**Форма № Н-9.02**

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ**
**імені адмірала Макарова**

Херсонська філія

Кафедра інформаційних технологій та фізико-математичних дисциплін

"Допущений до захисту"
Завідувач кафедри
д.т.н., доцент Гучек П.Й. "____"
_____ 2020 р.

# ПОЯСНЮВАЛЬНА ЗАПИСКА

***до кваліфікаційної роботи***

**на здобуття ступеня вищої освіти "магістр"**

на тему: «Дослідження методів розробки Web–додатків для мультиплеєрних онлайн–ігор»

Виконав: студент VI курсу, групи 6157м спеціальності
122 - "Комп'ютерні науки", ОПП –
(шифр і назва спеціальності та освітньо-професійної програми)
"Інформаційні управляючі системи та технології"
Ковальов А.М.
(прізвище та ініціали)

Керівник к.т.н., Гайдаєнко О.В.
(прізвище та ініціали)

Рецензент
(прізвище та ініціали)

30.12 – 2020 року

.

Анотація

Дипломна робота створена для автоматизації ігрового процесу у веб–додатку.

При розробці додатку були проаналізовані аналогічні ігрові додатки та реалізовано програмне забезпечення, яке надає змогу:

- Авторизуватися у грі
- Створювати ігрову кімнату, нову гру
- Заходити у існуючу гру
- Обирати та встановлювати кількість балів для перемоги
- Вести переписку у чаті
- Слідкувати за поточними балами гравців
- Здійснювати ігровий процес

Робота викладена українською мовою і виконана на 91 аркушах, містить 20 рисунків, 2 таблиць, 4 додатка. Список використаних джерел містить 7 найменувань.

Summary

The thesis is designed to automate the gameplay in a web application.

During the development of the application, similar game applications were analyzed and software was implemented, which allows:

• Log in to the game

• Create a game room, a new game

• Enter an existing game

• Choose and set the number of points to win

• Communicate in chat

• Monitor current player scores

• Play

Work outlined in the Ukrainian language and executed on 91 sheets, contains 20 figures, tables 2, 4 appendix. The list of the used sources contains 7 of the names.

Зміст

# ВСТУП

На сьогоднішній день ігри це один з найпопулярніших способів провести дозвілля. Існує велика кількість видів ігор, але найпопулярніший та найстаріший вид ігор – це браузерна гра або онлайн–гра. Браузерна відеогра –це різновид відеоігор, основною характеристикою якого є використання інтерфейсу веб-браузера. Такі ігри представлені різноманітними жанрами і, як правило, не вимагають встановлення іншого програмного забезпечення, окрім самого браузера.

Переважна більшість браузерних ігор орієнтовані на бізнес модель Free-to-play, тобто, дозволяють грати безкоштовно.

Актуальність обраної для дослідження теми полягає в тому, що розробка онлайн–гри “Dixit” дозволить вивчити ринок ігрової індустрії, його різноманітність, конкурентоспроможність, специфічність використання та сучасні технолоії розробки та реалізацію браузерних ігор.

# 1.ДОСЛІДЖЕННЯ МЕТОДІВ ЗАСТОСУВАННЯ ВЕБ–ДОДАТКУ У РОЗРОБЦІ ОНЛАЙН–ІГОР

## 1.1 Загальний опис веб–додатків

Вебсайт – сукупність вебсторінок та залежного вмісту, доступних у мережі Інтернет, які об'єднані як за змістом, так і за навігацією під єдиним доменним ім'ям. Фізично сайт може розміщуватися як на одному, так і на кількох серверах.

Веб–додаток – це будь–яка комп'ютерна програма, яка виконує певну функцію, використовуючи веб–браузер у якості свого клієнта. Додаток може бути таким же простим, як дошка оголошень або контактна форма на веб–сайті, або настільки ж складний, як текстовий процесор або багатокористувацький мобільний ігровий додаток, який ви завантажуєте на телефон.

Сайт це в першу чергу щось інформаційне і статичне: візитка компанії, сайт рецептів, міський портал або вікіпедія. Набір підготовлених заздалегідь HTML–файлів, які лежать на віддаленому сервері і віддаються браузеру за запитом. Сайти містять різну статику, яка як і HTML–файл не генерується на льоту. Найчастіше це картинки, CSS–файли, JS–скрипти, але можуть бути і будь–які інші файли: mp3, mov, csv, pdf.

А веб–додаток – це щось технічно більш складне. Тут HTML–сторінки генеруються на льоту в залежності від запиту користувача. Поштові клієнти, соцмережі, пошукові системи, інтернет–магазини, онлайн–програми для бізнесу та роботи чи ігор, це все веб–додатки.

Суттєва перевага побудови веб–додатків при підтримці стандартних функцій браузера полягає в тому, що функції повинні виконуватися незалежно від операційної системи даного клієнта.

## 1.2 Структура веб–додатку

Веб–додатки можна розділити на кілька типів, в залежності від різних поєднань їх основних складових:

1. Backend (бекенд або серверна частина програми) працює на віддаленому комп'ютері, який може знаходитися де завгодно. Вона може бути написана на різних мовах програмування: PHP, Python, Ruby, C # та інших. Якщо створювати додаток використовуючи тільки серверну частину, то в результаті будь–яких переходів між розділами, відправок форм, оновлення даних, сервером буде генеруватися новий HTML–файл і сторінка в браузері перезавантажуватиметься.
2. Frontend (фронтенд або клієнтська частина програми) виконується в браузері користувача. Ця частина написана на мові програмування Javascript. Додаток може складатися тільки з клієнтської частини, а то й потрібно зберігати дані користувача довше однієї сесії. Це можуть бути, наприклад, фоторедактори або прості іграшки.
3. Single page application (SPA або односторінкове додаток). Більш цікавий варіант, коли використовуються і бекенд і фронтенд. За допомогою їх взаємодії можна створити додаток, яке буде працювати зовсім без перезавантажень сторінки в браузері. Або в спрощеному варіанті, коли переходи між розділами викликають перезавантаження, але будь–які дії в розділі обходяться без них.

При розробці веб–додатку потрібно ретельно розробити його візуальну та функціональну частини, оскільки він повинен бути зрозумілим для користувачів, а також швидко завантажуватися з будь–яких пристроїв та браузерів.

### 1.2.1Функціональні можливості веб–додатку

Веб–додатки існують абсолютно різного призначення, тому їх можливості використання залежать від конкретної сфери для якої вони написані.

Усі веб–додатки являють собой інструмент для досягнення певних цілей. Наприклад: веб–додаток для бронювання номерів готелу дозволяє користувачам шукати та фільтрувати необхідку інформацію та обирати варіанти, тобто працювати з нею; банківський веб–додаток надає своїм клієнтам весь необхідний фунціонал для виконання грошових операцій. А ігровий веб–додаток надає користувачам можливість гарно проводити дозвілля, тобто грату у реалізовану в ньому гру.

До функціональних відносяться:

1. управління контентом веб–додатку ,які доступні даному додатку. Веб–додаток повинен надавати можливості завантаження контенту на сервер (uploading), класифікації, отримання з сервера (downloading);
2. пошук інформації в контенті додатку (в документах, файлах даних, базах даних);
3. підтримка взаємодії користувачів (форуми, обмін повідомленнями, рецензування документів);
4. більш доступного режиму (розрахунки відповідно до бізнес–логікою, обробка зображень, обробка документів і т. Рр).

Базова функціональність багато в чому реалізується середовищем виконання веб–додатку і його спеціалізацією.

Потрібно виділити декілька важливих критерів при розвробці веб–додатку:

1. Надійність. Додаток має працювати з заданими характеристиками і встановленою швидкістю незалежно від кількості користувачів.
2. Швидкодію. У будь–яких умовах середній час обробки запиту системою не повинно перевищувати заданих параметрів.
3. Безпека. Включає рівні прав доступу, авторизацію і аутентифікацію.
4. Масштабованість. Якщо в майбутньому буде прийнято рішення додати компоненти, система повинні бути здатна збільшити поізводітельность з урахуванням нових умов.

5. Вартість покупки. Фінансова позиція щодо розробки безпосередньо впливає на якість розробки, адже безкоштовні додатки частіше за все обмежені за своїм функціоналом, що обумовлено низькою спроможність довго утримувати програмістів. Адже якщо розробка не приносить коштів, то вона і не має фінансової підтримки, яка дає можливість залучати висококваліфікові кадри. Проте не завжди платні додатки є гарно розробленими, велика кількість з них націлена на швидку розробку та інтеграцію з ціллю принести найбільшу вигоду у короткий проміжок часу, і надалі не відбувається підтримки та розвитку продукту.

«Важливо визначити баланс між бажанням залучити більше користувачів з бажанням отримати гроші, адже це прямо впливає на реалізацію функціоналу. Деякі способи монетизації дозволяють почати заробляти відразу ж після релізу додатки, в той час як інші моделі забезпечують поступовий набір користувальницької аудиторії з монетизацією через певний період.

Краща модель монетизації веб–додатку виявляється така, яка що перш за все орієнтується на користувачів, на комфорт користуванням додатком, та добровільною оплатою функціоналу. Цей спосіб вирішує низку проблем серед яких, деякі браузери не справляються з навантаженням скриптів на сторінці залежно від платформи, багато клієнтів не видержують напливу реклами і покидають додаток після отримання бажаного результату, або намагаються уникнути показу реклами різними способами, і після використання ніколи не розкажуть про цей додаток нікому задля поширення кругу користувачів, оптимізація швидкості роботи додатку теж залежить від розрахунку скриптів показу реклами та збору аналітичних даних для обробки даних.

Це можна вирішити якщо користувач отримує весь функціонал, який прийнятий допомогти вирішити певну проблему або досягти цілі, без регулярного показу реклами, тим самим оптимізується швидкість роботи додатку, користувач задоволений, може розповсюджувати інформацію про додаток іншим людям, поширюючи круг користувачів, тим самим популярність компанії через їх продукт зростає» [1].

### 1.2.2 Архітектура веб–додатку

На сьогоднішній день використовуються чотири технології для реалізації веб–сервісів:

1. TCP/IP – стек протоколів мережі Інтернет, який розуміється практично з будь–яким мережним устаткуванням, від мейнфреймів до портативних пристроїв і PDA. Стек протоколів TCP/IP ділиться на 4 рівні: прикладний, транспортний, міжмережевий та рівень доступу до середовища передачі. Терміни, що використовуються для позначення блоку переданих даних, різні при використанні різних протоколів транспортного рівня: TCP і UDP.
2. HTML–універсальна мова гіпертекстової розмітки, яка використовується для демонстрації контенту пристроями споживачів HTML може вбудовувати програми, написані на мові сценаріїв, наприклад JavaScript, що впливає на поведінку та вміст веб–сторінок. Включення CSS визначає вигляд і компонування вмісту.
3. XML–універсальний засіб для обробки всіх різновидів даних. На його базі можуть працювати і інші протоколи обміну інформацією: SOAP і WSDL. Специфікація XML описує XML–документи і частково описує поведінку XML–процесорів (програм, які читають XML–документи і забезпечують доступ до їх вмісту). XML розроблявся як мова з простим формальним синтаксисом, зручний для створення і обробки документів програмами і одночасно зручний для читання і створення документів людиною, з підкресленням націленості на використання в Інтернеті.
4. UDDI– платформово–незалежний інструмент для розміщення описів веб–сервісів (WSDL) для забезпечення можливості їх пошуку іншими організаціями і інтеграції в свої системи.

## 1.3 Аналіз існуючих рішень розробки ігрових веб–додатків

На сьогоднішній день існує бізліч ігрових веб–додатків реалізованих завдяки сучасним веб–технологіям. Проте у багатьох з них існують суттєві недоліки, які і спонукали до розробки власної онлайн–гри.

Наприклад онлайн–гра «Імаджинаріум», аналог якої буде розроблятися, має перелік недоліків, які буде усунено у результаті розробки. Головні недоліки реалізації:

1. погана швидкодія– гра досить довго обробляє запити гравців;
2. зависання у процесі гри;
3. маленька бібліотека ігрових карток;
4. погано реалізований розрахунок балів.

У ході аналізу веб–гри «Імаджинаріум» стає зрозумілим, що основою всіх недоліків є погано реалізований функціонал.

Онлайн–гра «Воображариум» трохи відрізняється ігровим процесом, але також має схожі недоліки, такі як:

1. гравець відлючається від гри у процесі гри– пробелеми зі звязком із сервером;
2. зависання у процесі гри;
3. відсутність можливості спілкуавання у додатку.

У таблиці 1.1 представлена порівняльна характеристика функціоналу існуючих додатків.

Таблиця 1.1 – Порівняльна характеристика функціоналу оналайн–ігор

| | Назва гри/ Фунціональність | «Імаджинаріум» | «Воображариум» |
|---|---|---|---|
| 1 | Швидкодія | – | ✓ |
| 2 | Комунікація | ✓ | – |
| 3 | Профіль гравця | ✓ | – |
| 4 | Прорахунок балів | – | ✓ |

| | | | |
|---|---|---|---|
| 5 | Безкоштовність | ✓ | ✓ |

Проаналізувавши таблицю можна відмітити, що кожна з розглянутих онлайн–ігор має суттєві недоліки, що виникли в результаті загальної недопрацьованності додатків. Це можна легко пояснити, адже безкоштовні додатки часто розробляються в умовах обмежених часом і можливостями розробників. Безкоштовний додаток на відміну від платного ніхто не розрбляє з дотриманням всіх правил розробки, тому часто бувають допущені помилки у розробці або неповноцінно налаштована його робота.

Тому було вирішено розробити безкоштовний додаток з невеликим, але повністю реалізованим під основну ідею функціоналом.

## 1.5 Постановка задачі та опис діяльності веб–додатку мультиплеєрної онлайн–гри «Dixit»

Задачею даної роботи є реалізація власної мультиплеєрної онлайн–гри “Dixit”. Проаналізувавши уже готові рішення та способи їх реалізації, був зроблений висновок, що актуальною буде розробка безкоштовної мультиплеєрної онлайн–гри з невеликою кількістю функціоналу та зручним інтерфейсом.

Об'єкт дослідження – мультиплеєрна онлайн–гра «Dixit».

Предмет дослідження – процеси підрахунку балів, авторизації, передачі повідомлення.

Мета дослідження – підвищення ефективності роботи гри, обробки інформації та швидкодії.

Отже, розробка мультиплеєрної онлайн–гри “Dixit” повинна задовольняти наступні вимоги:

1. безкошновність;
2. швидкодія;
3. можливість комунікації гравців безпосередньо у додатку;

4. налагоджена система прорахунку балів;
5. функціональність.

Веб–додаток мультиплеєрної онлайн–гри “Dixit” являтиме собою веб–сайт, на якому будуть розміщені сторінки з певним вмістом, вмістом виступатимуть форми, які будуть заповнюватися користувачем або автоматично в залежності від конкретної. Автоматично буде формуватися вміст форми з картками гравця. Вміст формується кожного разу після того, як гравеці обирає картку, тобто якщо картка–картинка була обрана один раз, то вона була зіграна і після ходу видаляється, а у форму додається інша, кожного ходу заміщується одна картка. Загалом у гравця постійно повинно бути 5 картор–картинок. Користувач буде самостійно формувати вміст форми авторизації та вікна чату.

Також буде автоматизовано процес підрахунку балів.

## 1.6 Вимоги та архітектура веб–додатку для віртуальної організації

### 1.6.1 Дослідження проблем побудови веб–додатку мультиплеєрної онлайн–гри

Існує чимало проблем, які виникають при розробці ігор. Одна з них це погано продумана та розроблена логіка гри, яка найчастіше є основою реалізації.

Не менш важливою є серверна частина гри, адже від неї залежить роботоспроможність системи. Найбільша кількість проблем зв’язана насамперед з поганою пропускною спроможністю системи і погано налагодженою серверною частиною. Користувачі дуже часто втрачають зв’язок з системою в процесі безпосередньої активності в ній.

Також багато залежить від вибору мови для написання онлайн–гри. Кожна мова програмування має переваги, але і часто вагомі недоліки. Деякі мови веб–програмування просто не дозволять реалізувати ідею через відсутність певних

можливостей мови або погану інтеграію. Деякі не мають достатньої супровідної документації, для ознайомення з особливостями методів реалізації.

### 1.6.2 Розробка архітектури для веб–додатку мультиплеєрної онлайн– гри «Dixit»

Для розроблюваного веб–додатку мультиплеєрної онлайн–гри “Dixit”, буде використовуватися архітектура клієнт–сервер. Клієнт–серверна архітектура набула своєї популярності завдяки динамічному розвитку мережі Інтернет та зосередження значної частини інформації в базах даних на серверах.

Клієнт–серверну архітектуру можна означити, як концепцію інформаційної мережі в якій основна частина її ресурсів зосереджена в серверах, обслуговуючих своїх клієнтів. Така архітектура визначає такі типи компонентів:

1. набір серверів, які надають інформацію або інші послуги програмам, які звертаються до них;
2. набір клієнтів, які використовують сервіси, що надаються серверами;
3. мережа, яка забезпечує взаємодію між клієнтами та серверами.

На (Рис. 1.1)представена схема огранізації архітектури клієнт–сервер.

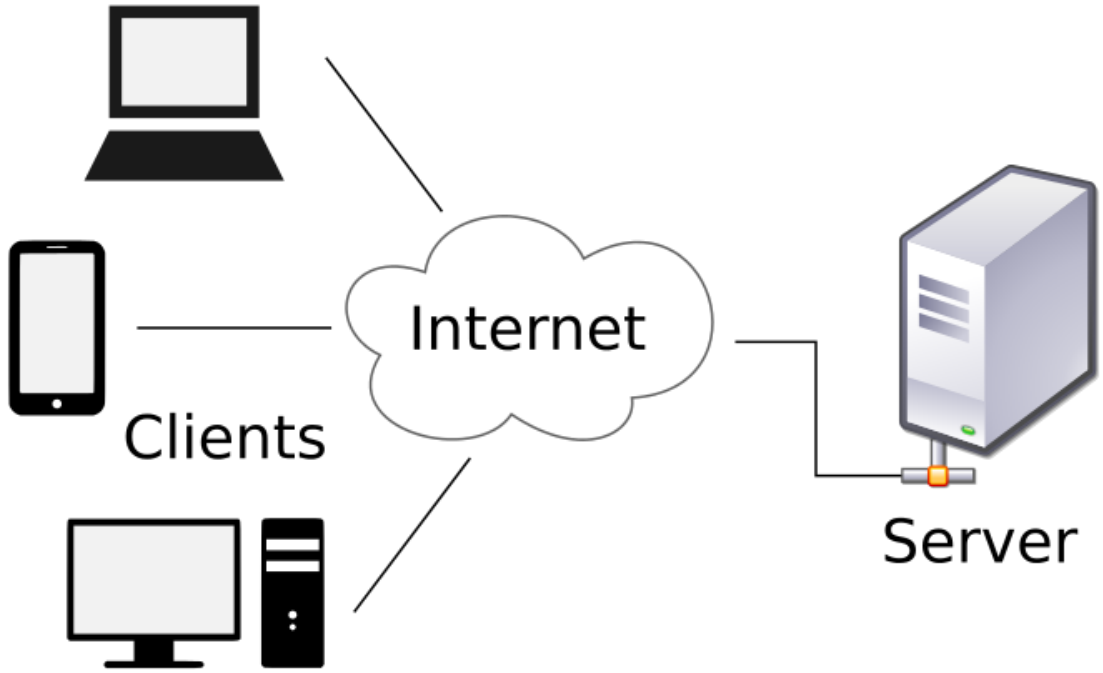



Рисунок 1.1 – схема огранізації архітектури клієнт–сервер

Модель клієнт–серверної взаємодії визначається перш за все розподілом обов'язків між клієнтом та сервером. Логічно можна відокремити три рівні операцій:

1. рівень представлення даних, який по суті являє собою інтерфейс користувача і відповідає за представлення даних користувачеві і введення від нього керуючих команд;
2. прикладний рівень, який реалізує основну логіку застосунку і на якому здійснюється необхідна обробка інформації;
3. рівень управління даними, який забезпечує зберігання даних та доступ до них.

### 1.6.3 Вимоги до якості програмного забезпечення веб–додатку мультиплеєрної онлайн–гри "Dixit"

Якість програмного забезпечення — характеристика програмного забезпечення, ступінь відповідності програмного забезпечення до вимог. При цьому вимоги можуть трактуватись по–різному, що породжує декілька незалежних визначень терміну. Якість програмного забезпечення – набір властивостей продукту (сервісу або програм), що характеризують його здатність задовольнити встановлені або передбачувані потреби користувача. Поняття якості має різні інтерпретації залежно від конкретної програмної системи і вимог до неї.

Розглянемо найбільш важливі вимоги до веб–додатку, наявність яких необхідна для успішного використання користувачами:

1) Кросплатформність.
2) Масштабованість.
3) Продуктивність.
4) Зручність використання.
5) Функціональність.

Висновоки за розділом 1

У даному розділі досліджено методи застосування веб–додатків у розробці онлайн–ігор. Кожна з розглянутих онлайн–ігор має суттєві недоліки, що виникли в результаті загальної недопрацьованності додатків. Під час дослідження були виявлені недоліки в існуючих розробках та поставлена мета, щодо їх уникнення.

# 2. ПРОЕКТУВАННЯ РІШЕНЬ ЩОДО ВЕБ–ДОДАТКУ МУЛЬТИПЛЕЄРНОЇ ОНЛАЙН–ГРИ «DIXIT»

## 2.1 Побудова моделі варіантів використання

Діаграма прецедентів – в UML, діаграма, на якій зображено відношення між акторами та прецедентами в системі. Також, перекладається як діаграма варіантів використання.[2]

Діаграми варіантів використання не можуть описувати внутрішній устрій системи. Діаграми прецедентів відображають елементи моделі варіантів використання.

Суть даної діаграми полягає в наступному: проектована система представляється у вигляді безлічі сутностей чи акторів, що взаємодіють із системою за допомогою так званих варіантів використання. Варіант використання (англ. use case) використовують для описання послуг, які система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, який виконує система при діалозі з актором. При цьому нічого не говориться про те, яким чином буде реалізована взаємодія акторів із системою.[2]

У мові UML є кілька стандартних видів відношень між акторами і варіантами використання:

1. асоціації;
2. включення;
3. розширення;
4. узагальнення.

На (Рис. 2.1) зображена модель варіантів використання веб–додатку для організації та контролю робочого процесу компанії. Актор: користувач (гравець). Прецеденти: вхід в систему, вхід в існуючу гру, створення нової гри, робота з головною формою гри, робота з чатом гри.

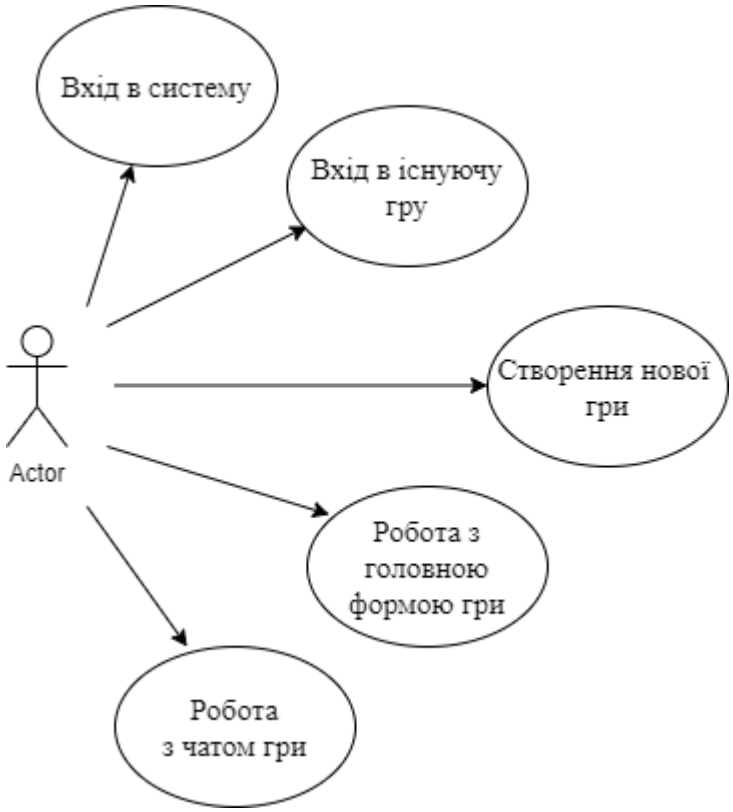



Рисунок 2.1 –Модель варіантів використання у вигляді діаграми прецендентів

## 2.2 Побудова концептуальної моделі

Концептуальна (змістовна) модель – це абстрактна модель, що визначає структуру модельованої системи, властивості її елементів і причинно–наслідкові зв'язки, властиві системи і суттєві для досягнення мети моделювання.

На (Рис. 2.2) представлена концептуальна модель у вигляді діаграми сутність–зв'язок:

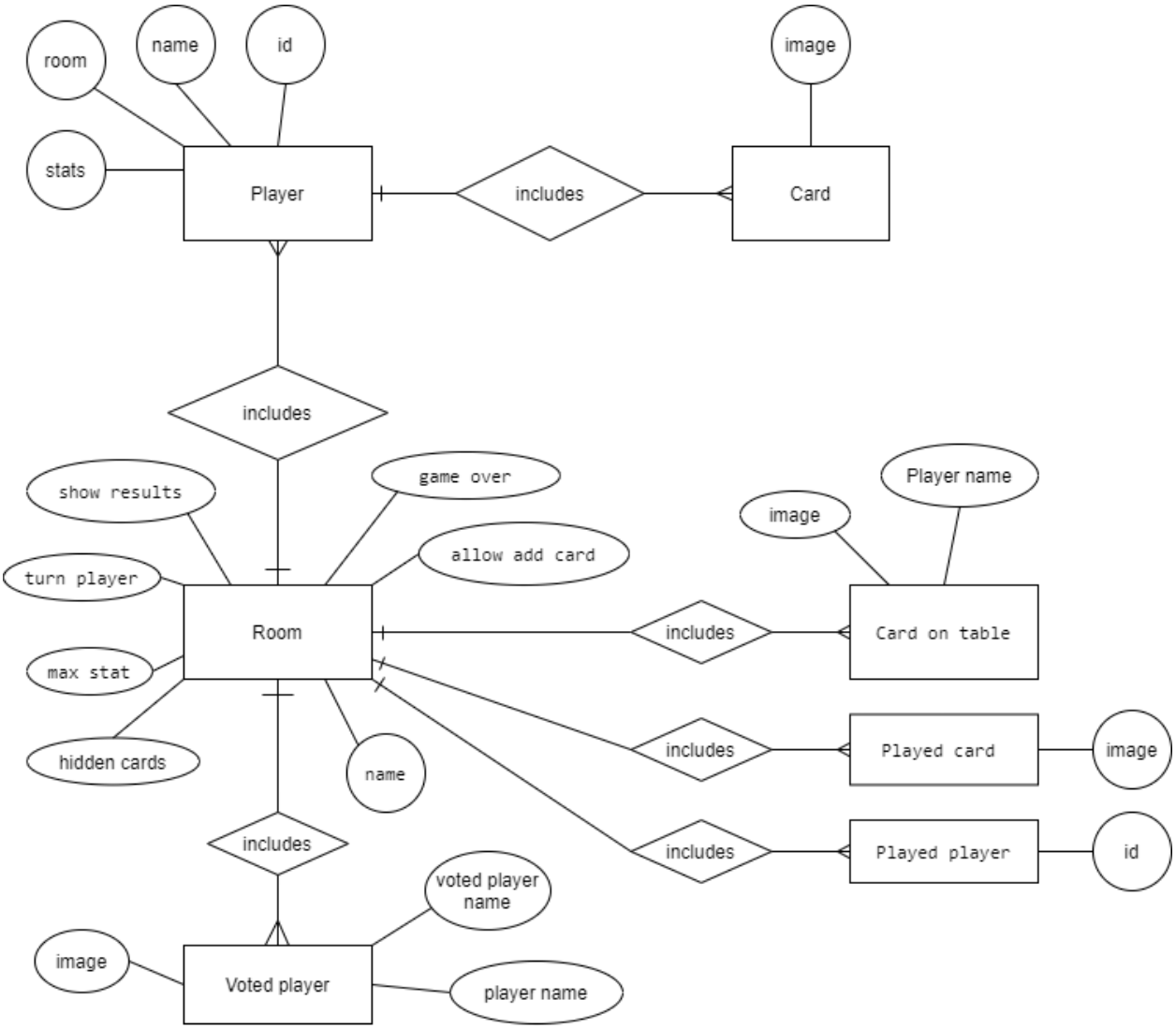



Рисунок 2.2 – Концептуальна модель у вигляді діаграми сутність–зв'язок

## 2.3 Рішення з інформаційного забезпечення

### 2.3.1 Логічна модель даних

Логічна модель даних, або логічна схема – модель даних конкретної предметної області, виражена незалежно від конкретного продукту керування базами даних або технології зберігання (фізична модель даних), але в термінах структур даних, таких як реляційні таблиці та колонки, об'єктно–орієнтовані класи чи теги XML. Вона є протилежністю концептуальній моделі даних, яка описує семантику організації без посилання на технологію.

Логічна модель показує: сутність, їх атрибути, типи даних, ключові атрибути і зв'язки між сутностями. Для побудови логічної моделі необхідно проаналізувати діаграму сутність–зв'язок та з’ясувати, які таблиці та атрибути повинні бути реалізовані в ній. Логічна модель даних додатку представлена на (Рис. 2.3).

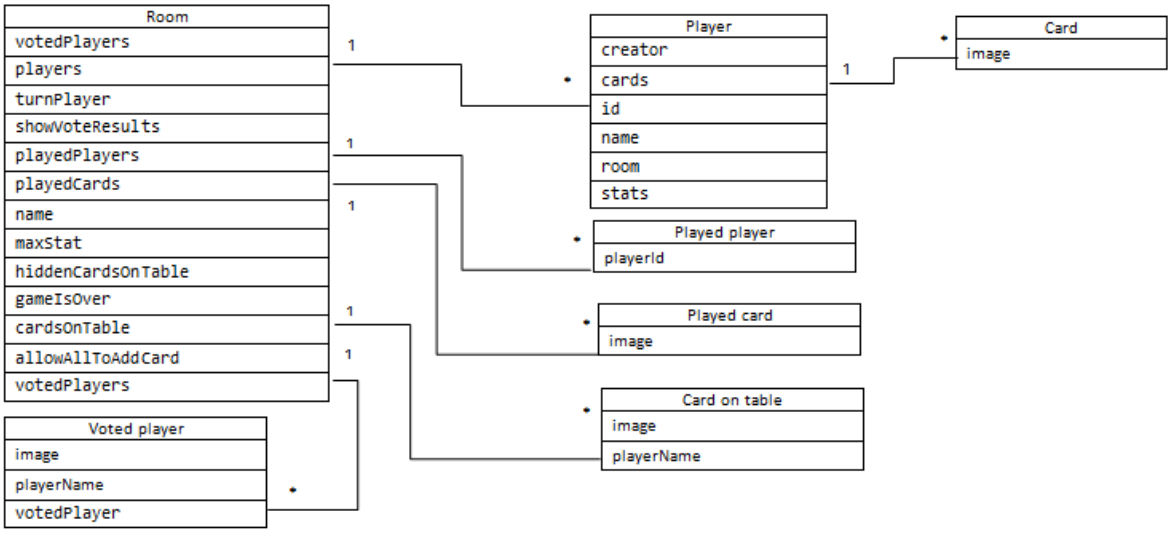



Рисунок 2.3 – Логічна модель данних

### 2.3.2 Фізична модель даних

Фізична модель даних – подання дизайну даних як реалізованого чи призначеного для реалізації у системі керування базами даних. У життєвому циклі проекту вона типово походить від логічної моделі даних, хоча вона може бути зворотно розроблена з даної реалізації бази даних. Завершена фізична модель даних включатиме всі артефакти бази даних, необхідні для створення відношень між таблицями чи для досягнення мети продуктивності, як–от індексів, визначень обмежень, зв'язаних і секціонованих таблиць або кластерів. Аналітики можуть зазвичай використовувати фізичну модель даних для обчислення оцінок зберігання; вона може включати специфічні деталі розподілу сховища для даної системи баз даних. На (Рис. 2.4) наведена фізична модель даних додатку.

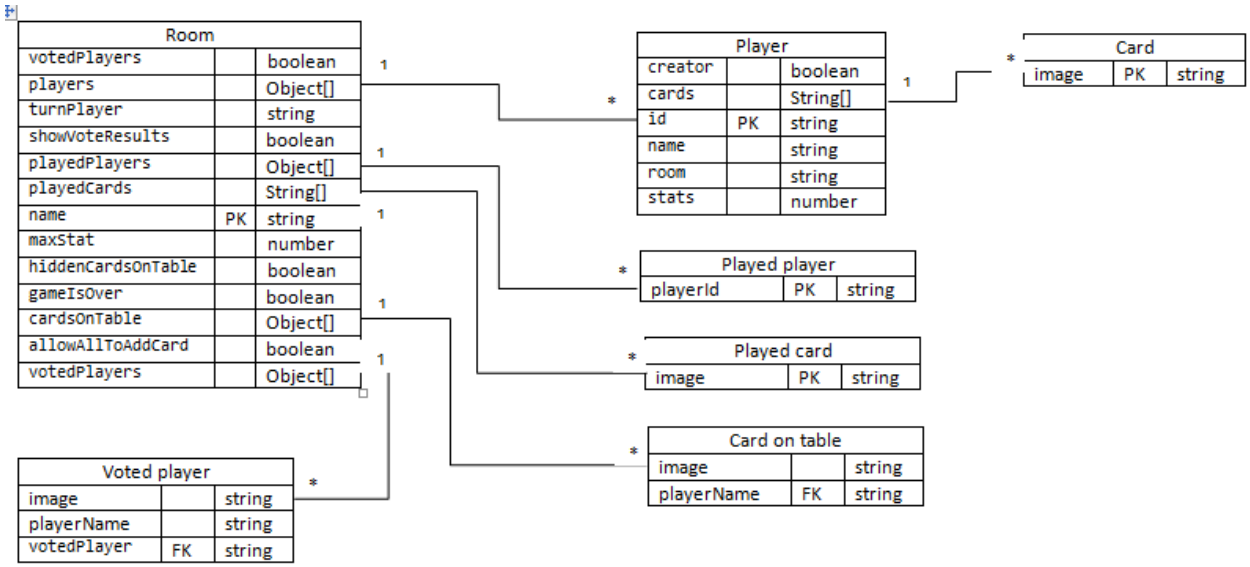



Рисунок 2.4 – Фізична модель данних

### 2.3.3 Діаграма класів

Діаграма класів – статичне представлення структури моделі. Відображає статичні (декларативні) елементи, такі як: класи, типи даних, їх зміст та відношення. Діаграма класів може містити позначення для пакетів та може містити позначення для вкладених пакетів. Також, діаграма класів може містити позначення деяких елементів поведінки, однак їх динаміка розкривається в інших типах діаграм. Діаграма класів служить для представлення статичної структури моделі системи в термінології класів об'єктноорієнтованого програмування. На цій діаграмі показують класи, інтерфейси, об'єкти й кооперації, а також їхні відносини.

На діаграмах класів показуються класи, інтерфейси і відносини між ними. На (Рис. 2.5) представлена діаграма класів веб–додатку мультиплеєрної онлайн–гри «Dixit».

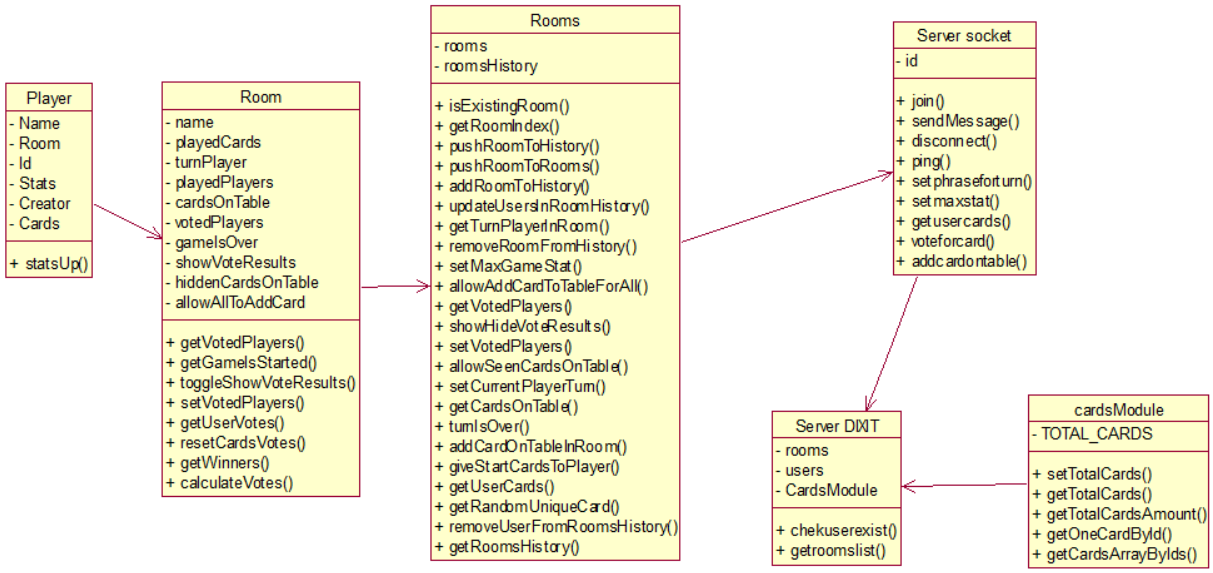



Рисунок 2.5 – Діаграма класів веб–додатку мультиплеєрної онлайн–гри «Dixit»

### 2.3.4 Діаграма послідовності

Діаграма послідовності– різновид діаграми в UML. Діаграма послідовності відображає взаємодії об'єктів впорядкованих за часом. Зокрема, такі діаграми відображають задіяні об'єкти та послідовність відправлених повідомлень.

На діаграмі послідовностей показано у вигляді вертикальних ліній різні процеси або об'єкти, що існують водночас. Надіслані повідомлення зображуються у вигляді горизонтальних ліній, в порядку відправлення.

На (Рис. 2.6) зображено діаграму послідовності для варіанту використання запит на список існуючих ігрових кімнат.

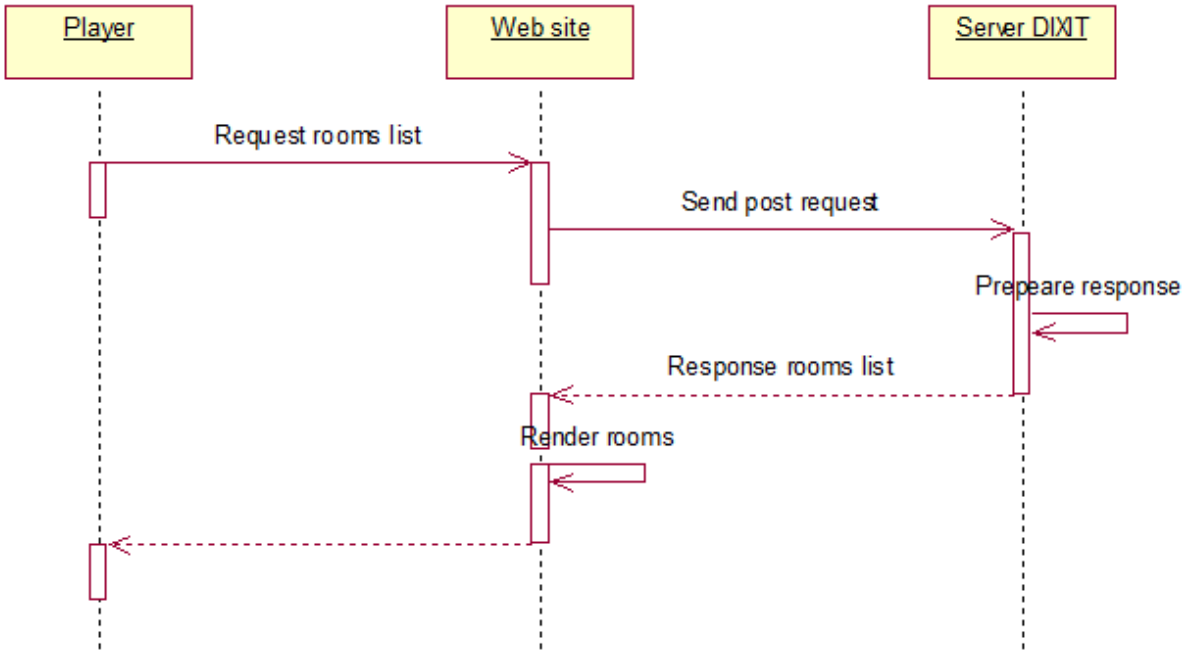



Рисунок 2.6 – Діаграма послідовності для варіанту використання запит на список існуючих ігрових кімнат

## 2.4 Рішення з технічного  забезпечення

### 2.4.1  Опис комплексу технічних засобів

При розробці додатку для організації і контролю робочого процесу компанії використовувались наступні технічні засоби:

1. Visual Studio Code – середа розробки додатків мовою JavaScript;
2. JavaScript – високорівнева мова  веб– програмування;
3. MongoDB – СУБД;
4. Html – мова розмітки що використовується (є офіційним стандартом) для програмування зовнішнього вигляду додатку (дизайну).

## 2.5 Рішення з математичного забезпечення

### 2.5.1 Математична модель

Для підрахунку балів у грі буде використовуватися метод сортування підрахунком.

Сортування підрахунком не використовує операції порівняння, і застосовується для масиву A даних (наприклад, цілих чисел, символів), які приймають значення з невеликого діапазону $ai$ = [0, $K$ - 1]. [3]

Для даного алгоритму застосовують допоміжний масив C, елементи якого з [$i$] = $count$ (A, $i$), де $count$ (A, $i$) - кількість елементів в масиві A, рівних значенням i. Розмір масиву C дорівнює максимальному елементу масиву A.

Наведемо приклад вихідного масиву A і допоміжного масиву C:

a[i] 2 5 3 0 2 3 0 3

i 0 1 2 3 4 5 6 7

c[i] 2 1 3 2 2 3 2 3

З масиву C легко отримати впорядкований масив A. Роблячи прохід по масиву C, ми стільки разів послідовно включаємо в масив A елемент i, скільки разів він зустрічався в A, а саме з [$i$] раз.

Сортування підрахунком може бути реалізовано так, щоб воно володіло властивістю стабільності.

## 2.6 Рішення з програмного забезпечення

### 2.6.1 Вибір мови проектування

Починаючи новий проект, програміст стикається з вибором: який JavaScript фреймворк вибрати для сайту – Vue.js, React або Angular. Відмінності між ними є, і

досить істотні. Однак кожен з них підходить для вирішення завдань. Тому залишається відкритим питання ефективності роботи.

Рендеринг – відображення кінцевого результату.

Javascript фреймворки в першу чергу варто порівняти з рендерингу сторінки. Сучасна архітектура допускає два види: на стороні клієнта (сторінка промальовується за рахунок потужностей ПК користувача) або на стороні сервера.Vue.js і React створюють копію DOM, обробляють її, а потім результат порівнюється з вихідною версією. В кінцевому документі замінюються тільки ті частини сторінки, які відрізняються від результатів обробки. Це значно прискорює завантаження і рендеринг сторінки. Відповідно скорочується обсяг трафіку, що особливо важливо для користувачів мобільних пристроїв.[4]

У корені відрізняється підхід до обробки DOM. Тут відбувається поділ на два потоки, причому за рендеринг DOM «відповідає» браузер (клієнтська частина), а за створення директив, завантаження коду і сервісів – загальний потік (серверна частина).

Але це зовсім не означає, що рендеринг відбувається на стороні клієнта – візуалізація як і раніше проводиться серверами. Отже, SEO оптимізація не викличе труднощів. Пошуковим роботам буде надана коректна сторінка при індексації.

Архітектура компонентів.

React не відноситься до фреймворкывв чистому вигляді. Це свого роду модифікована бібліотека, «заточена» під MVC (Model–View–Controller, де Модель відповідає за надання даних, Вид – відображає дані Моделі користувачеві, а Контролер інтерпретує дії користувача і змушує Модель вносити зміни).

Angular і Vue.js відносяться вже до самих фреймворків. Якщо в основі архітектури проекту лежить React, то це зумовлює:

1. необхідність пошуку і впровадження додаткових бібліотек для реалізації кожної з задач;
2. налаштування функціональної частини програми під конкретну бібліотеку;
3. складність залучення до проекту інших розробників, через різницю в стеці бібліотек кожної програми.

Отже, для реалізації архітектури знадобиться більше часу. Низькорівневий API (Application Programming Interface – набір готових команд. У разі використання готових фреймворків – Vue.js і Angular, проблем з підбором або налаштуванням бібліотек для різних завдань вже не виникає. Високорівнева API забезпечує зворотну сумісність для всіх бібліотек.

Це дозволить підключитися до проекту сторонньому програмісту без тривалого вивчення архітектури додатку. Саме в уніфікації процесів криється популярність повноцінних фреймворків.

Спрямованість і класи залежностей.

React і Vue.js підтримують тільки односторонню передачу даних. При цьому в React об'єкти вміщені. Говорячи простою мовою, кожен з об'єктів програми відноситься до кінцевих процедурам, які не вимагають дій користувача до закінчення роботи.

Однак React підтримує копіювання і передачу стану. Тобто, властивості прописаних об'єктів можуть бути відновлені на іншому пристрої, якщо запустити додаток і повідомити стан компонентів. Отже, рендеринг буде ідентичним, на екранах обох пристроїв буде одна і та ж «картинка».

Зворотна сумісність

Для розробника додатку важлива можливість відновлення архітектури для підключення нових модулів і бібліотек.

Angular – повноцінний фреймворк. Абсолютна залежність від попередніх версій і компонентів. Прямий перехід з 4.0 на 5.0 неможливий, доведеться послідовно встановлювати всі оновлення між версіями. Це призведе до необґрунтованого збільшення обсягу програми. До речі, цікавий факт, Angular версії 3.0 не існує. Після 2–ї версії відразу йде 4–а.

React – бібліотека. Повна сумісність між версіями. Можливе підключення до додатка бібліотек різних версій, оновлення застарілих зі спадкуванням властивостей.

Vue.js – прогресивний фреймворк (згідно із заявою головного розробника Vue.js Евана Ю). Модульна система аналогічна React, але включає в себе всі атрибути JS–фреймворка, що працюють з повною зворотною сумісністю. [5]

«Техпідтримка» – документація і ком'юніті.

При виборі певної архітектури не можна ігнорувати підтримку спільноти програмістів (ком'юніті). Адже навіть прості функції можуть викликати труднощі.

І чим складніше проект, тим частіше розробник звертається і до технічної документації обраної середовища, і за порадою до ком'юніті. Тому варто порівняти «підтримку» кожного JS–фреймворків

React – досить непросто організована «техпідтримка» середовища React. З одного боку, ком'юніті складають тисячі програмістів в різних країнах, що повністю знімає мовний бар'єр. Порадитися з «колегами по цеху» можна на будь–якій мові, російською, англійською, китайською та іншими.

Однак відкритий вихідний код і щоденне поява нових бібліотек на базі React є недоліками середовища. Складно знайти документацію, яка б вичерпно описувала процеси застосування, функції та властивості конкретної свіжої випущеної бібліотеки.[6]

Найчастіше нові модулі з'являються таким чином: програміст створює бібліотеку для вирішення якогось завдання в своєму проекті. Потім він пише інструкція і викладає напрацювання в інтернет.

Vue.js – детальна документація для фреймворка – «візитна картка» Vue.js. Однак прихильники середовища не згадують, що велика частина документації не має перекладу ні на англійську, ні на російську мову.

Спільнота користувачів, незважаючи на велику популярність Vue.js на «Хабре» або Github, задоволене скромне, особливо в порівнянні з Angular або React. Навіть JQuery користується набагато більше розробників, ніж Vue.js.[7]

Отже, для розробки мультиплеєрної онлайн–гри «Dixit» буде використовуватися фреймворк ReactJS.

### 2.6.2 Модель переходів станів

Діаграма станів — діаграма, що визначає зміну станів об'єкту у часі, одна з діаграм моделювання поведінки в UML. Представляє об'єкт як автомат з теорії автоматів зі стандартизованими умовними позначеннями.

Елементами діаграми є:

1. Коло, що позначає початковий стан.
2. Коло з маленьким колом усередині, що позначає кінцевий стан (якщо є).
3. Округлений прямокутник, що позначає окремий стан. Верхівка прямокутника містить назву стану, в середині може бути горизонтальна лінія, під якою записуються активності, що відбуваються в даному стані.
4. Стрілка, що позначає перехід.
5. Назва події (якщо є), що викликає перехід, відзначається над/під стрілкою. Вартовий вираз може бути доданий перед «/» і укладений у квадратні дужки (назва_події), він означає, що перехід відбувається лише за умови істинності виразу. Якщо при переході відбувається якась активність, то воно додається після «/» (назва події).
6. Товста горизонтальна лінія, яка є точкою об'єднання або розгалуження переходів.

На (Рис. 2.7) представлена модель переходів станів для процесу вибору картки.

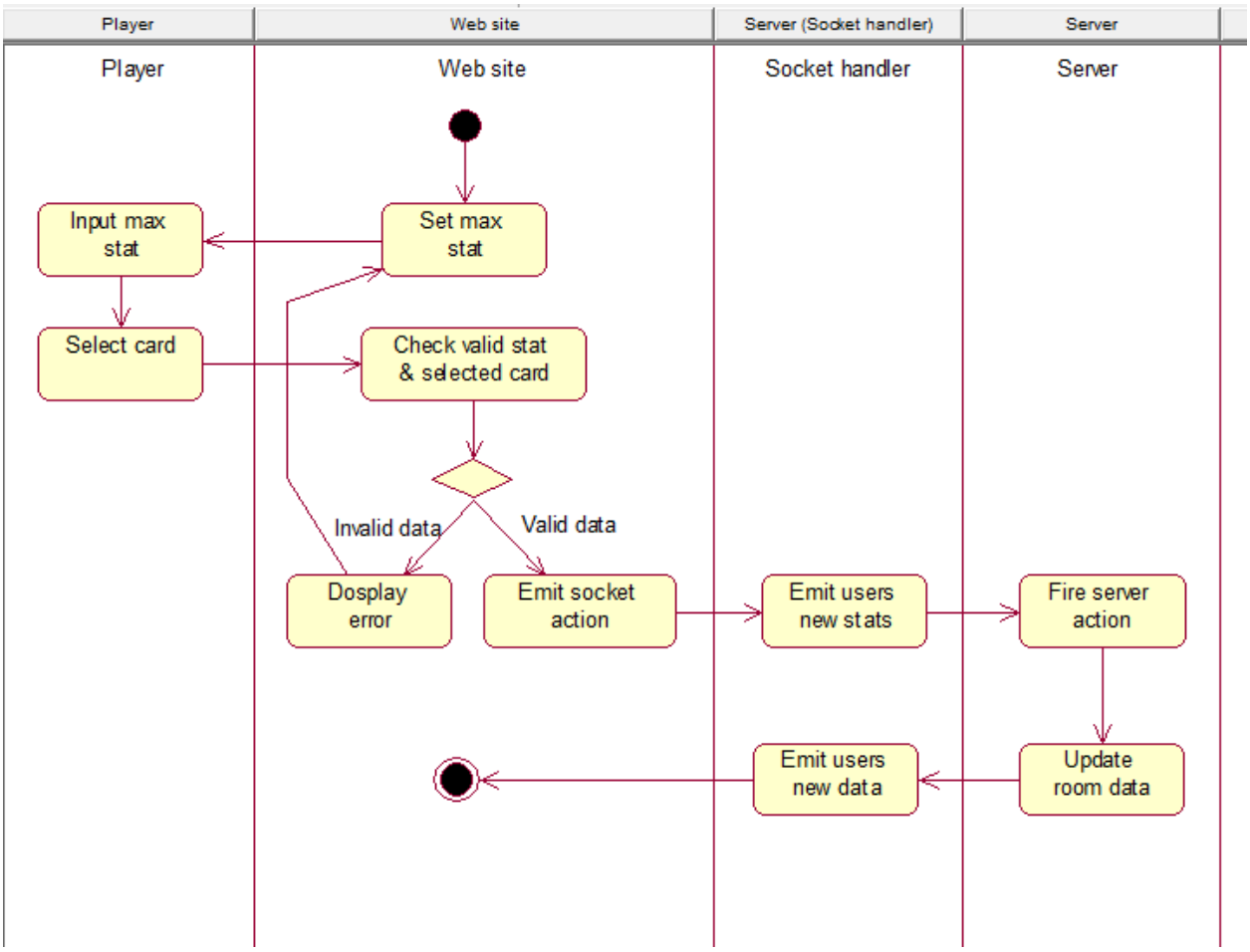



Рисунок 2.7 – Модель переходів станів для процесу вибору картки

## Висновки за розділом 2

У даному розділі були розроблені: моделі варіантів використання та концептуальна, логічна і фізична моделі, модель переходів станів, діаграма послідовності та класів.

# 3. РОЗРОБКА ПРОЕКТНИХ РІШЕНЬ ЩОДО ВЕБ–ДОДАТКУ МУЛЬТИПЛЕЄРНОЇ ОНЛАЙН–ГРИ “DIXIT”

## 3.1 Загальносистемні рішення

### 3.1.1 Представлення схеми організаційної структури

Організаційна структура – це впорядкована сукупність взаємопов'язаних елементів, що знаходяться між собою у стійких взаємостосунках та залежностях, які забезпечують їх функціонування і надають певні права або обмежують їх. У грі не має розподілу прав за правами, обов’язками та можливостями, тому всі грацві мають рівні права.

### 3.1.2 Представлення схеми функціональної структури

Функціональна схема – це схема взаємодії компонентів програмного забезпечення з описом інформаційних потоків, складу даних в потоках. Функціональні схеми більш інформативні, ніж структурні. На (Рис. 3.1) представлена схема функціональної структури для веб–додатку мультиплеєрної онлайн–гри “Dixit”.

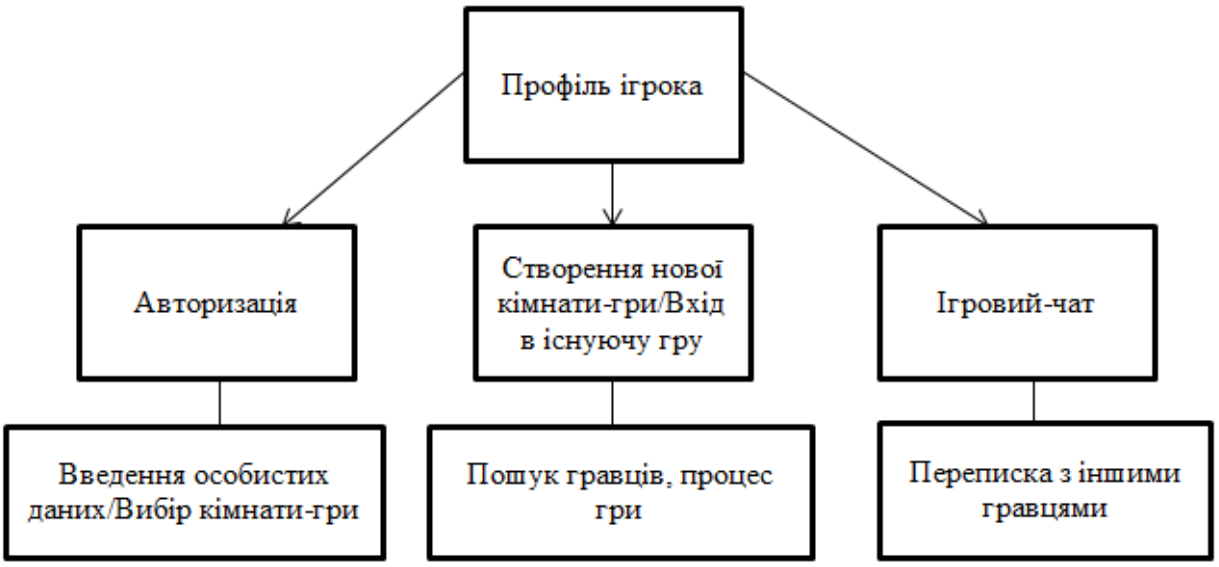



Рисунок 3.1 – Схема функціональної структури

### 3.1.3 Описання функцій, що автоматизуються

Додаток створений для автоматичного підрахунку балів гравців та виведенні результаті. Автоматизуються наступні процеси:

1) процес авторизації користувачів (гравців);
2) процес створення або вибору кімнат–ігор;
3) процес підрахунку балів;
4) процес комунікації користувачів;
5) процес оновлення списку карток.

### 3.1.4 Загальний опис системи

Веб–додаток мультиплеєрна онлайн–гра “Dixit” являє собою веб–сайт.

Граючи у додатку користувачі отримують по 5 карток–картинок, у кожного гравця картки різні і різні між собою. Гравець–організатор починає гру, загадуючи картку та описання під неї. Короткий опис повинен трохи натякати на зміст

картинки, але не розкривати його. Усі інші гравці під загадану картку обирають та додають одну свою, максимально схожу за змістом, адже від цього залежить їх перемога. Коли всі гравці обрали картки, то всі картки відкриваються, тоді кожному гравцю потрібно обрати максимально схожу за описом картку. Якщо ніхто з гравців не відгадав правильно загадану картку, то гравець який загадував отримує 2 бали за кожного гравця, якщо загадану картку відгадали правильно, то гравець який загадував отримує 0 балів, а гравці отримують по 3 бали. Якщо всі гравці обрали картку іншого гравця, то вони отримують 0 балів, а гравець картка якого виявилась більш вдалою до опису отримує 2 бали. Бали сумуються і перемагає той, хто перший набере необхідну кількість балів. Кількість балів для перемоги визначається перед початком кожної нової гри.

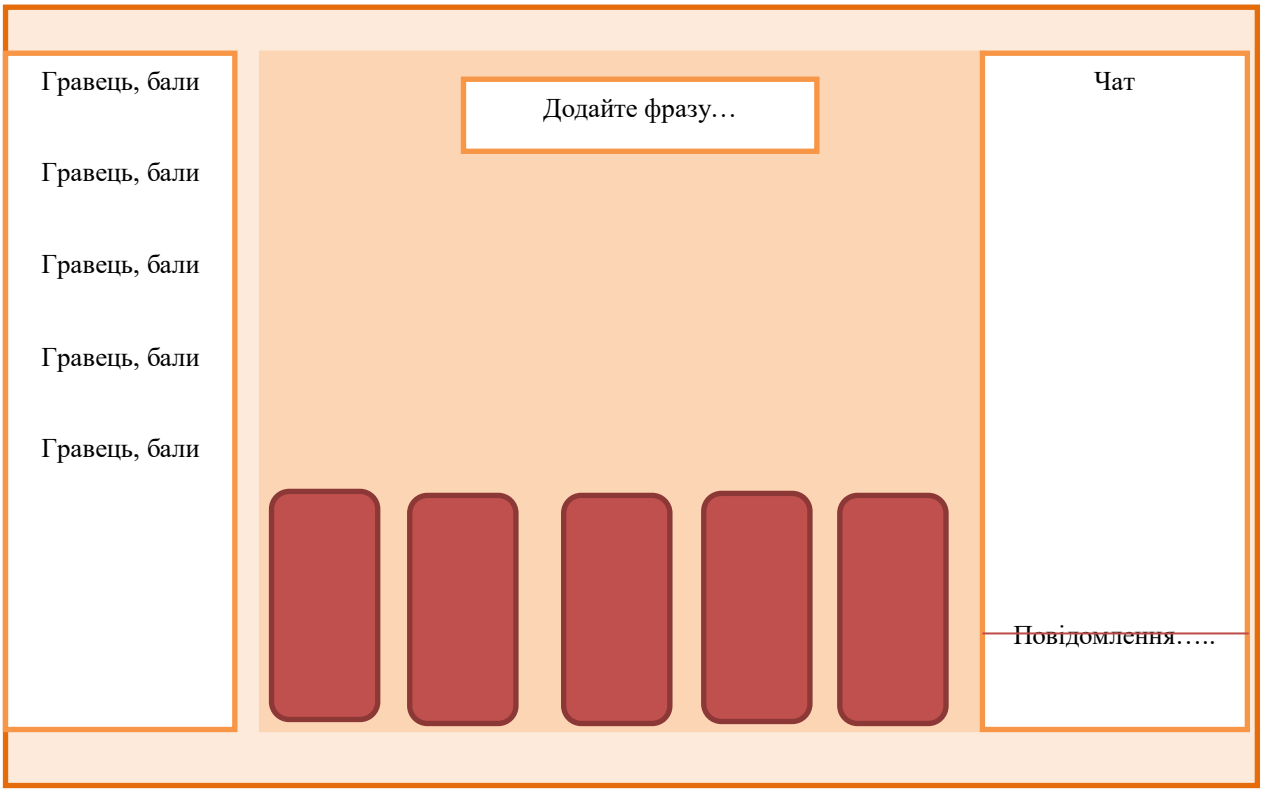
## 3.2 Рішення з організаційного забезпечення

Інтерфейс користувача – засіб зручної взаємодії користувача з інформаційною системою. Сукупність засобів для обробки та відбиття інформації, якнайбільше пристосованих для зручності користувача; у графічних системах інтерфейс користувача, втілюється багатовіконним режимом, змінами кольору, розміру, видимості (прозорість, напівпрозорість, невидимість) вікон, їхнім розташуванням, сортуванням елементів вікон, гнучкими налаштуваннями як самих вікон, так і окремих їх елементів (файли, теки, ярлики, шрифти тощо), доступністю багатокористувацьких налаштувань.

Інтерфейс гри має будти  лаконічним та зрозумілим, щоб гравець міг легко та швидко орієнтуватися, на сторінці гри не буде додано зайвих візуальних ефектів, щоб не відвертати увагу від карток та чату. Інтерфейс повинен бути виконаний у вигляді сторінок з формами та полями. Користувачам, тобто гравцям має виводитись список імен гравців та їх набраних балів, і власні також. З однієї сторони стрінки має бути присутнє вікно чату, у якому буде виводитись інформація гри, дії гравців та повідомлення гравців. Основну частину складатиме

форма з картками, яка міститиме у собі інтерактивне вікно з 5 картками, і вікно для показу карток дійсного ходу. Також буде з’являтися вікно для введення фрази, у якому гравцю потрібно задавати фразу–асоціацію, вікно буде з’являтися тільки в чергу ходу гравця.

На (Рис. 3.2) представлений проект користувальницького інтерфейсу для головної сторінки.




Риунок 3.2 – Проект користувальницького інтерфейсу для головної сторінки

## Висновки до розділу 3

У даному розділі представлена схема функціональної структури. Описані функції, що автоматизувалися у розробці. Виконаний опис системи та розроблене рішення з організаційного забезпечення.

# 4. РОЗРАХУНОК І ОЦІНКА ЕКОНОМІЧНОГО ЕФЕКТУ ВІД РОЗРОБКИ ВЕБ–ДОДАТКУ МУЛЬТИПЛЕЄРНОЇ ОНЛАЙН–ГРИ “DIXIT”

## 4.1 Загальна характеристика роботи

Техніко–економічне обґрунтування – це розрахунок економічної доцільності здійснення проекту, заснований на порівняльній оцінці витрат і результатів ефективності використання, а також строку окупності вкладень.

Розробка веб–додатку мультиплеєрної онлайн–гри “Dixit”:

1. розробка програмного продукту;
2. впровадження програмного продукту.

## 4.2 Розрахунок витрат на проектування та впровадження системи

Витрати на розробку і впровадження програмних засобів (К) включають:

К = К1,+К2

де К1 – витрати на розробку програмних засобів, грн.

К2 – витрати на відлагодження і дослідну експлуатацію програми рішення задачі на ЕОМ, грн.

Витрати на розробку програмних засобів включають:

1. витрати на оплату праці розробників;
2. витрати на відрахування у спеціальні державні фонди (Вф,);
3. витрати на куповані вироби (Кв);
4. витрати на придбання спецобладнання для експериментальних робіт (Об);
5. накладні витрати (Н);
6. інші витрати (їв).

Витрати на оплату праці розробників проекту визначаються за формулою:

де nij – чисельність розробників і–ої спеціальності j–го тарифного

розряду, які приймають участь в проектуванні, чол.;

tij – час, який затрачений на розробку проекту співробітника і–ої спеціальності j–го тарифного розряду, днів;

Сij – денна заробітна плата і–ої спеціальності j–го тарифного розряду, грн.;

де Сij – основна місячна заробітна плата розробника і–ої спеціальності j–го тарифного розряду, грн.;

h – коефіцієнт, що визначає розмір додаткової заробітної плати;

р – середня кількість робочих днів у місяці.

Таблиця 5.1. Вихідні дані для розрахунку витрат на оплату праці

| Посада виконавців | Середньоденна ставка, грн./дні |
|---|---|
| Доцент | 81,8 |
| Консультант з економіки | 81,8 |
| Консультант з охорони праці | 65,5 |
| Студент | 7,3 |

Таблиця 5.2. Розрахунок витрат на оплату праці

| Спеціальність розробника | Час розробки, ДНІ | Денна заробітна плата, грн. | Витрати на розробку, грн. |
|---|---|---|---|
| Доцент | | 81,8 | 1636,0 |
| Консультант з економіки | | 81,8 | 163,6 |
| Консультант з охорони праці | | 65,5 | 65,5 |
| Студент | | 7,3 | 474,5 |
| Разом | | | 2339,6 |

Величину відрахувань у спеціальні державні фонди визначають у процентному співвідношенні від суми основної та додаткової заробітної плати. Згідно діючого нормативного законодавства сума відрахувань у спеціальні державні фонди складає 36,2% від суми заробітної плати:

Вф=36,2: 100*З

Вф=0,362*2339,6=846,9 грн.

При розробці даного програмного забезпечення спеціальне обладнання не використовувалось, тому витрати на спеціальне обладнання відсутні.

Накладні витрати проектних організацій включають три групи видатків: витрати на управління, загальногосподарські витрати, невиробничі витрати. Вони розраховуються за встановленими процентами до витрат на оплату праці:

Н = 0,3*2339,6=701,9 грн.

Інші витрати відображають видатки, які не враховані в інших статтях витрат. Вони розраховуються за встановленими процентами до витрат на оплату праці:

Ів =10:100*З

Ів=0,1*2339,6=233,96 грн.

Витрати на розробку програмного забезпечення розраховуються за формулою:

К=З + Вф + Кв + Об + Н + Ів

К =2339,6+846,9+26+701,9+233,9=4122,3 грн.

Витрати на відлагодження і дослідну експлуатацію програмного забезпечення визначаються за формулою:

К2=SMr*tВід,

де SMr – вартість однієї машино–години роботи конкретного типу ЕОМ, грн./год.;

tВід – машинний час, витрачений на відлагодження і дослідну експлуатацію програмних засобів, год.

Загальна кількість днів роботи на ЕОМ рівна 60 днів. Середній щоденний час роботи на ЕОМ – 2 год., тому:

tвід=60*2=120год.

За даними обчислювального центру вартість для ЕОМ типу IBM PC/AT SMг = 2,5 грн.

Отже: К2= 2,5*120 = 300 грн.

Кошторис витрат на розробку програмного забезпечення.

Найменування елементів витрат та сума витрат, грн.:

1. Витрати на оплату праці=2339,6

2. Відрахування у спеціальні державні фонди=846,9

3. Накладні витрати=701,9

4. Інші витрати=233,9

5. Витрати на відлагодження і дослідну експлуатацію програмного забезпечення=300,0
6. Всього=4422,3

Висновки до розділу 4

У даному розділі був виконаний розрахунок витрат на проектування та впровадження додатку мультиплеєрна онлайн–гра «Dixit», що розроблялася у ході виконання дипломної роботи.

# 5. ОХОРОНА ПРАЦІ

## 5.1 Вступ

Охорона праці – це система правових, соціально–економічних, організаційно–технічних, санітарно–гігієнічних та лікувально–профілактичних заходів і засобів, спрямованих на збереження здоров'я та працездатності людини в процесі праці.

Законодавчими актами, що визначають основні положення про охорону праці, є загальні закони України, а також спеціальні законодавчі акти. До загальних законів, що визначають основні положення про охорону праці належать: Конституція України, Закони України «Про охорону праці», «Про охорону здоров'я», «Про пожежну безпеку», «Про використання ядерної енергії та радіаційний захист», «Про забезпечення санітарного та епідемічного благополуччя населення», «Про загальнообов'язкове державне соціальне страхування від нещасного випадку на виробництві та професійного захво–рювання, які спричинили втрату працездатності», Кодекс законів про працю України (КЗпП). Спеціальними законодавчими актами в галузі охорони праці є Державні нормативні акти про охорону праці, Державні стандарти Системи стандартів безпеки праці, будівельні норми та правила, санітарні норми, правила технічної експлуатації електроустановок споживачів та інші нормативні документи.

## 5.2 Аналіз небезпечних і шкідливих факторів

У приміщенні по розрахунках з побутовими і промисловими споживачами в обчислювальному центрі виробничого управління теплової енергії можуть мати місце такі фізичні і психологічні шкідливі фактори, як – порушення стану мікроклімату, недостатня освітленість робочої зони, забруднення повітря на робочих місцях, виробничий шум та вібрація, електромагнітні випромінювання,

електростатичні поля, іонний склад повітря, відсутність чи недолік природного світла, поразка електричним струмом, загоряння, монотонність праці, перенапруга очей, емоційні перевантаження.

### 5.2.1 Освітленість робочого місця

Норми освітлення залежать від параметрів, які передбачено роботою. Відстань від очей до предмета праці повинна бути визначена в кожному окремому випадку. Що менше відношення діаметра деталі до відстані від очей , то інтенсивнішим повинно бути освітлення. При цьому необхідно ураховувати й здатність поверхні відбивати світло. Спектр джерел світла повинен максимально наближатися до спектра сонячного випромінювання. Важливо також захистити очі робітники від сліпучого світла. Усі системи освітлення повинні забезпечувати правильне сприйняття відтінків світла, аби в робочих приміщеннях було рівномірне освітлення. Тому слід подбати про загальне та місцеве. Освітлювальні пристрої мають забезпечувати гігієнічні вимоги: освітлення, якого було б достатньо для виконання певної роботи без напруження зору; рівномірність освітлення, без тіней, у межах робочої поверхні, рівень освітлення проходів; захист очей від блиску; виконання вимог безпеки (шляхом обладнання в окремих випадках аварійного освітлення).

Нормативні величини освітленості робочих місць для різних видів робіт та відповідних зорових навантажень визначаються ДБН Б.2.5.–28–2006 «Природне і штучне освітлення». Для роз’яснення зазначимо, що робоча поверхня – головний об’єкт при встановленні регламентованих норм освітлення. Під робочою поверхнею, як об’єкта для нормування рівнів освітленості, розуміють поверхню робочого столу, верстака, частини обладнання, або інструмента, на якій проводиться робота та для якої нормується або на якій вимірюється освітленість.

### 5.2.2 Вплив випромінювання

Джерела енергії ЕМП радіочастотного діапазону поділяються на технологічні (основні) та додаткові. До технологічних належать плавильні або гартувальні контури, пластини конденсаторів, фідерні лінії. У радіотехнічних пристроях це генератори та ЗВЧ–блоки, антенні системи, елементи хвилеводних трактів. До додаткових джерел належать виносні трансформатори, батареї конденсаторів змінного струму. У радіотехнічних пристроях додатковими джерелами є неякісно екрановані ВЧ–елементи передатчиків і пристроїв складання потужностей та роздільних фільтрів, неекрановані лінії передачі електромагнітної енергії на антени.

Напруга електричного поля вимірюється у вольтах на метр – В/м, а магнітного поля – в амперах на метр – А/м. Інтенсивність електромагнітного поля з різними хвилями, що діють на працівника, оцінюється за величиною щільності потоку енергії, яка падає на одиницю поверхні, і виражається у ватах на квадратний метр (Вт/м2) або в довільних одиницях: міліватах, мікроватах на квадратний сантиметр (мВт/см2, мкВт/см2).

Електромагнітні поля особливо негативно впливають на організм людини, яка безпосередньо працює з джерелом випромінювання. В діапазоні промислових частот більше негативний вплив на біологічний об’єкт має електрична складова поля.

Найчутливішими до ЕМП є нейродинамічні процеси, які прямо чи побічно перемикають хронобіологічні процеси організму на патологічний або стресовий режим функціонування.

При дії ЕМП на людину можливі гострі та хронічні форми порушення фізіологічних функцій організму. Такі порушення виникають в результаті дії електричної складової ЕМП на нервову систему, а також на структуру кори головного та спинного мозку, серцево–судинної системи.

У більшості випадків такі зміни в діяльності нервової та серцево–судинної системи мають зворотній характер, але в результаті тривалої дії вони накопичуються, підсилюються з плином часу, але, як правило, зменшуються та зникають при виключенні впливу та поліпшенні умов праці. Тривалий та інтенсивний вплив ЕМП призводить до стійких порушень в організмі людини та захворювань.

Сумісна дія випромінювань широкого діапазону може викликати окрему радіохвильову хворобу.

Тяжкість її наслідків прямо залежить від напруженості ЕМП, фізичних особливостей різних діапазонів частот, тривалості впливу, умов навколишнього середовища, а також від функціонального стану та стійкості організму до впливу різних чинників, можливостей адаптації. Збільшується ризик виникнення загальних захворювань, захворювань органів дихання, травлення тощо. Це може відбуватися також і за дуже невеликої інтенсивності ЕМП, яка незначно перевищує гігієнічні нормативи.

Результатом дії на організм людини електромагнітних випромінювань в діапазоні 30 кГц – 300 МГц є: загальна слабкість, підвищена втома, порушення сну, головний біль та біль в ділянці серця. З’являється роздратованість, втрачається увага, сповільнюються рухово–мовні реакції.

### 5.2.3 Вплив електричного струму

Під час монтажу та експлуатації ліній електромережі необхідно повністю унеможливити виникнення електричного джерела згорання внаслідок короткого замикання та перевантаження проводів, обмежувати застосування проводів з легкозаймистою ізоляцією і, за можливістю, перейти на негорючу ізоляцію.

Усі провідники повинні відповідати номінальним параметрам мережі та навантаження, умовам навколишнього середовища, умовам розподілу провідників, температурному режиму та типам апаратури захисту, вимогам ПВЕ.

У приміщенні, де одночасно експлуатується або обслуговується більше п'яти персональних ПК, на помітному та доступному місці встановлюється аварійний резервний вимикач, який може повністю вимкнути електричне живлення приміщення, крім освітлення.

Протікання струму через тіло людини супроводжується термічним, електролітичним та біологічним ефектами.

Термічна дія струму полягає в нагріванні тканини, випаровуванні вологи, що викликає опіки, обвуглювання тканин та їх розриви парою. Тяжкість термічної дії струму залежить від величини струму, опору проходженню струму та часу проходження. За короткочасної дії струму термічна складова може бути визначальною в характері і тяжкості ураження.

Електролітична дія струму проявляється в розкладі органічної речовини (її електролізі), в тому числі і крові, що призводить до зміни їх фізико–хімічних і біохімічних властивостей. Останнє, в свою чергу, призводить до порушення біохімічних процесів у тканинах і органах, які є основою забезпечення життєдіяльності організму.

Біологічна дія струму проявляється у подразненні і збуренні живих тканин організму, в тому числі і на клітинному рівні. При цьому порушуються внутрішні біоелектричні процеси, що протікають в організмі, який нормально функціонує, і пов'язані з його життєвими функціями. Збурення, спричинене подразнюючою дією струму, може проявлятися у вигляді мимовільного непередбачу–ваного скорочення м'язів.

### 5.2.4 Виробничий шум

Шум – одна з форм фізичної дії середовища, безпосередньо впливає на працездатність.

Джерелами шуму є: транспорт, насоси, двигуни, пневматичні та електричні інструменти, верстати, будівельна техніка. У нашому випадку також додається

шум, який утворює людський натовп, оскільки торгівельні організації підприємства розміщені на ринках.

Для забезпечення нормованих рівнів шуму у виробничих приміщеннях та на робочих місцях застосовуються шумопоглинальні засоби, вибір яких обґрунтовується спеціальними інженерно–акустичними розрахунками.

Як засоби шумопоглинання повинні застосовуватися негорючі або важкогорючі спеціальні перфоровані плити, панелі, мінеральна вата з максимальним коефіцієнтом звукопоглинання в межах частот 31,5 – 8000 Гц, або інші матеріали аналогічного призначення, дозволені для оздоблення приміщень органами державного санітарно–епідеміологічного нагляду. Крім того, необхідно застосовувати підвісні стелі з аналогічними властивостями.

Рівні вібрації під час виконання робіт з ПК у виробничих приміщеннях не повинні перевищувати допустимих значень, визначених в СН 3044–84 та ДСТУ 12.1.012–90 «Вибрационная безопасность. Общие требования».

Для зниження вібрації обладнання, пристрої, пристосування необхідно встановлювати на спеціальні амортизуючи прокладки, передбачені нормативними документами.

### 5.2.5 Умови мікроклімату

Суттєвий вплив на стан організму працівника, його працездатність здійснює мікроклімат у приміщенні, що визначається діючою на організм людини сукупністю температури, волоДСТУі, руху повітря та теплового випромінювання нагрітих поверхонь.

Приміщення по розрахунках з побутовими і промисловими споживачами в обчислювальному центрі виробничого управління теплової енергії повинно бути обладнане системами опалення, кондиціювання повітря або припливно–витяжною вентиляцією відповідно до СНиП 2.04.05–91 «Вентиляция производственных помещений».

Параметри мікроклімату, іонного складу повітря, вміст шкідливих речовин на робочих місцях, оснащених відеотерміналами, повинні відповідати вимогам пункту 2.4 СН 4088–86 «Санитарные нормы микроклимат производственных помещений», ДСТУ 12.1.005–88 «ССБТ Общие санитарно– гигиенические требования к воздуху рабочей зоны» (таблиця 5.1), СН 2152–80 «Санитарно– гигиенические нормы допустимых уровней ионизации воздуха производственных и общественных помещений» (таблиця 5.2).

Таблиця 5.1 – Нормовані параметри мікроклімату для приміщень з ВДТ та ППК

| Пора року | Категорія робіт згідно з ДСТУ 12.1–005–88 | Температура повітря, °С оптимальна | Відносна вологість повітря, % оптимальна | Швидкість руху повітря, м/с оптимальна |
|---|---|---|---|---|
| Холодна | Легка – 1 а | 22–24 | 40–60 | 0,1 |
| | Легка – 16 | 21–23 | 40–60 | 0,1 |
| Тепла | Легка – 1 а | 23–25 | 40–60 | 0,1 |
| | Легка – 1 б | 22–24 | 40–60 | 0,2 |

Таблиця 5.2 – Рівні іонізації повітря приміщень при роботі на ВДТ та 1111К (відповідно до СН 2152–80)

| Рівні | Кількість іонів в 1 $см^3$ повітря | |
|---|---|---|
| | $n^+$ | $n^-$ |
| Мінімально необхідні | 400 | 600 |
| Оптимальні | 1500 – 3000 | 3000 – 5000 |
| Максимально допустимі | 50000 | 50000 |

Для підтримки допустимих значень мікроклімату та концентрації позитивних та негативних іонів необхідно передбачати установки або прилади зволоження

та/або штучної іонізації, кондиціювання повітря.

### 5.2.6 Організація робочого місця

Робоче місце – це місце постійного або тимчасового перебування працівника в процесі трудової діяльності. Організація робочого місця користувача приміщення по розрахунках з побутовими і промисловими споживачами в обчислювальному центрі Виробничого управління теплової енергії повинна забезпечувати відповідність усіх елементів робочого місця та їх розташування ергономічним вимогам ДСТУ 12.2.032–78 «ССБТ. Рабочее место при выполнении работ сидя. Общие эргономические требования».

Площа, виділена для одного робочого місця з відео терміналом або персональною ПК, повинна складати не менше 6 м , а обсяг – не менше 20 м .

Робочі місця з відео терміналами відносно світлових прорізів повинні розміщуватися так, щоб природне світло падало збоку, переважно зліва.

При розміщенні робочих місць необхідно дотримуватись таких вимог: робочі місця розміщуються на відстані не менше 1 м від стін зі світловимипрорізами;

1) відстань між бічними поверхнями відео терміналів має бути не меншою за 1,2 м;

2) відстань між тильною поверхнею одного відео термінала та екраном іншого не повинна бути меншою 2,5 м;

3) прохід між рядами робочих місць має бути не меншим 1 м.

Вимоги щодо відстані між бічними поверхнями відео терміналів та відстані між тильною поверхнею одного відео терміналу та екраном іншого враховуються також при розміщенні робочих місць з відео терміналами та персональними комп’ютерів в суміжних приміщеннях, з урахуванням конструктивних особливостей стін та перегородок.

Висота робочої поверхні столу для відео терміналу має бути в межах 680–800 мм, а ширина – забезпечувати можливість виконання операцій в зоні досяжності моторного поля.

Робочий стіл для відео терміналу, як правило, має бути обладнаним підставкою для ніг шириною не менше 300 мм та глибиною не менше 400 мм, з можливістю регулювання по висоті в межах 150 мм та кута нахилу опорної поверхні – в межах 20°. Підставка повинна мати рифлену поверхню та бортик на передньому краї заввишки 10 мм.

Робоче сидіння (стілець, крісло) користувача відео терміналу та персональної ПК повинно мати такі основні елементи: сидіння, спинку та стаціонарні або знімні підлокітники, також повинно бути підйомно–поворотним, таким, що регулюється за висотою, кутом нахилу сидіння та спинки, за відстанню спинки до переднього краю сидіння, висотою підлокітників. Поверхня сидіння має бути плоскою, передній край – заокругленим. Висота спинки сидіння має становити 300±20 мм, ширина – не менше 380 мм, радіус кривизни в горизонтальній площині – 400 мм. Кут нахилу спинки повинен регулюватися в межах 0–30° відносно вертикального положення. Відстань від спинки до переднього краю сидіння повинна регулюватись у межах 260–400 мм.

Екран відео термінала та клавіатура мають розташовуватися на оптимальній відстані від очей користувача, але не ближче 600 мм, з урахуванням розміру алфавітно–цифрових знаків та символів.

Клавіатуру слід розміщувати на поверхні столу або на спеціальній, регульовуваній за висотою, робочій поверхні окремо від столу на відстані 100–300 мм від краю, ближчого до працівника. Кут нахилу клавіатури має бути в межах 5–15°.

## 5.3 Розрахунок освітленості робочого місця

Зробити розрахунок КПО попереднім методом для виробничої лабораторії з боковим одностороннім освітленням. Приміщення розташоване в 4 поясі світлового клімату. Довжина приміщення: 13м, глибина: 8м. Розрахункова точка А знаходиться від зовнішньої стіни на відстані: 7 м. Відстань від робочої поверхні до

верха вікон: 4.4 м.

Нормоване значення коефіцієнту бокового природного освітлення для III розряду зорової роботи III світлового поясу $e_н^{III}$ – 1.5%. У зв'язку з тим, що приміщення знаходиться в IV світловому поясі, КПО буде визначено за формулою:

1. $e_н^{IV} = e_н^{III} mc$

де m = 0,9–коефіцієнт світлового клімату;

2. c = 0,7 – коефіцієнт сонячності клімату.

3. $e_н^{IV} = 1.5x0.9x0.7 = 0.945\%$

Загальний коефіцієнт світло пропускання визначається за формулою:

$\tau_0 = \tau_1 \tau_2 \tau_3 \tau_4 \tau_5$

де $\tau_1$ = 0,8 – засклення подвійне;

$\tau_2$ = 0,65 – Плетення подвійне дерев'яне;

$\tau_3$ = 1 – Несущі конструкції відсутні;

$\tau_4$ = 1 – Сонце захисні конструкції відсутні

$\tau_5$ = 1 – Захисних сіток немає

$\tau_0$=0.8x0.65x1x1x1=0,52

Площа світлових прорізів при боковому освітленні визначається за формулою:

$$S_б = \frac{S_п e_н K_з \eta_в K_б}{100 r_1 \tau_0}$$

де $S_п$ – площа підлоги приміщення,

$S_п$ = 13 х 8 = 104 м$^2$;

$K_З$ = 1,2 – коефіцієнт запасу;

$\eta_в$ = 13 – світлова характеристика вікон;

$к_б$ = 1 – коефіцієнт, що враховує затінення вікон будинками, що стоять навпроти;

$r_1$ = 2,4 – коефіцієнт, що враховує підвищення КПО при боковому освітленні завдяки світлу, відбитому від поверхонь приміщення і підстильного шару, що прилягає до будинку.

$$S_б = \frac{104x1,26x1,2x13x1}{100x2,4x0,52} = 16,38м^2$$

Загальна ширина вікон знаходиться за допомогою наступної формули:

$$b_в = \frac{S_б}{h_в};$$

де $h_В$ = 4 м – висота вікна;

$$b_в = \frac{8,24}{4} = 2,06м;$$

Приймається 2 вікна, які мають ширину 2 м.

## 5.4 Розробка заходів щодо зменшення впливу небезпечних і шкідливих факторів

### 5.4.1. Заходи щодо зниження величини виникаючих зарядів статичної електрики

Для запобігання можливості виникнення небезпечних розрядів з поверхні обладнання, речовин, що перероблюються, а також з тіла людини необхідно передбачати, з урахуванням особливостей виробництваі заходи, які можуть забезпечити відведення заряду:

1) зниження інтенсивності генерації заряду статичної електрики;
2) відведення заряду шляхом заземлення обладнання та комунікацій, а також забезпечення постійного електричного контакту з заземленням тіла людини;
3) відведення заряду шляхом зменшення питомого об’ємного та поверхневого електричного опору;
4) нейтралізація заряду шляхом використання різних засобів захисту від статичної електрики по ДСТУ 12.4.124–83.

### 5.4.2 Заходи щодо зниження шуму на робочому місці

Заходи та засоби захисту від шуму поділяються на колективні та індивідуальні, причому останні застосовуються лише тоді, коли заходами та засобами колективного захисту не вдається знизити рівні шуму на робочих місцях до допустимих значень. Призначення засобів індивідуального захисту від шуму – перекрити найбільш чутливі канали проникнення звуку в організм – вуха. Тим самим різко послаблюються рівні звуків, що діють на барабанну перетинку, а відтак – і коливання чутливих елементів внутрішнього вуха. Такі засоби дозволяють одночасно попередити розлад нервової системи від дії інтенсивного подразника, яким є шум.

До засобів захисту від шуму належать навушники, протишумові вкладки, шумозаглушувальні шоломи. Вибір обумовлюється видом та характеристикою шуму на робочому місці, зручністю використання засобу при виконанні даної робочої операції та конкретними кліматичними умовами.

Засоби колективного захисту від шуму подібно до віброзахисту поділяються за такими напрямками:

1) зменшення шуму в самому джерелі;

2) зменшення шуму на шляху його поширення;

3) організаційно–технічні заходи;

4) лікувально–профілактичні заходи.

Зменшення шуму в самому джерелі – найбільш радикальний засіб боротьби з шумом, що створюється устаткуванням.

## Висновки до розділу 5

У даному розділі проведено аналіз потенційно – небезпечних та шкідливих факторів у робочому приміщенні. Розроблені заходи щодо усунення або зниження впливу цих факторів на робітників до нормативних вимог. Дотримання умов, що визначають оптимальну організацію робочого місця програміста, дозволить зберегти гарну працездатність протягом усього робочого дня, що у свою чергу підвищить як у кількісному, так і в якісному відношеннях продуктивність праці.

Список використаних джерел

1. Безпека життєдіяльності. Охорона праці / П.П. Кукін і ін. − М.: Вища школа, 2015. − 336 с.

2. Безпека життєдіяльності. Виробнича безпека та охорона праці. − М.: Вища школа, 2015. − 432 с.

3. Бєляков, Г. І. Безпека життєдіяльності. Охорона праці / Г.І. Бєляков. − М.: Юрайт, 2016. − 576 с.

4. Білярі Ю. А. Охорона праці в організації. Практичні рекомендації / Ю.А. Білярі, В.В. Ударів. − М.: Книжковий світ, 2016. − 176 с.

5. Вашко І. М. Охорона праці / І. М. Вашко. − М.: ТетраСистемс, 2016. − 208 с.

6. Михайлов Ю. М. Охорона праці в офісі / Ю.М. Михайлов. − М.: Альфа−прес, 2016. − 256 с.

7. Попов Ю. П. Охорона праці / Ю.П. Попов. − М.: КноРус, 2015. − 224 с.

8. Сергєєв А. Г. Менеджмент і сертифікація якості охорони праці на підприємстві / А.Г. Сергєєв, Е.А. Баландіна, В.В. Баландіна. − М.: Логос, 2015. − 216 с.

# 6. ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

## 6.1 Вступ

Охорона навколишнього середовища – це система заходів щодо раціонального використання природних ресурсів, збереження особливо цінних та унікальних природних комплексів і забезпечення екологічної безпеки. Це сукупність державних, адміністративних, правових, економічних, політичних і суспільних заходів, спрямованих на раціональне використання, відтворення і збереження природних ресурсів землі, обмеження негативного впливу людської діяльності на довкілля.

Система включає державні, суспільні та міжнародні заходи, які забезпечують раціональне використання, відновлення, примноження та збереження природних ресурсів від руйнування, забруднення та виснаження. Охорона навколишнього середовища має велике економічне та соціально-політичне значення, вона здійснюється з господарською, науковою, оздоровчою та культурною метою. При оцінюванні наслідків антропогенного впливу на навколишнє середовище важливе місце належить визначенню допустимих масштабів впливу, зокрема гранично допустимих концентрацій різних речовин – забруднювачів атмосфери, води та ґрунту.

## 6.2 Загальні відомості про забруднення

Забруднення довкілля – процес зміни складу і властивостей однієї або декількох сфер Землі внаслідок діяльності людини. Приводить до погіршення якості атмосфери, гідросфери, літосфери та біосфери. Допустима міра забруднення довкілля в різних країнах регламентується відповідними стандартами, нормативами, законами. Розрізняють забруднення отруйні, хвороботворні, хімічні,

механічні і теплові.

Джерела забруднюючих речовин різноманітні, також багаточисельні види відходів і характер їхнього впливу на компоненти біосфери. Біосфера забруднюється твердими відходами, газовими викидами і стічними водами металургійних, металообробних і машинобудівних заводів. Величезної шкоди завдають водяним ресурсам стічні води целюлозно-паперової, харчової, деревообробної, нафтохімічної промисловості.

Розвиток автомобільного транспорту призвів до забруднення атмосфери міст і транспортних комунікацій важкими металами і токсичними вуглеводнями, а постійне зростання масштабів морських перевезень викликало майже повсюдне забруднення морів і океанів нафтою і нафтопродуктами.

## 6.3 Заходи щодо запобігання забруднення навколишнього середовища

Жорсткість вимог до виробництва й матеріалів, а також розробка нових виробничих й утилізаційних технологій дозволяє зменшити антропогенне навантаження на навколишнє середовище.

Заходи щодо запобігання забруднення навколишнього середовища:

1) збереження і раціональне використання земельних та водних ресурсів;
2) повторне використання ресурсів та ін.;
3) функціональне зонування, організація санітарно-захисних зон та санітарних розривів, озеленення та ін.;
4) технічна і біологічна рекультивація, нормалізація стану окремих компонентів навколишнього середовища тощо;
5) захисні заходи.

Висновки за розділом 6

У даному розділі проведено аналіз впливу небезпечних та шкідливих речовин, які потрапляють у середовище. Встановлені заходи щодо їх усунення та утилізації. Встановлені способи вирішення та запобігання впливу цих факторів на оточуюче середовище згідно до нормативних вимог.

# Висновки

У результаті виконання дипломної роботи був розроблений веб–додаток мультиплеєрної онлайн–гри «Dixit», який повністю відповідає вимогам поставленої задачі.

Були розроблені такі основні розділи: дослідження методів застосування веб–додатку у розробці онлайн–ігор, проектування рішень щодо веб–додатку мультиплеєрної онлайн–гри «Dixit», розробка проектних рішень щодо веб–додатку мультиплеєрної онлайн–гри “Dixit”. Було розроблено концепцію веб–додатку та проектних рішень щодо мультиплеєрної онлайн–гри «Dixit», розділ з економічного розрахунку вартості розробки та впровадження додатку. Розроблено розділи з охорони праці, охорони навколишнього середовища.

Веб–додаток був розроблений за допомогою сучасної мови програмування, та фреймворку, що надає змогу використовувати його головну особливість – реактивність, для швидшої та комфортнійшої роботи.

Веб–додаток мультиплеєрна онлайн–гра «Dixit», був протестований та готовий до впровадження у мережі Internet.

Список використаних джерел


1) Ковальов А.М. Монетизація веб-додатків [Текст] / А.М Ковальов , Н.В. Тендітна, О.В. Гайдаєнко //Інформаційні технології: моделі, алгоритми, системи (ITMAS – 2020): Всеукраїнська науково-практична інтернет конференція (26-28 жовтня 2020 р.) – Миколаїв: НУК ім. адмірала Макарова, 2020 – Режим доступу: http://itconf.nuos.edu.ua/2020/proceedings/
2) UML 2 і Уніфікований процес, практичний об'єктно-орієнтований аналіз і проектування./А. Нейштадт, Д. Арлоу. – М.: Символ-ПлюсМова 2017. – 617с.
3) Семакін І.Г. Основи алгоритмізації і програмування. Практикум: Учбове пос. для студ. закладів серед. проф. освіти / І.Г. Семакіін, А.П. Шестаков . – М.: ІЦ Академія, 2013. – 144 с.
4) Матеріали інформаційного сайту «React JavaScript–бібліотека для створення користувальницьких інтерейсів» [Електронний ресурс]. – Режим доступу: https://ru.reactjs.org/
5) Зиков С. В. Введення в теорію програмування. Курс лекцій. Учбовий посібник / С.В. Зиков. – М.: Інтернет–університет інформаційних технологій, 2012. – 400 с.
6) Семакін І.Г. Основи алгоритмізації і програмування: Підручник для студ. закладів серед. проф. освіти / І.Г. Семакін, А.П. Шестаков . – М.: ІЦ Академія, 2013. – 304 с.
7) Матеріали інформаційного сайту «Сучасний підручник JavaScript» [Електронний ресурс]. – Режим доступу: https://learn.javascript.ru/

Додаток А

# Інструкція користувача

Веб–додаток, який був розроблений в ході написання дипломної роботи, складається з двох екранних форм, які виконують функції: авторизації (вхід в систему) та безпорсередньо сторінка самої гри. Головна екранна форма веб–додатку мультиплеєрної онлайн–гри «Dixit» демонструє гру та форми взаємодії користувача з інформацію.

Для початку взаємодії необхідно відкрити веб–додаток, який показує вікно авторизації та виробу або створення ігрової кімнати, що представлено на (Рис. 1).

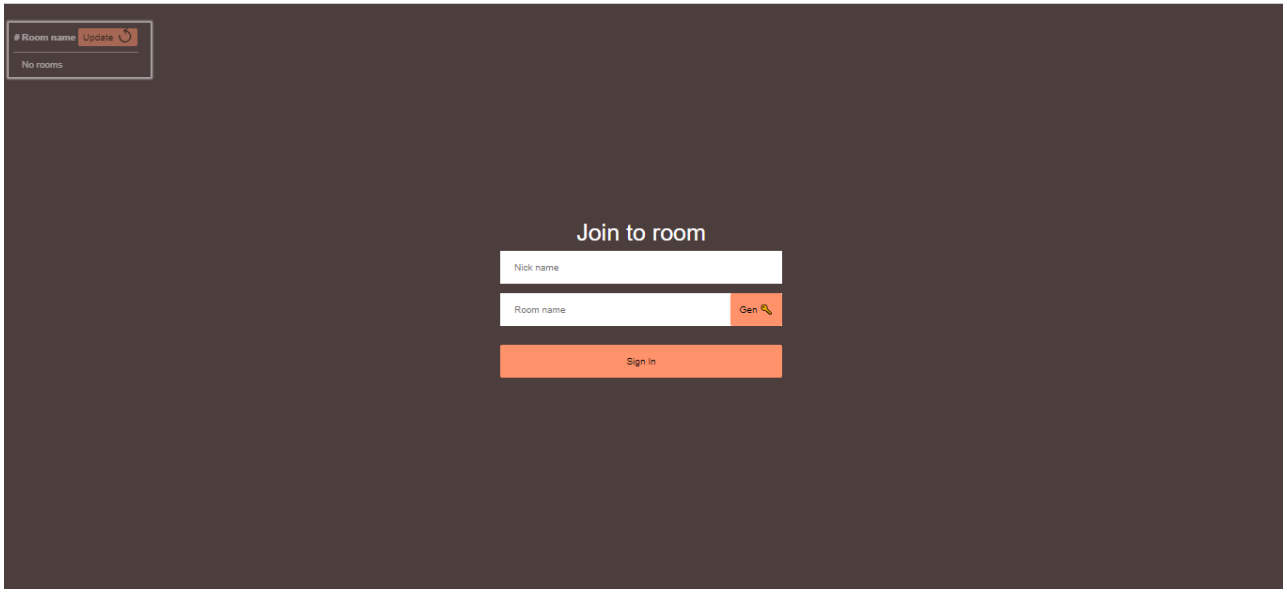



Рисунок 1 – Вікно авторизації виробу або створення ігрової кімнати

Далі користувач (гравець), потрапляє на сторінку гри, організатор (той хто створив кімнату–гру) обирає бажану кількість балів, яка буде визначати переможця та починає гру, коли під’єдналися інші гравці, всі гравці отримують по 5 різних карток, організатор починає гру загадуючи фразу–асоціацію та обираючи для неї картку. Сторінка гри представлена на (Рис. 2).

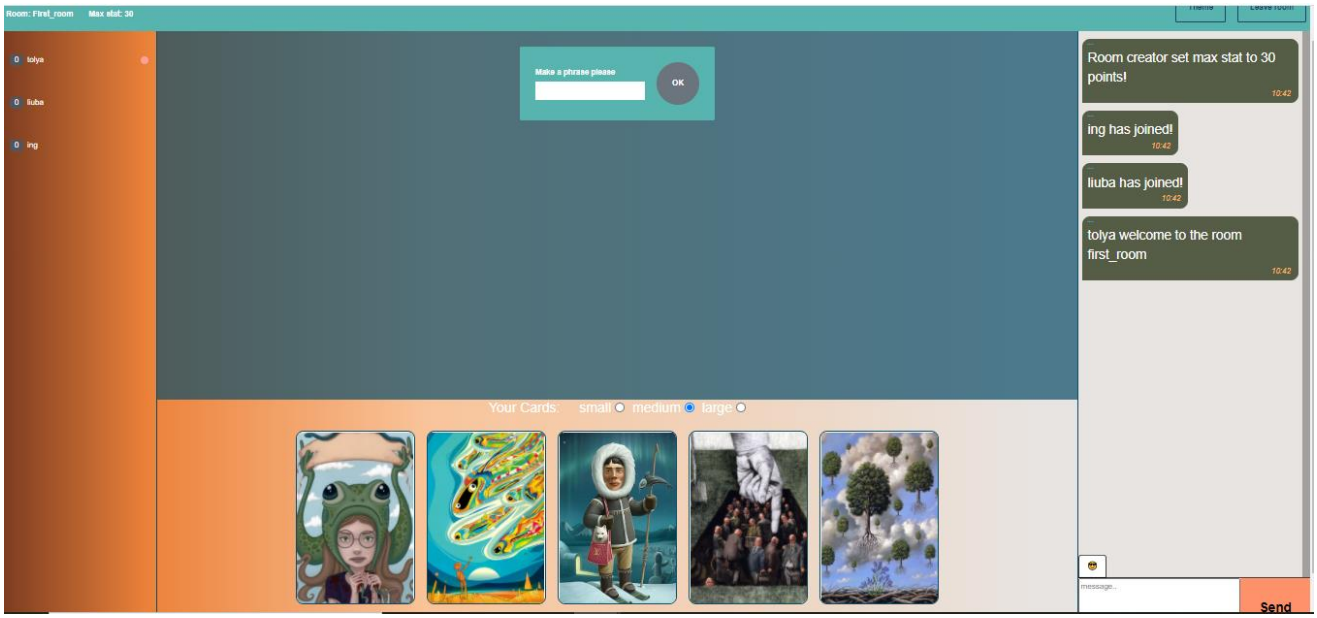



Рисунок 2 – Сторінка гри

На (Рис. 3) представлено чат, у якому гравці можуть спікуватися один з одним, також у цьому вікні виводяться дії гравців у грі.

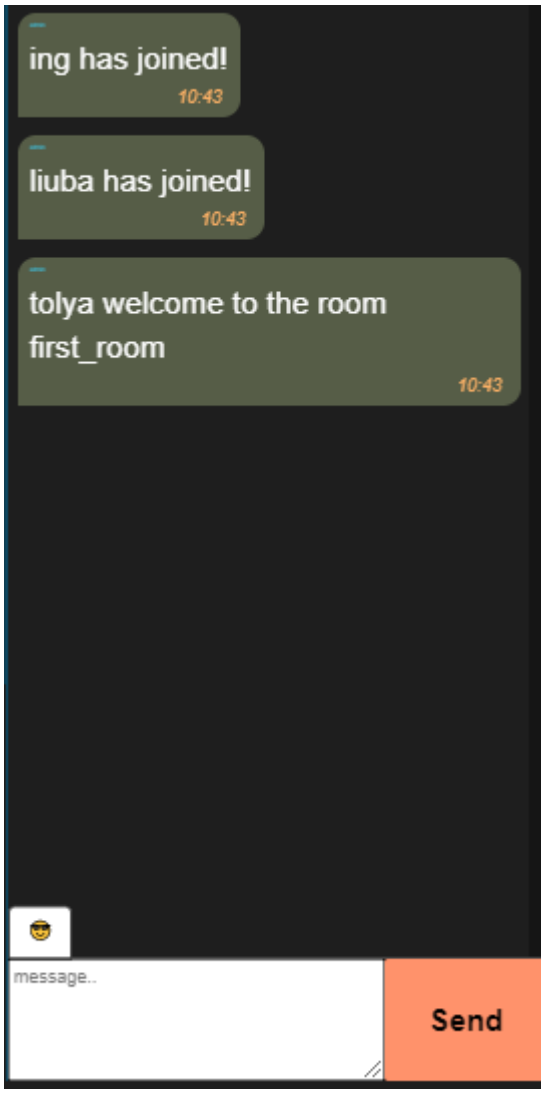



Рисунок 3 – Чат гри

На (Рис. 4) представлено форма з картками гравця. Для зручності була додана можливість обирати бажаний розмір карток.

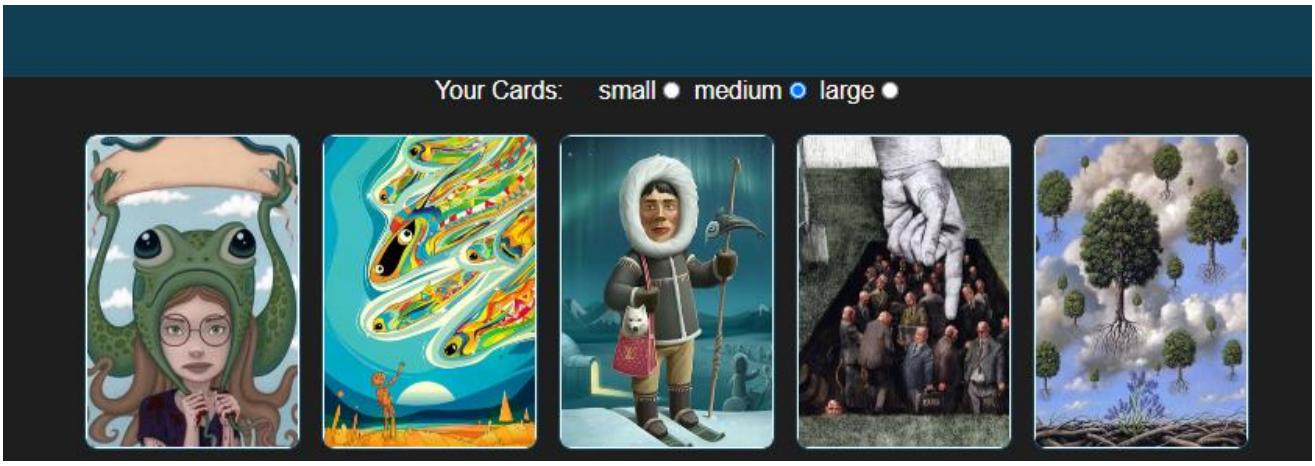



Рисунок 4 – Форма з картками

На (Рис. 5) представлений список гравців та їх балів у процесі гри.

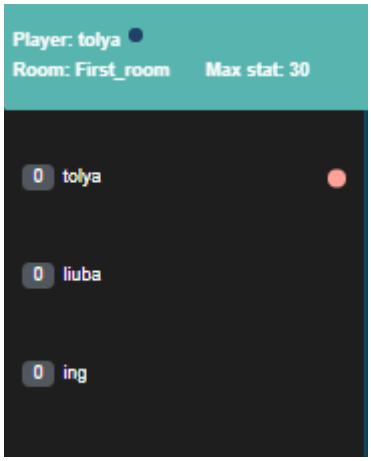



Рисунок 5 – Список гравців

На (Рис. 6) представлена сторінка гри у момент ходу гравця, коли потрібно задавати фразу та обирати картку, якщо гравець випадково забув обрати карту, йому показується нагадування.

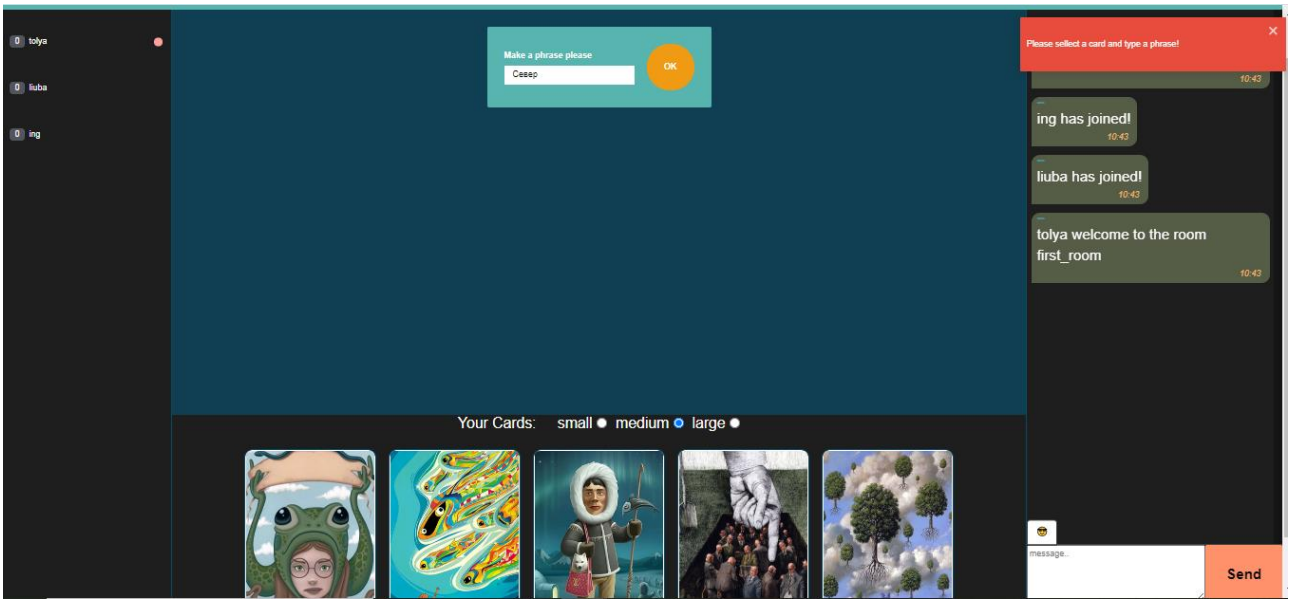



Рисунок 6 – Сторінка гри у момент ходу гравця

На (Рис. 7) представлена сторінка гри іншого гравця у момент вгадування карти по загаданій фразі. Гравці мають можливість обирати бажаний фон гри, для цього потрібно натиснути кнопку «Theme» у правому верхньому куті.

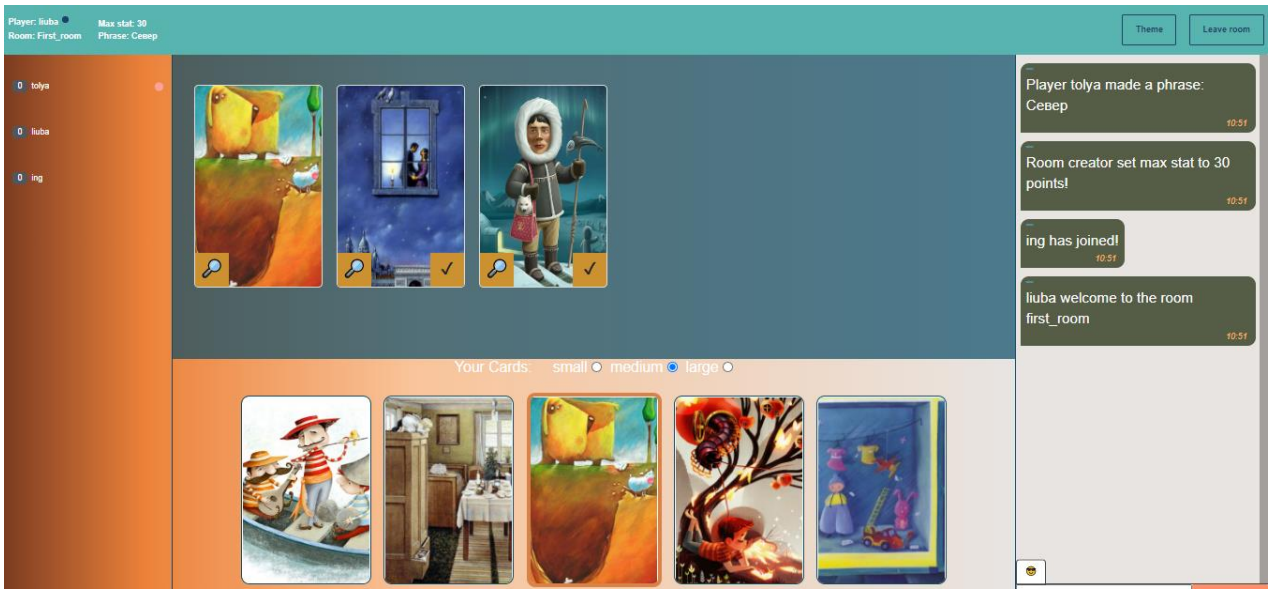



Рисунок 7 – Сторінка гри іншого гравця у момент вгадування карти

На (Рис. 8) представлена гра у момент закінчення ходу гравця, коли інші граці вже відгадали картку. З лівої сторони можна побачити як розподілиля набрані бали гравців.

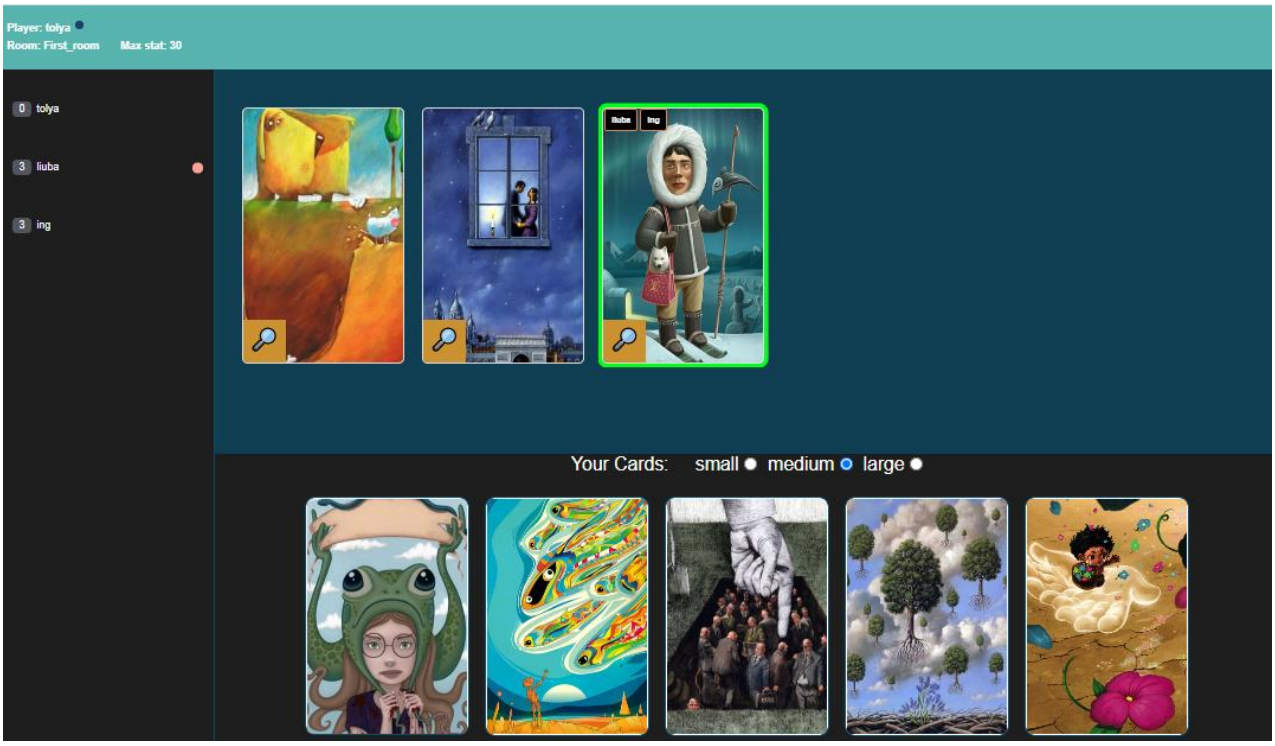



Рисунок 8 –Гра у момент закінчення ходу гравця

На (Рис. 9) представлений кінець гри у момент перемоги одного з гравців.


Рисунок 9 – Кінець гри у момент перемоги одного з гравців

ДОДАТОК Б

# Програма і методика випробувань веб–додатку мультиплеєрної онлайн–гри «Dixit»

## Б.1 Об’єкт випробування

Об’єктом випробування є веб–додатку мультиплеєрної онлайн–гри «Dixit».

Програмний продукт автоматизує процес гри у середовищі Internet, завдяки можливості легко підключатися до гри та взаємодіяти з іншими гравцями. Також автоматизовано процес комунікації між гравцями завдяки розробці чату.

## Б.2 Мета випробування

Метою випробування є перевірка веб–додатку мультиплеєрної онлайн–гри «Dixit» на наявність у ньому допущених помилок, а також відповідність реалізованих функцій тим, що зазначені у вимогах. Якщо рівень відповідності програмного забезпечення вимогам буде досягнутий, а також відсутні помилки та грубі недоліки, тоді веб–додаток приймається в експлуатацію.

## Б.3 Вимоги до програми

Під час проведення випробувань функціональні характеристики будуть перевірені на відповідність вимогам, викладеним у пункті «Вимоги до функціональних характеристик».

Розгянемо всі реалізовані функції додатку:

1. функція «Авторизації»;
2. функція «Створення кімнати–гри»;
3. функція «Чат гри»;
4. функція «Загадування фрази–асоціації та картки».

## Б.4 Вимоги до програмної документації

Документація, що пропонується до випробування програмного забезпечення, повинна включати наступні документи:

1. завдання та вимоги до розробки;
2. опис веб–додатку;
3. текст програмного коду;
4. програму та методику випробувань;
5. інструкцію користувача.

Б.5 Засоби та порядок випробувань

Для випробувань данного додатку необхідно, щоб на комп’ютері, де буде встановлене програмне забезпечення, була встановлена операційна система Microsoft Windows 7,8,10 та вище.

Для роботи додатку необхідна наявність у користувача зєднання з мережею інтернет, та персонального комп’ютера, що має характеристики, не нижче, ніж:

1. IBM–сумісний комп'ютер з тактовою частотою процесора 1.2 МГц і вище;
2. обсяг оперативної пам'яті не нижче 1 Гб;
3. монітор;
4. клавіатура;
5. миша.

При виявленні помилок у роботі програмного забезпечення складається перелік помилок і виправляється розробником.

Б.6 Методи випробування

Випробування будемо проводити за стратегією «чорного ящику».Програма випробування:

1.Випробування функції «Авторизації».

Заходимо на сайт гри, після чого потрапляємо на форму авторизації, ми маємо можливість обрати ім’я гравця. Заповнюємо поле «Name», та у лівому

верхньому куті копіюємо ключ до дійсної гри.  Натискаємо кнопку «Sing in». При виконанні всіх перерахований дій гравець повинен потрапити у гру.

2.Випробування функції «Створення кімнати–гри».

Заходимо на сайт гри, після чого потрапляємо на форму авторизації, далі потрібно задати імя гравця. Заповнюємо поле «Name», після чого у поле «Room name» вводиму назву нової кімнати–гри. Після успішного виконання перерахованих дій гравець повинен потрапити у щойно створену гру.

3.Випробування функції «Чат гри»

Заходимо на сайт гри, після чого потрапляємо на форму авторизації, потрібно обрати ім’я гравця, заповнюємо поле «Name» та у лівому верхньому куті копіюємо ключ до дійсної гри. Гравець повинен потрапити у гру, в якій з правої сторони знаходиться чат, программа попередньо повинна привітати нового гравця, це один з показників того, що чат працює, для повної перевірки треба ввести у вікно чата будь–яку фразу і надіслати її. У результаті успішного виконання всіх кроків у чаті повинна відобразитися введена фраза.

4.Випробування функція «Загадування фрази–асоціації та картки».

Заходимо на сайт гри, після чого потрапляємо на форму авторизації, потрібно обрати ім’я гравця, заповнюємо поле «Name» та у лівому верхньому куті копіюємо ключ до дійсної гри. Гравець повинен потрапити у гру. У свою чергу ходу, гравцю повинно відобразитися поле для вводу фрази. Після вводу бажаної фрази та вибору картки гравець повинен підтвердити свій вибір. У результаті успішного виконання всіх перерахованих дій всі гравці повинні побачити загадану фразу у вікні чату.

Додаток В

# Опис веб–додатку мультиплеєрної онлайн–гри «Dixit»

## В.1 Загальні відомості

Найменування програмного забезпечення: веб–додаток мультиплеєрна онлайн–гра «Dixit».

Для роботи з програмним забезпеченням необхідно:

1. Мати доступ до мережі інтернет;
2. Мати встановлену ОС Windows.
3. Або Телефон з можливістю виходу в інтернет.

## В.2 Функціональне призначення

Функціональне призначення розробки: веб–додаток мультиплеєрна онлайн–гра «Dixit» повинна надавати можливість гравцям під’єднуватися до гри з будь–якої точки світу та грати з гравцями у мережі інтернет. Завдяки автоматизованій функції додаток дозволяє гравцям спілкуватися безпосередньо у додатку.

## В.3 Опис логічної структури

У момент відкриття веб–додатку мультиплеєрної онлайн–гри «Dixit» з'являється вікно авторизації для входу у гру, що представлено на (Рис. 1):

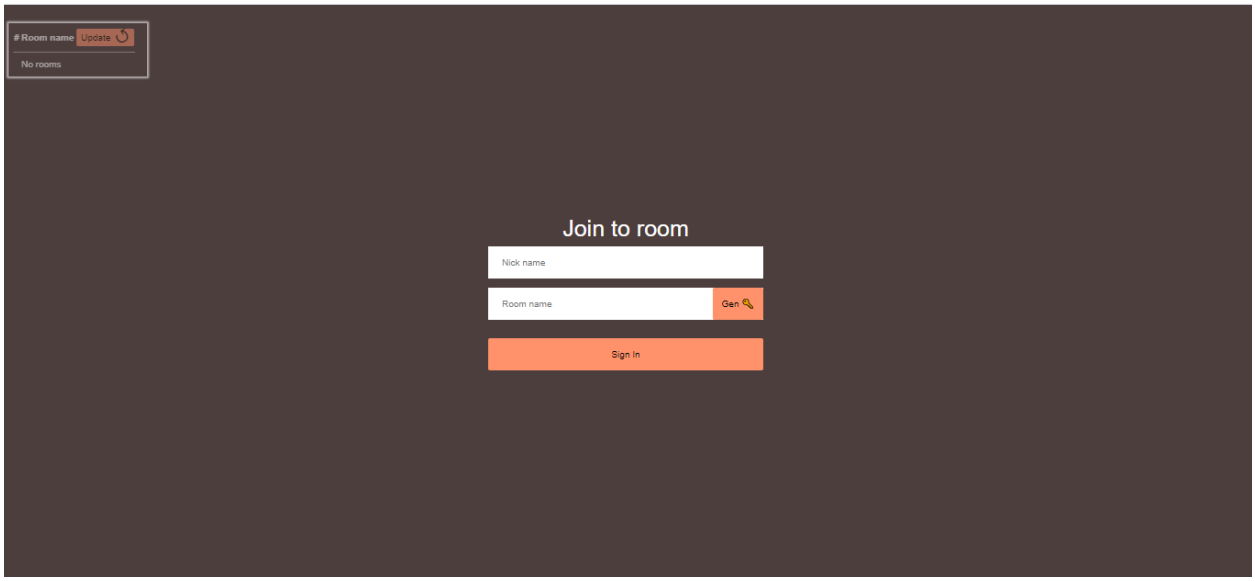



Рисунок 1 – Вікно авторизації

## В.4 Використані технічні засоби

Для реалізації програмного забезпечення використовувались:

1. мова JavaScript;
2. фреймворк ReactJS;
3. середа розробки VisualStudioCode.

## В.5 Виклик та завантаження

Для взаємодії з додатком у користувача повинен бути доступ до мережі інтернет. Для виклику додатку потрібно перейти за посиланням з назвою «https://dixit.com.ua».

## В.6 Вхідні дані

Вхідною інформацією для даної системи є :

- інформація про гравця, яка заповнюється для виведення у грі;
- інформація чату, для виведення у вікні чату;
- інформація про введену фразу і обрану картку.

В.7 Вихідні дані

Вихідною інформацією є перегляд карток, інформації чату та даних про гравців.

Програмний код додатку:

```
import { useEffect, useRef, useState } from "react";
import socketIOClient from "socket.io-client";

// require("socket.io-client/dist/socket.io.slim");
import store from "../../redux/store/store";
import { setUser, setMyCards } from "../../redux/actions/userActions";
import {
  setMaxStatAction,
  setCurrentTurnPlayer,
  setPhrase,
  setCardsOnTable,
  setAllowAddCardOnTable,
  setAllowSeeCardsOnTable,
  setVotedPlayers,
  setAllowSeeVoteResults,
  updatePlayersStats,
} from "../../redux/actions/roomActions";
import { toast } from "react-toastify";
import { renderWinnersModal } from "../../utils/helpers";

const useRoom = (props) => {
  const [messages, setMessages] = useState([]);
  const [players, setPlayers] = useState([]);
  const socket = useRef();
  const room = store.getState().room.name;
  const name = store.getState().user.name;
  const { newStats } = store.getState().room;
```

```
const ENDPOINT =
  window.location.hostname == "localhost"
    ? "http://localhost:5000"
    : "https://app-dixit.herokuapp.com/";

useEffect(() => {
  let currentPlayersData = players.map((playerData) => {
    newStats.forEach((statData) => {
      statData.userName == playerData.name &&
        (playerData.stats = statData.stats);
    });
    return playerData;
  });
  setPlayers(currentPlayersData);
}, [newStats]);

useEffect(() => {
  socket.current = socketIOClient(ENDPOINT);
  socket.current.emit("join", { name, room }, (user) => {
    if (user.error) {
      toast.error(user.error, {
        position: toast.POSITION.TOP_RIGHT,
        hideProgressBar: true,
        autoClose: 4000,
        // onClose: () => (location = "/"),
      });
    } else store.dispatch(setUser(user));
  });

  socket.current.on("message", (message) => {
```

```
  setMessages((messages) => [message, ...messages]);
});

socket.current.on("setplayerturn", (userName) => {
  store.dispatch(setCurrentTurnPlayer(userName));
});

socket.current.on("roomData", (usersInRoom) => {
  setPlayers([...usersInRoom, ...players]);
});

socket.current.on("allowseevoteresults", (allowSeeVoteResults) => {
  store.dispatch(setAllowSeeVoteResults(allowSeeVoteResults));
});

socket.current.on("updateStats", (turnResult) => {
  store.dispatch(updatePlayersStats(turnResult));
  getMyCards();
});

socket.current.on("addcardontable", (cardsOnTable) => {
  store.dispatch(setCardsOnTable(cardsOnTable));
});

socket.current.on("allowaddcardontable", (allow) => {
  store.dispatch(setAllowAddCardOnTable(allow));
});

socket.current.on("allowseecardsontable", (allow) => {
  store.dispatch(setAllowSeeCardsOnTable(allow));
```

```
});

socket.current.on("setmaxstat", (stat) => {
  store.dispatch(setMaxStatAction(stat));
  getMyCards();
});

socket.current.on("setphraseforturn", (madePhrase) =>
  store.dispatch(setPhrase(madePhrase))
);

socket.current.on("voteforcard", (votedPlayers) => {
  store.dispatch(setVotedPlayers(votedPlayers));
});

socket.current.on("disconnect", () => {
  // location = "/";
  // socket.current.connect();
  console.log("DISCONNECT");
});

socket.current.on("ping", (e) => {
  // console.log(e);
});

socket.current.on("grh", (data) => {
  console.log(data);
});

socket.current.on("gameisover", (winners) => {
```

```
    return renderWinnersModal(winners);
  });

  return () => {
    socket.current.disconnect();
    socket.current.emit("disconnect");
    socket.current.off();
  };
}, []);

const sendMessage = ({ message }) => {
  socket.current.emit("sendMessage", message, () => {});
};

const setMaxStat = (stat) => {
  socket.current.emit("setmaxstat", stat, () => {});
};

const setPhraseforTurn = (phrase) => {
  socket.current.emit("setphraseforturn", phrase, () => {});
};

const voteForCard = (cardData) => {
  socket.current.emit("voteforcard", cardData);
};

const getMyCards = () => {
  socket.current.emit("getusercards", (myCards) => {
    store.dispatch(setMyCards(myCards));
  });
```

```
  };

  const playCardOnTable = (card) => {
    socket.current.emit("addcardontable", card, () => {});
  };

  return {
    messages,
    sendMessage,
    players,
    setMaxStat,
    setPhraseforTurn,
    playCardOnTable,
    getMyCards,
    voteForCard,
  };
};

export default useRoom;
```