливість легко відміняти дії або повертатися до попереднього стану чи повторно виконувати попередні дії.

Приклади UX-UI рішень:

- кнопка «Скасувати» або «Назад» у формі;
- кнопка «Повторити»;
- кнопка «Назад» для повернення до попередньої сторінки;
- кнопка прямого переходу на головну сторінку сайту;
- «хлібні крихти» для переходу на будь-яку сторінку з попереднього шляху;
- відсутність блокування користувача в певному процесі (наприклад, можливість залишити сторінку оформлення покупки та продовжити пошук інших товарів);
- можливість редагувати дані перед остаточним підтвердженням дії.

4. Цілісність і стандарти. Стиль дизайну, кольорові палітри, гарнітури шрифтів, термінологія, порядок виконання задач повинні бути однаковими незалежно від платформи чи місцеположення користувача в застосунку.

Користувачі не повинні задаватися питанням, чи означають різні слова або дії одне й те саме.

Приклади UX-UI рішень:

- кнопка «Додати до кошику» завжди має однаковий вигляд на всіх сторінках сайту;
- на всіх екранах мобільного додатку іконка «Назад» розташована в одному й тому ж місці – у верхньому лівому куті;
- постійне використання однакових шрифтів для заголовків;
- постійне використання однакових шрифтів для текстових описів;
- однаковий формат повідомлень про помилки (наприклад, червоний текст, іконка попередження);
- однакова логіка фільтрів у всіх розділах сайту;
- розташування головного меню веб-застосунку у верхній частині сторінки;
- стандартизовані елементи навігації в мобільному застосунку.

5. Запобігання помилкам. Система повинна попереджати користувачів про можливі помилки **перед** їх появою. Продуманий дизайн користувальницького інтерфейсу, що запобігає появі помилок користувача завжди краще добре продуманих повідомлень про помилки. При проектуванні інтерфейсу необхідно або повністю усунути елементи, в яких можуть виникати помилки користувача, або повідомляти йому про потенційно можливе виникнення проблеми при введенні даних в цих елементах.

Приклади UX-UI рішень:

- попередження типу «Ви впевнені, що хочете *назва операції*?» перед виконанням критичних операцій (наприклад видалення файлу, відміна операції, критичні зміни в даних тощо);

- заборона доступу до елементів UI, поки не виконано всіх необхідних умов чи не введено всі необхідні дані (наприклад, відключення кнопки «Надіслати», поки не заповнені всі обов’язкові поля форми);
- підказки про вимоги до полів введення (наприклад, підказки про мінімальну довжину пароля та необхідні обов’язкові символи в паролі);
- заборона введення недопустимих символів (наприклад, можливість введення тільки цифр в поле номеру телефону);
- перехід на цифрову клавіатуру при введенні числових полів на мобільному пристрої;
- позначення зірочкою чи текстовою підказкою обов’язкових полів, роз’яснення символу зірочки;
- маски введення;
- приклади даних в полях введення;
- заміна полів введення на вибір зі списку можливих значень.

6. Розуміння замість запам’ятовування. Використання цієї евристики передбачає зменшення навантаження на пам'ять користувача, зробивши необхідну інформацію доступною безпосередньо в інтерфейсі.

Приклади UX-UI рішень:

- користувач не повинен запам'ятовувати інформацію під час переходу від одного діалогового вікна до іншого, наприклад, при виконанні складних багатокрокових операцій в скороченому вигляді відображається результат виконання попередніх кроків;
- при заповненні полів введення «знайома» системі інформація про користувача підставляється у відповідні поля автоматично;
- система може запам’ятовувати останні дії користувача та використовувати їх в формах введення чи інших діях (наприклад,

спадаючий список із пропозиціями при введенні тексту в пошуковий рядок, відображення даних останньої адреси доставки, щоб користувач не вводив їх повторно;

– збереження заповненої інформації у формі при випадковому оновленні сторінки.

7. Гнучкість та ефективність використання. Інтерфейс повинен підтримувати як новачків, так і досвідчених користувачів.

Приклади UX-UI рішень:

– гарячі клавіші для швидкого виконання завдань для досвідчених користувачів (наприклад, Ctrl+Z для скасування);

– наявність різних режимів роботи системи (наприклад, у графічному редакторі є як базовий режим для новачків, так і розширений із більш складними інструментами);

– можливість налаштування панелі інструментів під потреби користувача;

– збереження попередніх дій як шаблонів для повторного використання (наприклад, через запис макросів);

– опція «Зберегти як шаблон» у складних формах.

8. Естетичний і мінімально необхідний дизайн. Інтерфейс повинен бути зрозумілим, без зайвих елементів, які відволікають від виконання основного завдання. Кожен інтерфейсний елемент, що містить марну інформацію, грає роль інформаційного шуму і відволікає користувача від дійсно корисних інтерфейсних елементів. При оцінці мінімально необхідного дизайну треба враховувати наявність елементів, що не корелюються з метою використання конкретної сторінки – чим їх більше, тим гірше дотримано евристику. Типовим прикладом ідеального мінімалістичного UX-UI рішення при виконанні якоїсь дії є «зробити в один клік». При оцінці естетичності варто оцінити дотримання показників, які

стосуються загальних принципів проєктування інтерфейсів, як от: гаманець Мілера, принцип угруповання, принцип бритви Оками, принцип розумного запозичення, принцип «видимість відображає корисність» (детальніше див. в лекційному матеріалі).

9. Допомога користувачам в розпізнаванні, діагностиці та усуненні помилок. Система повинна надавати чіткі повідомлення про помилки та способи їх вирішення.

Приклади UX-UI рішень:

- повідомлення про помилки повинні бути написані природньою мовою, а не замінюватися кодами помилок;
- іконка попередження поруч із помилково заповненим полем;
- список поширених запитань із розділом «Що робити, якщо...»;
- пропозиція дій для виправлення, наприклад «Оновіть номер картки або виберіть інший спосіб оплати».

10. Довідка та документація. Система повинна надавати необхідну документацію/засоби, якщо користувачам потрібна допомога при виконанні завдання. Ця допомога повинна бути представлена в такому форматі, щоб її легко було знайти, вона відповідала поставленим завданням і була сформульована таким чином, щоб допомогти користувачам виконати необхідні кроки для вирішення проблеми, з якою вони стикаються. Не зважаючи на те, що в ідеальному випадку краще, коли системою можна користуватися без документації, вона все одно необхідна – як у вигляді системи допомоги, так і, можливо, навіть, у вигляді друкованого керівництва (особливо актуально для систем «виробничого» використання).

Приклади UX-UI рішень:

- іконка «?» поруч із кожним складним елементом форми;
- підказки з прикладами у формах заповнення, які пояснюють, як правильно вводити дані;

- розділ «Допомога» в меню навігації;
- розділ поширених запитань;
- спливаючі підказки з поясненням функції кнопок;
- відеоінструкції для початку роботи з продуктом чи для виконання складних задач;
- інтерактивний підручник, що демонструє основні функції програми;
- чат із підтримкою для вирішення проблем у режимі реального часу.

ГЛОСАРІЙ

Accessibility – доступність інтерфейсу для людей з обмеженими можливостями (зір, слух, моторика тощо).

Call to Action (CTA) – елемент, який закликає користувача до дії.

Design System – уніфікований набір компонентів, стилів і принципів, який забезпечує цілісність дизайну продукту.

Empathy Map – карта емпатії; візуальна модель, яка допомагає зрозуміти, що користувач думає, відчуває, бачить, чує, робить.

Figma – популярний онлайн-інструмент для створення UI-дизайну та прототипів.

Flow – послідовність дій, які виконує користувач для досягнення мети.

Grid – сітка, яка використовується для впорядкування елементів інтерфейсу.

Mockup – візуально деталізоване зображення інтерфейсу з кольорами, шрифтами, іконками, без інтерактивності.

Persona – узагальнений профіль типового користувача з конкретними цілями, болями, контекстом і поведінкою.

Task Flow – лінійна послідовність дій, які виконує користувач для досягнення однієї мети.

UI (User Interface) – всі компоненти інтерактивної системи (програмне або апаратне забезпечення), які надають користувачу інформацію та елементи керування для виконання певних завдань за допомогою інтерактивної системи.

UI Components – елементи інтерфейсу.

UI Kit – набір стандартних компонентів інтерфейсу для повторного використання в межах проєкту.

Usability – зручність у використанні; характеристика, що описує, наскільки легко й ефективно користувач може взаємодіяти з інтерфейсом.

User Flow – діаграма, що показує всі можливі маршрути користувача в межах системи для досягнення певної мети (з варіантами та розгалуженнями).

UX (User eXperience) - сприйняття та відповіді людини, що є наслідком використання та/або очікуваного використання продукту, системи або служби.

Wireframe – схематичне зображення структури сторінки або екрану, що показує розташування основних елементів без деталізації дизайну.

Больові точки – проблеми або труднощі, з якими стикається користувач під час взаємодії з продуктом.

Когнітивне навантаження – обсяг зусиль, які користувач витрачає на сприйняття, розуміння та використання інтерфейсу.

Мікровзаємодія – невелика анімована реакція інтерфейсу на дію користувача.

Прототип – спрощена модель інтерфейсу, що використовується для перевірки логіки, структури та взаємодії.

Сценарій користувача – опис конкретної ситуації використання продукту користувачем з урахуванням контексту, мети та мотивації.

РЕКОМЕНДОВАНІ БЕЗКОШТОВНІ КУРСИ З РОЗРОБКИ UX-UI

1. Prometheus: Основи Web UI розробки 2023. https://prometheus.org.ua/course/course-v1:LITS+114+2022_T2?authuser=0

   Зауваження: даний курс присвячено веб-програмуванню, але він містить окремі теми, що охоплюють аспекти UI-розробки.
2. Навчальний курс Дія.Освіта: Інклюзивний вебдизайн. https://osvita.diia.gov.ua/courses/inkluzivnij-vebdizajn?authuser=0
3. Курс Udemi (укр.мовою): UI Kit та механіка роботи в Figma. https://www.udemy.com/course/ui-kit-figma/?authuser=0
4. Безкоштовні курси Udemy з Figma (англ.мовою):
   - Learn Figma for UI UX Design (with a Design Project) https://www.udemy.com/course/learn-figma-ui-ux-design-project/?authuser=0
   - Figma UI UX designing course. https://www.udemy.com/course/figma-ui-ux-designing-course/?authuser=0
   - Figma Introduction Course https://www.udemy.com/course/codedfigmacourse/?authuser=0
   - Learn Figma for UI UX Design Within 1 Hour In 2023 https://www.udemy.com/course/figma-ui-ux-design-fundamental-course/?authuser=0
   - Learning Figma in 1 hour https://www.udemy.com/course/learning-figma-in-1-hour/?authuser=0
   - From idea to MVP without coding | Intro to Figma & Bravo https://www.udemy.com/course/idea2mvp/?authuser=0

- Learn How to Design a Website in Figma https://www.udemy.com/course/learn-how-to-design-a-website-in-figma/?authuser=0

5. Курси від Coursera з можливістю отримати професійний сертифікат Google UX Design: https://www.coursera.org/google-certificates/ux-design-certificate?authuser=0#courses

ЛІТЕРАТУРА

1. Чемерис Г. Ю. UX/UI дизайн : навчальний посібник для здобувачів ступеня вищої освіти бакалавра спеціальності «Дизайн» освітньо-професійної програми «Графічний дизайн». Запоріжжя : ЗНУ, 2021. 290 с.
2. Theo Mandel. The Elements of User Interface Design.
3. International Journal of Human–Computer Interaction. https://www.tandfonline.com/toc/hihc20/current
4. Android UI design guides https://developer.android.com/design/ui/mobile/guides/foundations/accessibility?authuser=0
5. Human Interface Guidelines (Apple platform) https://developer.apple.com/design/human-interface-guidelines?authuser=0
6. Flat-Design Best Practices. Nielsen Norman Group. https://www.nngroup.com/articles/flat-design-best-practices/
7. Material Design https://m3.material.io/?authuser=0
8. Настанови з доступності вебвмісту (WCAG) https://www.w3.org/Translations/WCAG21-ua/
9. Національна стратегія зі створення безбар'єрного простору в Україні до 2030 року. https://zakon.rada.gov.ua/laws/show/366-2021-%D1%80#n10
10. Chemerys, Hanna & Demirbilek, Muhammet & Briantseva, Hanna & Sharov, Sergii & Podplota, Svitlana. (2022). Fundamentals of UX/UI design in professional preparation of the future bachelor of computer science. AIP Conference Proceedings. 2453. 030025. 10.1063/5.0094433.
11. Chemerys, Hanna & Briantseva, Hanna. (2021). Актуальність

упровадження проєктування універсального та доступного дизайну у професійну підготовку майбутніх дизайнерів. Pedagogy of the formation of a creative person in higher and secondary schools. 3. 151-155. 10.32840/1992-5786.2021.76-3.27.

12. Rajesh, P & Selvadurai, M & Selvamani, V & Chandrasekar, P. (2022). Fundamentals of UX/UI (An Approach to Design Principles). 10.47715/JPC.B.87.2022.9789391303389. Sulianta, Feri. (2025). UX UI Design. 350 p.
13. Jain, Nilakshi. (2023). UI Design : Key to Captivate User Understanding. 173 p.
14. Tripathy, Abhijit & Nanda, Sai Soumyak & Gupta, Khusbu & Mishra, Priyanka & Monisha, Chippada. (2022). Demystifying UI/UX: A client's guide on understanding User Interfaces and User Experience.
15. Stull E. UX Fundamentals for Non-UX Professionals: User Experience Principles for Managers, Writers, Designers, and Developers. Apress, 2018. 331 p.
16. Tomlin Craig W. UX Optimization: Combining Behavioral UX and Usability Testing Data to Optimize Websites. Apress, 2018. 198 p.
17. Canziba Elvis. Hands-On UX Design for Developers. Packt Publishing, 2018. 350 p.
18. Lull, D. (2017). UX Psychology Basics. In: Discussions in User Experience . Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4842-3267-5_4
19. Від користувацьких інтерв'ю до роботи мозку. Книжки для дизайнерів інтерфейсів. https://prjctr.com/mag/interface-books
20. Золотухіна О.А., Ільїн О.Ю., Горячев Т.В. Автоматизована обробка спеціалізованих запитів користувачів. Наукові праці ДонНТУ, Серія “Інформатика, кібернетика та обчислювальна техніка”, №1, 2024. С.44-50.

21. Methods of Biometric Authentication for Person Identification/ Daria Polunina, Oksana Zolotukhina, Olena Nehodenko, and Iryna Yarosh. 2nd International Congress of Electrical and Computer Engineering (ICECENG'23), November 22 to 25, 2023, Bandirma, Turkey. P.327-339. https://doi.org/10.1007/978-3-031-52760-9_23.

Електронне навчальне видання

Золотухіна Оксана Анатоліївна

Худік Богдан Олександрович

UX/UI ДИЗАЙН. ПРОЄКТУВАННЯ ТА РОЗРОБКА ІНТЕРФЕЙСУ КОРИСТУВАЧА

Навчальний посібник