**ЗАВДАННЯ**

**АНОТАЦІЯ**

В даній дипломній роботі розроблено туристичний веб-портал «GuideMe». Інформаційна система розроблена згідно архітектури клієнт-сервер. Клієнт веб-ресурсу може експлуатуватися на будь-якій операційній системі: основною вимогою є наявність веб-браузера. Для серверної частини веб ресурсу необхідним є наявність бази даних MySQL 5.6 та веб-сервера Apache Tomcat 8. Проект розроблений за допомогою технологій Java EE, що дозволяє використовувати на сервері будь-яку з операційних систем, для яких існує реалізація Java VM.

Продукт пропонує наступні функціональні можливості:

* Пошук події з різними фільтрами;
* Долучення до подій;
* Коментування профілів друзів і подій;
* Оцінювання профілів користувачів та подій;
* Створення власних подій;
* Для гідів - можливість вказати список власних послуг;
* Замовлення додаткових послуг гідів користувачами;

Основною ідеєю проекту було створення ресурсу, який би допомагав людям з різних куточків землі знаходити собі компанію для проведення спільного відпочинку в різних містах, а також в різних країнах. Особливістю проекту є те, що він розроблений у вигляді своєрідної соціальної мережі, де користувачі мають змогу додавати один одного в друзі, спілкуватись в чаті, ставити один одному оцінки та залишати в профілях друзів та подій коментарі.

Обсяг роботи складає 108 сторінок.

**ABSTRACT**

In this diploma work was developed web portal named «GuideMe». The information system developed under client-server architecture. Client can be used on any operating system, the basic requirement is to have a web browser. For server side it required MySQL 5.6 database and web server Apache Tomcat 8. The project was developed using Java EE technologies. That allows to use on server any of the operating systems for which there is implementation of Java VM.

The product offers the following functionality:

• Search for events with different filters;

• Joining to events;

• Commenting profiles of friends and events;

• Evaluation of user profiles and events;

• Create your own events;

• Ability to specify a list of their services for guide;

• Ordering additional services;

The main idea of the project was to create a resource that would help people from different corners of the earth to find a company for leissure in different cities and in different countries. The main feature of the project is that it designed as a of social network where users can add each other to friends, chat, ask each other to leave the assessment for their profiles of and events, comment their users and events profiles.

The capacity of work is 108 pages.

**ЗМІСТ**

ВСТУП……………………………………………………………………..….…......7

1. РОЗДІЛ 1. ОГЛЯД ТУРИСТИЧНИХ ТА ВІДПОЧИНКОВИХ ВЕБ-РЕСУРСІВ……………………………………………………………………..……..8

1.1. Ексурсійні веб-сайти м.Львова.......................................................................8

1.2. Веб-ресурси для бюджетного відпочинку...................................................12

2. РОЗДІЛ 2. ПОСТАНОВКА ЗАДАЧІ ТА СПЕЦИФІКАЦІЯ ВИМОГ ТУРИСТИЧНОГО ВЕБ-ПОРТАЛУ “GUIDEME”..................................................19

2.1. Змістовна постановка задачі..............................................................................19

2.2. Специфікація вимог до туристичного веб-порталу “GuideME”....................24

2.3. Вибір технологів та інструментів.....................................................................31

3. РОЗДІЛ 3. ПРОЕКТУВАННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ТУРИСТИЧНОГО ВЕБ-ПОРТАЛУ “GuideME……………………….................35

3.1. Використані шаблони проектування................................................................35

3.2. Проектування бази даних та основних класів.................................................39

3.3. Проектування користувацьких інтерфейсів....................................................45

4. РОЗДІЛ 4. ТЕСТУВАННЯ ТА СУПРОВІД ВЕБ-ПОРТАЛУ “GuideME”……………………………………………………………………..........51

4.1. Звіт про тестування............................................................................................51

4.2. План впровадження............................................................................................53

4.3. План супроводу..................................................................................................57

5. РОЗДІЛ 5. ЕКОНОМІЧНА ОЦІНКА ПРОЕКТНОГО РІШЕННЯ……....…...61

5.1. Економічна характеристика проектного рішення……………….…….……61

5.2. Розрахунок витрат на розробку інтелектуальної інформаційної системи…………………………………………………………………………......63

5.3. Визначення комплексного показника якості………………………….…….68

5.4. Визначення експлуатаційних витрат……………………………….……......70

5.5. Розрахунок ціни споживання проектного рішення…………….…………...73

5.6. Визначення показників економічної ефективності………….………….…..74

ВИСНОВКИ..............................................................................................................77

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ...........................................................78

Додаток А. Зразки програмного коду проекту…………………………………..80

Додаток Б. Концептуальна діаграма класів Туристичного веб-порталу……….92

Додаток В. Діаграма прецендентів ……………………………………………...93

Додаток Г. Діаграма діяльності ……………………………………………….....94

Додаток Д. Інструкція користувача………………………………………………95

Додаток Е. Результати функціонального тестування………………………….100

Додаток Є. Результати тестування на безпеку………………………………….106

**ВСТУП**

Основними задачами, з якими стикається людина, яка має намір здійстити подорож для проведення якісного та безпечного відпочинку є: обрати місце подорожі, знайти компанію для подоророжі і звичайно вкластись в бюджет, запланований для даної подорожі. Знайти інформацію про цікаві місця для подорожі в різних місцях країни проживання, а також за кордоном не буде проблемою, оскільки в інтернеті є багато інформаційних ресурсів, які у цьому допоможуть.

Якщо користувач хоче здійстити подорож в інше місто чи країну, йому потрібно потурбуватись про місце проживання, якщо ж звичайно ця подорож триватиме більше одного дня. Також, якщо людина є прихильником екскурсійного відпочинку, то потрібно знайти професійного гіда, який би надав дані послуги. Якщо нема в наявності коштів, щоб оплатити проживання в готелі та послуги гіда, можна спробувати знайти друзів в локації куди планується їхати, щоб вони допомогли з житлом і провели екскурсію по визначних місцях населенного пункту, де вони проживають. Незалежно який з варіантів обрано, користувачу буде цікаво почитати відгуки про професійність гіда чи про особисті якості людини до якої їдуть в гості.

У дипломному проекті запропоновано вирішення попередньо описаних проблем у веб-ресурсі під назвою “GuideMe”.

**РОЗДІЛ 1. ОГЛЯД ТУРИСТИЧНИХ ТА ВІДПОЧИНКОВИХ ВЕБ-РЕСУРСІВ**

* 1. **Ексурсійні веб-сайти м. Львова**

Розглянемо популярні веб-сайти Львова, які послужать в допомозі для проведення відпочинку. Один з найпопулярніших - <http://www.justlviv.it/> (рис. 1.1). На цьому сайті можна отримати інформацію про:

* Тури та екскурсії;
* Хостели та готелі;
* Гідів Львова;
* Побачити розташування цікавих місць Львова на мапі.

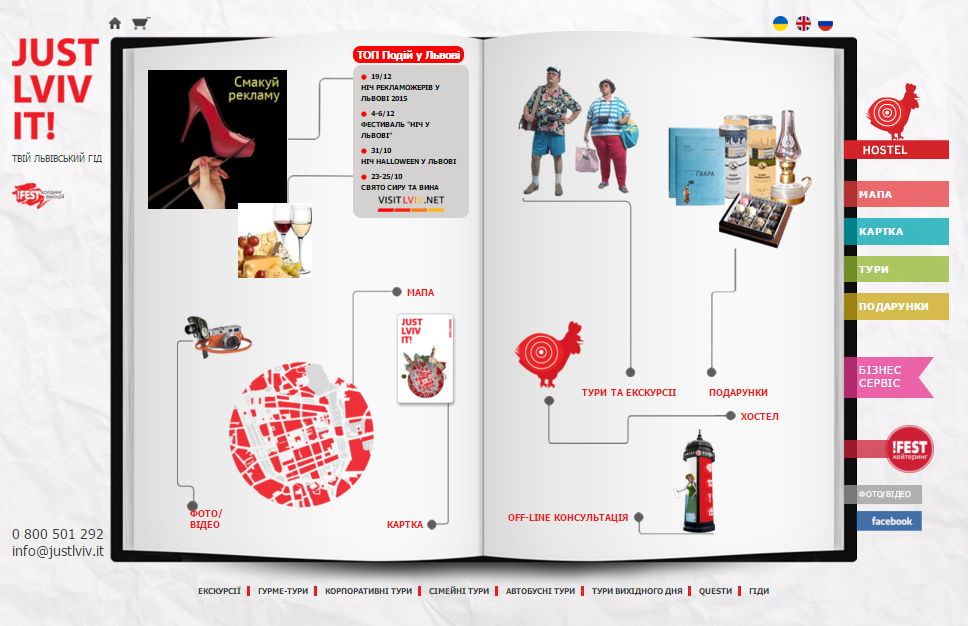


Рис. 1.1. Веб-сайт «Just Lviv It!»

Ще одним цікавим ресурсом для планування відпочинку у Львові є сайт <http://lviv.travel/> (рис. 1.2). На цьому сайті зібрана інформація про фестивалі та концерти, які на даний момент відбуваються чи в найближчому майбутньому відбудуться у Львові. Також на даному ресурсі користувач зможе знайти перелік архітектурних цінностей, які на думку авторів слід відвідати перебуваючи у Львові. Перейшовши на деталізовану сторінку певної архітектурної цінності користувач побачить історичні відомості про неї, а також розташування цього місця на карті.

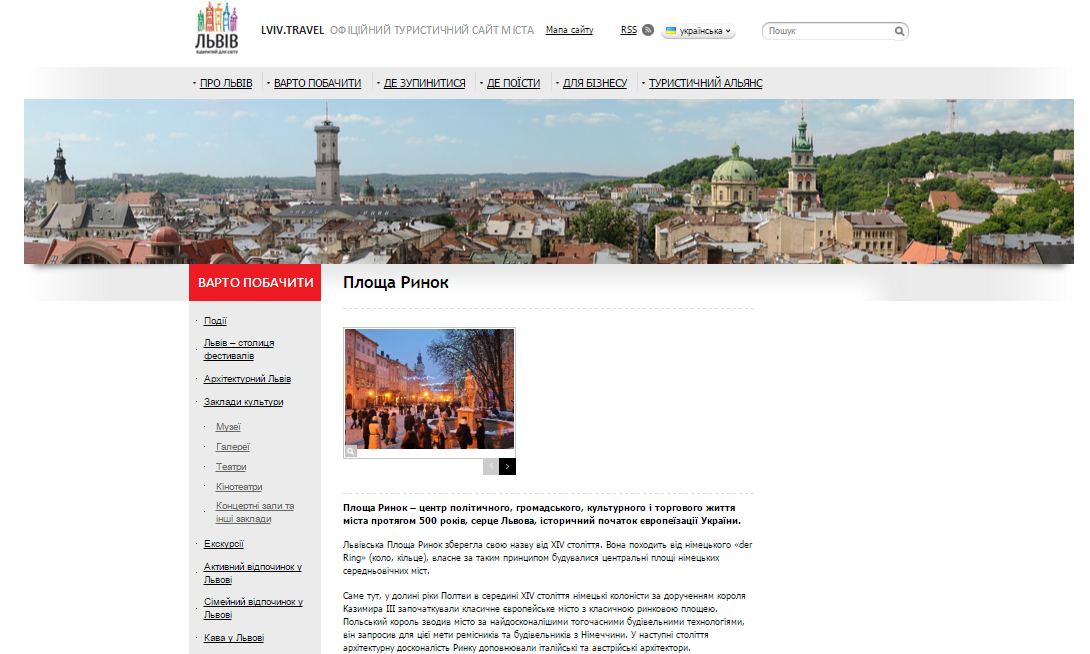


Рис. 1.2. Веб-сайт «Lviv Travel»

Також тут користувач знайде інформацію про різні види відпочинку у місті, згруповані по наступним категоріям:

* Активний відпочинок у Львові;
* Сімейний відпочинок у Львові;
* Кава у Львові;
* Пиво у Львові;
* Галицька кухня;
* Нічне життя у Львові;
* Шопінг у Львові;
* Довкола Львова.

Ще одним з популярним відпочинковим ресурсом є туристичний портал <http://ekskursiya.com.ua/> (рис. 1.3). Даний портал представляє собою сайт, на якому користувач зможе вибрати екскурсію на свій смак з достатньо широкого переліку екскурсій, які пропонує даний веб-портал. Автори сайту так описують свій проект: «Туристичний портал "Екскурсія" - це проект, що об'єднує професіоналів своєї справи у сфері туризму Західною Україною. До нашого колективу входять спеціалісти в сфері надання туристичних послуг - організатори, екскурсоводи, перевізники. Ми займаємось організацією екскурсій та подорожей по Львову, по Львівській області та Заході України. А у майбутньому ми плануємо географічно розширити контингент наших послуг по території України. Ми співпрацюємо з досвідченими готельєрами та рестораторами, тому усе це дозволяє нам забезпечити Вам гарний відпочинок»

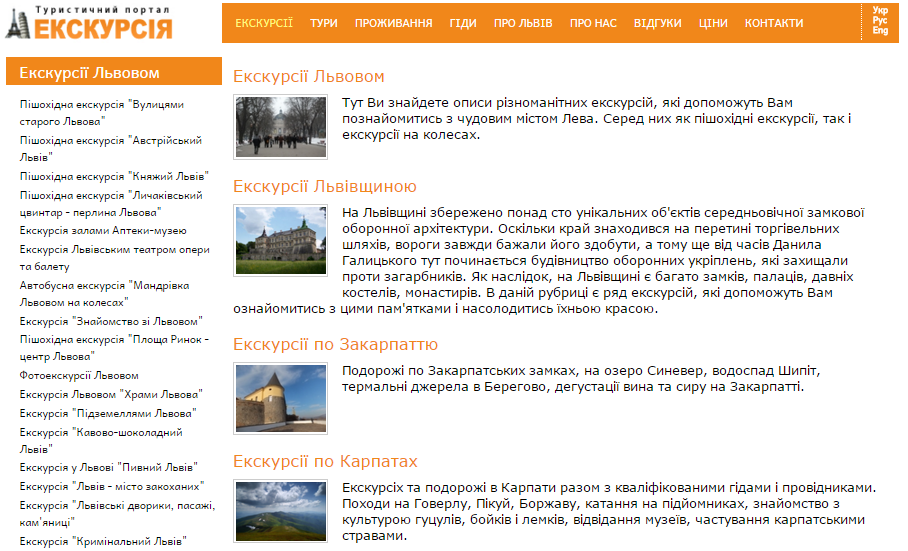


Рис. 1.3. Туристичний портал «Екскурсія»

Ресурс <http://gui-de.lviv.ua/> пропонує піші та автомобільні екскурсії, лижні, а також велисопедні тури (рис. 1.4):



Рис. 1.4. Туристичний портал «Екскурсія»

Також серед інтернет простору Львівщини зустрічаються персональні сайти-візитки гідів, де можна забронювати собі екскурсію визначними місцями, наприклад <http://www.lviv4you.com/> (рис. 1.5):

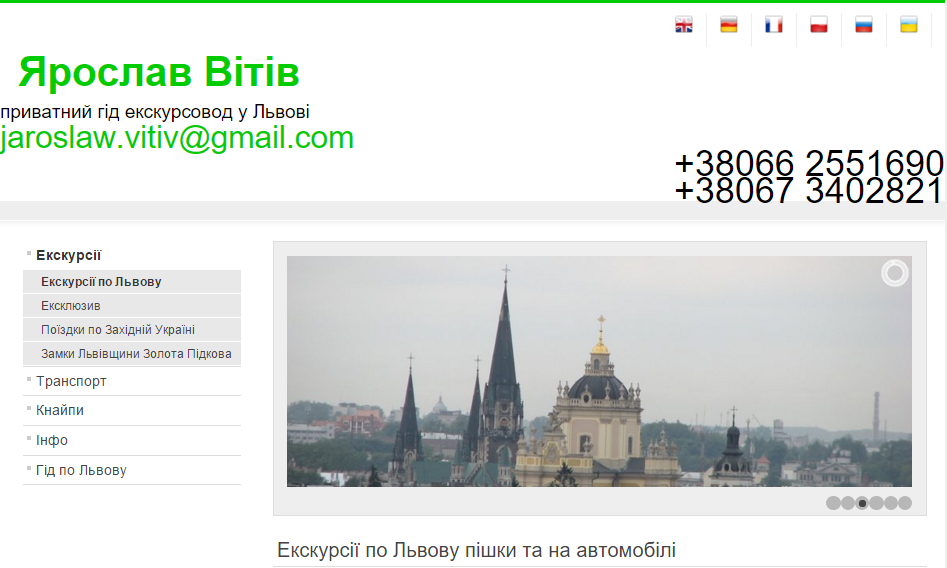


Рис. 1.5. Сайт-візитка приватного гіда

Вивчивши розважально-туристичні веб-ресурси Львівщини, можна зробити висновок, що здебільшого вони носять суто інформаційний характер. Багатофунціональний проект, який би містив можливості як для замовлення професійних гідів, так і соціальної мережі для організації бюджетного відпочинку міг би користуватися попитом, адже туристичний ринок Львівщини активно розвивається.

В наступному пункті розглянемо, які ресурси пропонує нам всесвітня мережа інтернет для бюджетного відпочинку.

* 1. **Веб-ресурси для бюджетного відпочинку**

Не потрібно витрачати багато грошей, щоб подорожувати, оскільки в даний час, можливо поїхати майже в будь-яке місце, при обмеженому бюджеті. В даному розділі розглянуто деякі сайти, які допоможуть туристам досягти місця призначення, навіть без наявності у них великої кількості грошей на відпочинок.

**Couchsurfing**

Couchsurfing це мережа для справжніх мандрівників та господарів житла, які хочуть, зустрітися з людьми з усього світу і поділитися своїм досвідом і культурою (рис. 1.6). Якщо мандрівник хоче заощадити гроші на проживання, Couchsurfing може бути веб-ресурсом саме для цього. Зареєструватися і почати пошук власників житла, що пропонують безкоштовне проживання, дуже просто. Можна знайти попутників або місцевих жителів, щоб проводити відпочинок з ними. Якщо мандрівник вагається, чи ночувати в незнайомих людей, то йому не слід цього боятися. Лише той, хто в повній мірі заповнив профіль може користуватися даним ресурсом. Спільнота couchsurfers організовує різні заходи, час від часу.

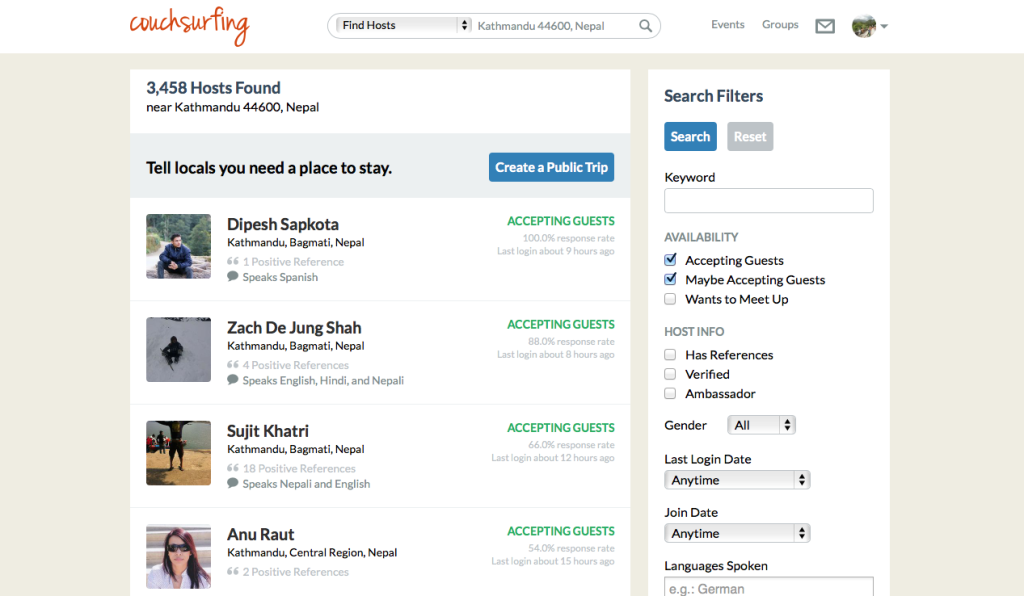


Рис. 1.6. Веб-сайт Couchsurfing

**HitchWiki**

Якщо мандрівник є шукачем пригод і хоче спробувати автостоп – то HitchWiki є сайтом, де автостопщики усього світу діляться своїми знаннями про подорожі (рис. 1.7). Так як це не найбезпечніший спосіб подорожувати, то краще прочитати кілька порад досвідчених людей. Користувач дізнається, як збільшити свої шанси на отримання поїздки, як залишатися в безпеці, і те, як ставляться до автостопу в конкретних країнах. Якщо у користувача є які-небудь питання, то можна приєднатися до дискусійної групі й попросити більш досвідчених учасників їхньої поради.



Рис. 1.7. Веб-сайт HitchWiki

**Momondo**

Якщо туристу потрібно авіаквиток, можна скористатися ресурсом Momondo (рис. 1.8). Перш за все, користувач зможе отримати велику вигоду. Але якщо він просто хоче подорожувати, і не знає, куди, Momondo вирішить також цю проблему. Слід вказати ліміт грошей, аеропорт і користувач отримає список всіх місць, де зможе полетіти в рамках його бюджету. При натисканні на місті зацікавлення, Momondo забезпечить загальну інформацію про місце, прогноз погоди, пропозиції, і ідеї для додаткових поїздок від обраного місця. Щоб заощадити трохи грошей, надаються поради про найкращий момент, коли варто купити квиток.

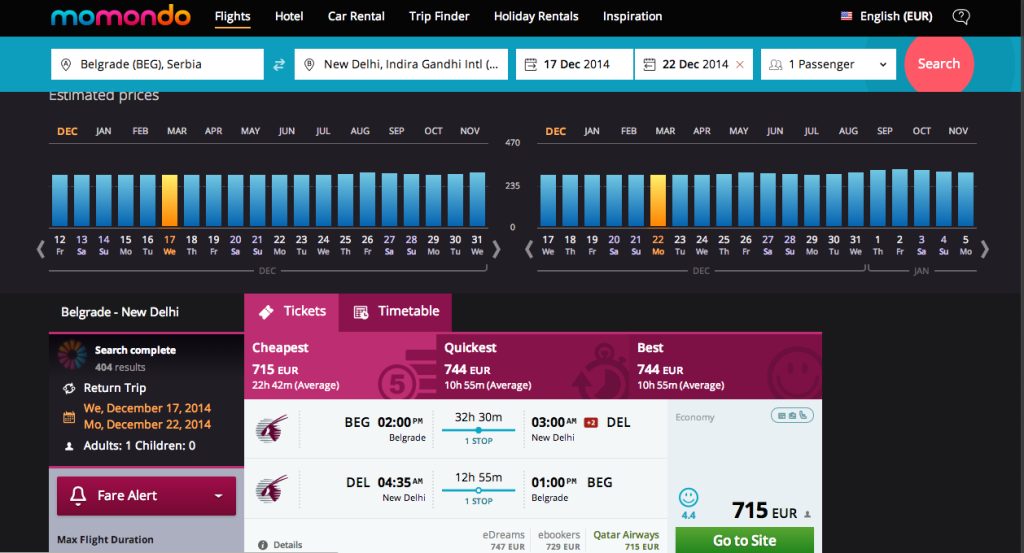


Рис. 1.8. Веб-сайт Momondo

**Staydu**

Staydu (рис. 1.9) - це мережа для бюджетних мандрівників, які потребують місце для проживання, і господарів, які люблять, зустрічати нових людей з різних країн. Мандрівник зможе отримати житло від господаря безкоштовно, за невелику плату. Якщо потрібен приятель для подорожі, можна зареєструватися і поділитися своїми планами. Єдиний недолік доволі низька база користувачів в порівнянні з більш сильними сайтами, як Couchsurfing.

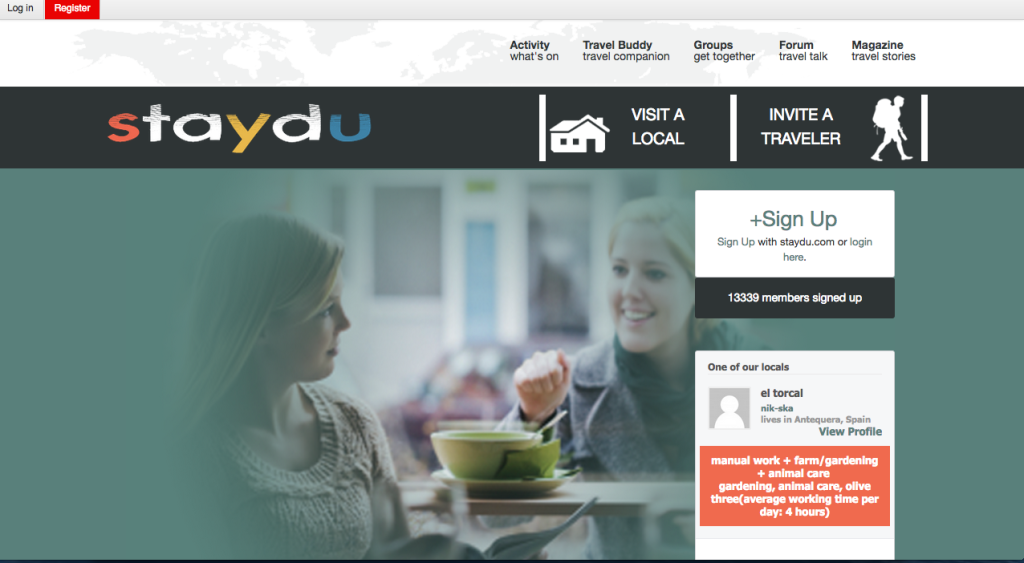


Рис. 1.9. Веб-сайт Staydo

**Last Minute**

Якщо мандрівник належить до групи людей спонтанних, і не хоче довго шукати найкращу ціну, а почати свою подорож відразу, на *Lastminute.com* (рис. 1.10), можна знайти пропозиції в останню хвилину: авіаквитки, квитки на поїзд, бронювання готелів в одній базі даних. Знижки на курорти, театри, ресторани які надаються у Великобританії, США і деяких країнах Західної Європи.

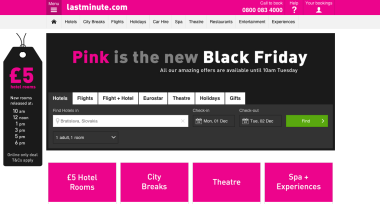


Рис. 1.10. Веб-сайт Last Minute

**Stay.com**

*Stay.com* (рис. 1.11) має близько 100 пунктів призначення, де можна створити свій особистий маршрут. Користувач обирає своє розташування і місця де хоче побувати. Є можливість завантажити свій маршрут у форматі PDF і роздрукувати його, або завантажити їх в додаток для IOS та Android, і використовувати його. Можна змінити список в будь-який час і додавати нові місця. Користувачу не потрібно купувати карти з позначенням визначних місць. Додаток працює в автономному режимі і безкоштовно.



Рис. 1.11. Веб-сайт Stay.com

**Hipmunk**

Політ в літаку з 2-ма або 3-ма зупинками тривалістю кілька годин може стомлювати туристів. *Hipmunk* (рис. 1.12) є пошуковою системою для авіаквитків, яка порівнює пропозиції від авіакомпаній та туристичних сайтів. Тут можна шукати авіаквитки відповідно до ціни, тривалості, зняття, посадки.

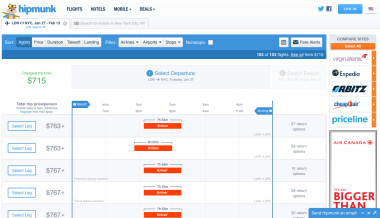


Рис. 1.12. Веб-сайт Hipmunk

**Workaway.info**

Дізнатися щось нове, допомогти комусь, познайомитися з місцевими жителями, подорожувати і отримати безкоштовне проживання та харчування можна реалізувати з допомогою ресурсу workaway.info (рис. 1.13). Тут можна знайти робочі місця в більш ніж 135 країнах по всьому світу. Стандартна ставка волонтер 5,00 доларів в день протягом 5 днів на тиждень. Також є можливість допомогти сім’ям або невеликим організаціям. Можливо проживати з місцевими жителями і набути нових навичок, які зроблять подорож цікавою.

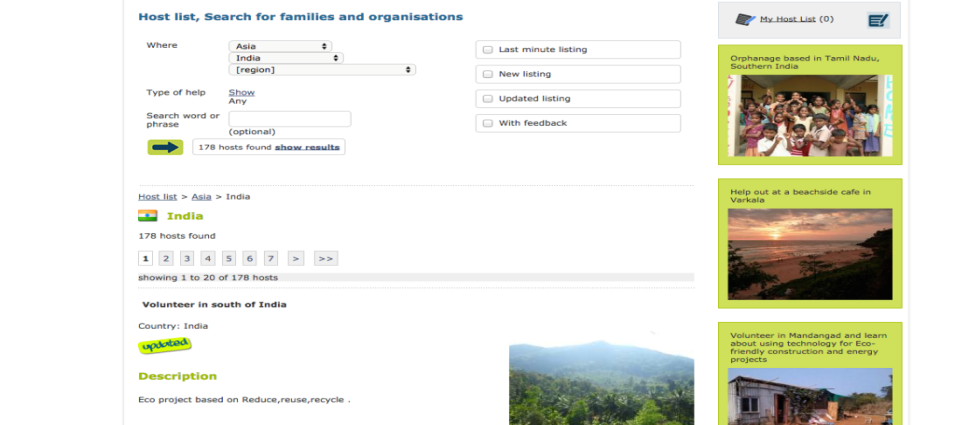


Рис. 1.13. Веб-сайт Workaway.info

**Trivago**

*Trivago* (рис. 1.14) є найбільшим пошуковиком готелів у світі, яка порівнює пропозиції з близько 238 сайтів бронювання. Тут користувач може встановити критерії, важливі для нього, як бюджет або розмір готелю чи відстань до центру міста.

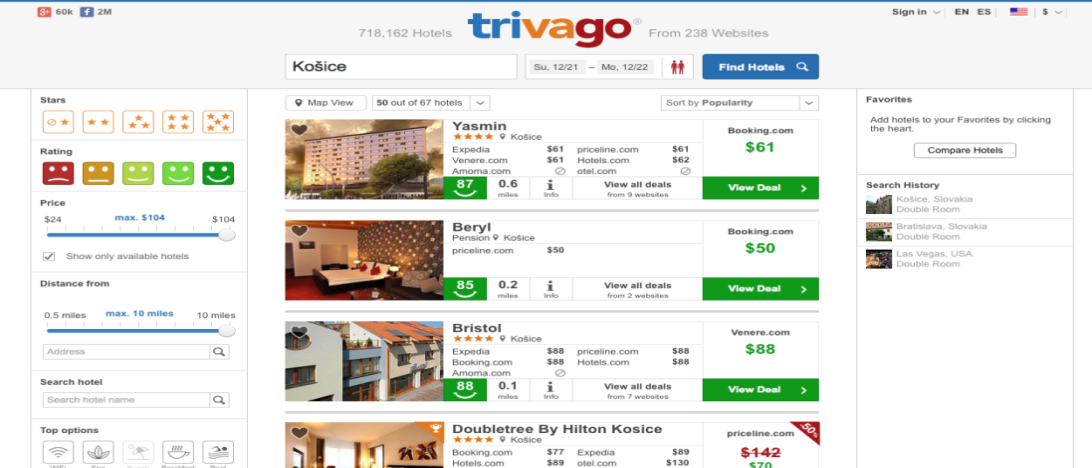


Рис. 1.14. Веб-сайт Trivago

**Bla Bla Car**

Bla Bla Car(рис. 1.15) досить популярний сайт у Великобританії і Європі, набуває популярності останнім часом і в Україні, обслуговуючи майже 10 мільйонів користувачів. Крім, знаходження дешевих перевезень, є можливість познайомитися з цікавими людьми. Кожен користувач має свій профіль, рейтинги, і можна побачити, яким типом автомобіля вони керують. Це схоже на безпечний варіант автостопу.

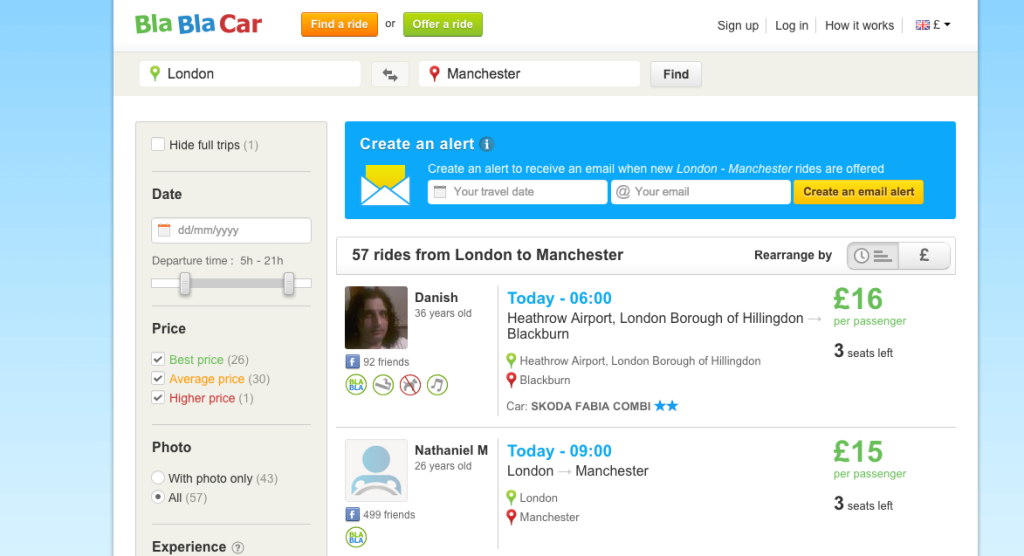


Рис. 1.2.10. Веб-сайт Bla Bla Car

Більшість з розглянутих варіантів ще не є популярними в Україні і у Львові загалом, тому можна зробити висновок, що веб-ресурс з функціоналом для бюджетного туризму, міг би знайти тут свою нішу, чому би сприяли умови не високої конкуренції.

**РОЗДІЛ 2. ПОСТАНОВКА ЗАДАЧІ ТА СПЕЦИФІКАЦІЯ ВИМОГ ТУРИСТИЧНОГО ВЕБ-ПОРТАЛУ “GUIDEME”**

**2.1. Змістовна постановка задачі**

Основною ідеєю проекту – є створити ресурс, який би допомагав людям з різних куточків землі знаходити собі компанію для проведення спільного відпочинку в різних містах, а також в різних країнах.

На даному сайті користувачі матимуть можливість, як створювати власні події, так і долучатись до подій створених іншими користувачами. Одним із видом подій є екскурсії. Для того, щоб створити екскурсію, користувач повинен спочатку зареєструватися в системі, як професійний гід. Цю реєстрацію повинен затвердити адміністратор сайту і тільки після цього користувач зможе створювати власні екскурсії, а також вказувати додаткові послуги та сервіси, які можуть на цій екскурсії надаватись. Наприклад, користувач збирається на екскурсію в інше місто чи країну, а ця екскурсія має тривати декілька днів і відповідно йому потрібно знайти житло в готелі, але він цим пошуком не має бажання займатися, то можна обрати відповідну опцію на сторінці екскурсії і тим займеться його гід. Вартість замовлених послуг відображається на сторінці екскурсії і користувачу залишається тільки приїхати на екскурсії і оплатити своє замовлення. Або можлива ситуація, коли гід окрім основної екскурсії може додатково запропонувати якийсь додатковий тур, тоді інформацію про це він може опублікувати на сторінці екскурсії і його клієнти, зможуть цей сервіс замовити.

Якщо ж користувач не має можливості чи бажання їздити на платні екскурсії – він може обрати інший шлях. Знайшовши подію, яку створив інший користувач, він може до неї долучитися. На сторінці даної події він зможе знайти інформацію про те, чи хтось з учасників має змогу безкоштовно надати житло і таким чином вирішити проблему з проживанням.

Також слід зазначити, що проект слід оформити у вигляді своєрідної соціальної мережі, де користувачі мають змогу додавати один одного в друзі, спілкуватись в чаті, ставити один одному оцінки та залишати в профілях друзів та подій коментарі.

Також функціонал сайту потрібно доповнити різноманітними можливостями для його модерації та адміністрування.

Отже основними цілями нашого веб-ресурсу буде:

* Стати ресурсом, який буде допомогати людям організувати власні події і приєднуватися до подій, які створюють інші користувачі, планувати спільні поїздки;
* Допомагати мандрівникам в пошуку місця проживання;
* Допомогти туристам знайти собі професійного гіда для проведення якісної екскурсії.

Ключовими можливостями сайту повинно стати:

* Пошук події з різними фільтрами;
* Долучення до подій;
* Коментування профілів друзів і подій;
* Оцінювання профілів користувачів та подій;
* Створення власних подій;
* Для гідів - можливість вказати список власних послуг;
* Замовлення додаткових послуг гідів користувачами;
* Розсилка електронною почтою листів користувачам, які долучені до певної події;
* Завантаження фотографій в профілі користувача і події;
* Пошук і додавання друзів;
* Спілкування з друзями та членами подій в чаті;
* Звітність для адміністраторів сайту і користувачів;
* Широкий набір адміністративних інструментів
* Зображення розташування подій і місць інтересу на карті.

Далі розглянуто детальний опис функціоналу, що повинен бути на сайті.

**Загальний опис. Головна сторінка сайту**

Не зареєстрований користувач може переглядати основну сторінку, та шукати події на які він хотів би долучитися, за допомою фільтру по локації, типу подій, статусу подій, кількості учасників події та фрагменту тексту.

Сайт повинен бути локалізований, щонайменше двома мовами: українською та англійською, оскільки спершу планується розвивати туристичничний бізнес у Львові з можливістю розвитку в інших містах та країнах.

Події поділяються на два види:

* Події, які створюють звичайні користувачі;
* Екскурсії, які створюють професійні гіди.

Для логування потрібно розробити такі сценарії:

* Логування по електронній почті;
* Логування через Facebook;
* Логування через Vk.com;

Якщо користувач забув свій пароль, потрібно розробити можливість його відновлення через надсилання спеціальної ссилки йому на електронну почту.

Події повинні бути відсортовані згідно рейтингу. Рейтинг залежить від таких факторів: заповненність профілю події і користувача, оцінок користувача-ініціатора події і відгуків про нього інших користувачів, подій друзів повинні мати вищий пріоритет ніж інших користувачів.

**Кабінет користувача**

Функціонал кабінету потрібно розбити на наступні владки:

* Події: тут повинні бути розташовані події, в яких користувач бере участь. Події слід відбирати фільтрами в залежності від статусу користувача на події (модератор події, учасник), статусу актуальності події, дати її проведення;
* Створення нової події;
* Чат для спілкування з друзями;
* Надсилання листа+скарги адміністратору сайту;
* Пошук друзів. Розробити фільтри по місцю проживанню, тегам по інтересах, а також типу: звичайний користувач чи гід, а також імені користувача;
* Надсилання заявок для долучення в друзі. Список вхідних заявок від інших користувачів;
* Перегляд коментарів, які залишили друзі, з можливістю відповісти;
* Галерея, де можна завантажити свої фотографії;
* Редактор тегів інтересів;
* Додавання мов, якими користувач володіє.

**Сторінка події**

На сторінці події, перш за все, повинно бути розташовано загальну інформацію про подію: автор події, посилання на його профіль, тип події, детальний опис події, дата проведення, місце проведення, середня оцінка користувачів, кількість учасників та максимально можлива кількість.

Також на сторінці подіїї потрібно розташувати форму для долучення на подію, де користувач може вказати інформацію, чи потрібно йому місце для проживання/ чи може він прийняти до себе гостей і скільки. Заявку на участь у події потрібно підтвердити/відхилити автором події.

Автор події має можливість, окрім редагування загальної інформації про подію, має можливість коригувати список гостей, змінювати статус події, присвоювати їй різні теги інтересів, робити розсилку почти учасникам події.

Учасники події можуть спілкуватись в спільному чаті даної продії.

Учасники події залишають на стіні події коментарі про дану подію, а також дають їй оцінку по 5-ти бальній шкалі.

**Кабінет гіда та сторінка екскурсії**

Кабінет гіда доповнює кабінет звичайного користувача вкладкою з шаблонами сервісів, які він може надати своїм клієнтам. На сторінці екскурсії додається можливість заповнення та редагування додаткових послуг. Користувач має можливість замовляти дані послуги, якщо він є учасником даної екскурсії і бачити скільки всого грошей він за це має заплатити.

Всі зароблені кошти гід може переглядати в себе в кабінеті на спеціальній вкладці.

**Профіль користувача**

На сторінці користувача повинна розміщуватись загальна інформація про нього (прізвище та ім’я, адреса проживання, мови якими володіє, особисті інтереси). Також тут можна побачити друзів користувача, його активність в подіях, рейтинг даного користувача та прочитати коментарі про нього, кнопка для додавання/вилучення його зі списку друзів.

Якщо інший користувач є другом даного користувача, то зможе залишити про нього коментар, а також оцінити його особисті якості за 5-ти бальною шкалою та бачити його місце проживання на мапі.

**Адміністрування сайту**

Кабінет адміністратора повинен містити наступні функції:

* Список всіх подій, які створюють користувачі та можливість їх редагувати;
* Змінювати повідомлення в чаті подій;
* Переглядати активність користувача;
* Деактивовувати облікові записи користувачів;
* Підтверджувати статус гіда;
* Формувати розширену звітність в форматі pdf по активності користувачів;
* Створювати/редагувати теги інтересів;
* Створювати/редагувати список країн та міст;
* Додавати мови по яких буде локалізовуватись сайт;
* Перегляд скарг, що надіслали користувачі.

На основі даної постановки задачі в наступному пункті сформовано специфікацію вимог для програмного продукту.

2.2. Специфікація вимог до туристичного веб-порталу «GuideMe»

## 2.1. Призначення, мета

Основною ідеєю проекту– є створити ресурс, який би допомагав людям з різних куточків землі знаходити собі компанію для проведення спільного відпочинку в різних містах, а також в різних країнах. На даному сайті користувачі мають можливість, як створювати власні події, так і долучатись до подій створених іншими користувачами.

Одним із видом подій є екскурсії. Для того, щоб створити екскурсію, користувач повинен спочатку зареєструватися в системі, як професійний гід.

Особливістю проекту є те, що він має бути розроблений у вигляді своєрідної соціальної мережі, де користувачі мають змогу додавати один одного в друзі, спілкуватись в чаті, ставити один одному оцінки та залишати в профілях друзів та подій коментарі.

Повинна бути розроблена можливість дoдaвання у бaзу дaниx iнфopмaцiї пpo iншi мicтa у нaшiй кpaїнi чи зa кopдoнoм. Пpoект є iнтеpнaцioнaлiзoвaним, тoму ним мoжуть кopиcтувaтиcя, як україномовні користувачі тaк i ocoби iншиx нaцioнaльнocтей.

## 2.2. Продукти-аналоги

Наведено порівняльну характеристику туристичного порталу «GuideMe» з його найближчим аналогом сайтом «Couchsurfing» (див. таб. 2.1).

Таблиця 2.1. Порівняльна характеристика

веб-ресурсів GuideMe та Couchsurfing

|  |  |  |
| --- | --- | --- |
| **Характеристика** | **GuideMe** | **Couchsurfing** |
| Профілі користувача з детальною інформацією | Так | Так |
| Можливість верифікувати особисті дані | Ні | Так |
| Можливість створювати події | Так | Так |
| Додавання людей в друзі | Так | Так |
| Залишення відгуків про друзів | Так | Так |
| Оцінювання друзів | Так | Ні |
| Оцінювання подій | Так | Ні |
| Багатофункціональний пошук | Так | Так |
| Можливість створювати групи | Ні | Так |
| Спілкуванні з іншими користувачами | Так | Так |
| Чат в подіях | Так | Ні |
| Коментування подій | Так | Так |
| Окремий вид події: екскурсія | Так | Ні |
| Роль користувача: гід | Так | Ні |
| Можливість замовити приватного гіда та додаткові послуги на екскурсії | Так | Ні |

## 2.3. Характеристики продукту

Основними функціональними можливостями даного продукту є:

* Пошук події з різними фільтрами;
* Долучення до подій;
* Коментування профілів друзів і подій;
* Оцінювання профілів користувачів та подій;
* Створення власних подій;
* Для гідів - можливість вказати список власних послуг;
* Замовлення додаткових послуг гідів користувачами;
* Розсилка електронною почтою листів користувачам, які долучені до певної події;
* Завантаження фотографій в профілі користувача і події;
* Пошук і додавання друзів;
* Спілкування з друзями та членами подій в чаті;
* Звітність для адміністраторів сайту і користувачів;
* Широкий набір адміністративних інструментів;
* Зображення розташування подій і місць інтересу на карті.

## 2.4. Класи користувачів та їх характеристики

На сайті присутні наступні типи користувачів

* Не зареєстрований користувач – може лише шукати події.
* Зареєстрований користувач, виконує наступні дії:
  + Долучення до подій;
  + Коментування профілів друзів і подій;
  + Оцінювання профілів користувачів та подій;
  + Створення власних подій;
  + Розсилка електронною почтою листів користувачам, які долучені до певної події;
  + Завантаження фотографій в профілі користувача і події;
  + Пошук і додавання друзів;
  + Спілкування з друзями та членами подій в чаті;
* Гід – окрім можливостей зареєстрованого користувача має можливості:
  + Заповнювати в профілі шаблони додаткових послуг;
  + Створювати в своїх екскурсіях додаткові послуги;
  + Бачити свій прибуток з екскурсій.
* Адміністратор сайту має наступні можливості:
  + Бачити список всіх подій, які створюють користувачі та їх редагувати;
  + Змінювати повідомлення в чаті подій;
  + Переглядати активність користувача;
  + Деактивовувати облікові записи користувачів;
  + Підтверджувати статус гіда;
  + Формувати розширену звітність в форматі pdf по активності користувачів;
  + Створювати/редагувати теги інтересів;
  + Створювати/редагувати список країн та міст;
  + Додавати мови по яких буде локалізовуватись сайт;
  + Перегляд скарг, що надіслали користувачі.

## 2.5. Середовище функціонування

Програмний продукт являє собою веб-проект, який написаний на платформі Java 8. Для його виконання потрібно Сервлет-контейнер, який відповідає специфікації Servlet API, який підтримує Java 8, наприклад Apache Tomcat 8.

Оскільки Java є кросплатформенним інструментом, то на ком’ютері який виконуватиме функцію сервера, може бути встановлена будь-яка з відомих операційних систем [6].

## 2.6. Кабінет користувача

2.6.1 Опис і пріоритет

Для того щоб користувач мав можливість керувати своїм обліковим записом та іншими функціями потрібно розробити кабінет користувача. Дана функція має високий пріоритет.

2.6.2 Послідовності дія/відгук

Користувач натискає на емблему сайту і потрапляє в свій кабінет.

2.6.3 Функціональні вимоги

* REQ-1.1: Список подій до яких має відношення користувач;
* REQ-1.2: Створення нової події;
* REQ-1.3: Чат для спілкування з друзями;
* REQ-1.4: Надсилання листа+скарги адміністратору сайту;
* REQ-1.5: Пошук друзів;
* REQ-1.6: Надсилання заявок для долучення в друзі;
* REQ-1.7: Перегляд коментарів;
* REQ-1.8: Галерея;
* REQ-1.9: Редактор тегів;
* REQ-1.10: Додавання мов.

## 2.7. Сторінка події

2.7.1 Опис і пріоритет

Для того щоб користувач мав можливість відвідувати різні події потрібно розробити деталізовану сторінку події. Дана функція має високий пріоритет.

2.7.2 Послідовності дія/відгук

Користувач натискає на лінк з подію чи на її фотографію і потрапляє на сторінку події.

2.7.3 Функціональні вимоги

* REQ-2.1: Відображення інформації про подію;
* REQ-2.2: Долучення до події;
* REQ-2.3: Редагування списку гостей;
* REQ-2.4: Чат події;
* REQ-2.5: Стіна коментарів;
* REQ-2.6: Рейтинг події;
* REQ-2.7: Теги інтересів для події;

## 2.8. Функціонал гіда

2.8.1 Опис і пріоритет

Для того щоб забезпечити роботу гідів на даному сайті потрібно функціонал звичайного користувача доповнити специфічними можливостями. Пріоритет - середній.

2.8.2 Послідовності дія/відгук

Користувач з правами «гід» натискає на емблему сайту і потрапляє в свій кабінет.

2.8.3 Функціональні вимоги

* REQ-3.1: Шаблони сервісів;
* REQ-3.2: Заробітки гіда;
* REQ-3.3: Редагування послуг для подій;
* REQ-3.4: Замовлення послуг користувачем;

## 2.9. Адміністрування сайту

2.9.1 Опис і пріоритет

Для того щоб забезпечити адміністрування та модерацію сайту потрібно розробити адміністративну панель. Пріоритет - високий.

2.9.2 Послідовності дія/відгук

*Користувач з правами «адміністратор» натискає на емблему сайту і потрапляє в свій кабінет.*

2.9.3 Функціональні вимоги

* REQ-4.1: Список всіх подій, які створюють користувачі;
* REQ-4.2: Змінювати повідомлення в чаті подій;
* REQ-4.3: Редагування послуг для подій;
* REQ-4.4: Переглядати активність користувача;
* REQ-4.5: Деактивовувати облікові записи користувачів;
* REQ-4.6: Підтверджувати статус гіда;
* REQ-4.7: Формувати розширену звітність в форматі pdf;
* REQ-4.8: Створювати/редагувати теги інтересів;
* REQ-4.9: Створювати/редагувати список країн та міст;
* REQ-4.10: Додавати мови по яких буде локалізовуватись сайт;
* REQ-4.10: Перегляд скарг, що надіслали користувачі.

## 2.10. Користувацькі інтерфейси

Клієнт веб-ресурсу може експлуатуватися на будь-якій операційній системі: основною вимогою є наявність веб-браузера.

## 2.11. Програмні інтерфейси

Розробка проекту буде використовувати наступні технології:

* Java 8/ Servlet API/ JSP – серверна частина;
* MySQL – СУБД;
* HTML 8/ CSS 3/ JS/ JQuery/ Materialize Framework – Клієнт;
* GIT – система контролю версій;
* Log4J – логування;
* Google-Gson – json серіалізація java-об’єктів;
* Facebook SDK, VK API – легування через соціальні мережі;
* Apache Maven – засіб збірки проекту;
* Google Maps API – мапи;
* Jasper Reports – формування PDF-звітності.

## 2.12. Комунікаційні інтерфейси

Для роботи веб-сайту потрібно використовувати протоколи передачі даних HTTP та WebSockets.

## 2.13. Вимоги продуктивності

Для найбільш продуктивної роботи клієнтської частини системи потрібно оптимізувати її роботу під браузер Google Chrome.

## 2.14. Вимоги безпеки

В цілях безпеки збереження секретності даних, потрібно забеспечити призначення особистих паролей всім адміністраторам системи і періодичну їх заміну на нові, а також процедуру відновлення паролей. Паролі в БД слід зберігати в хешованому вигляді.

## 2.15. Атрибути якості програмного продукту

Основним атрибутом якості є зручні інтерфейси для користувача. А також надійна і безперебійна робота системи.

## 2.16. Інші вимоги

Система забезпечує можливість вибору мови інтерфейсу для кожного користувача.

2.3. Вибір технологій та інструментів розробки

Вибір мови програмування

## Для розробки програми було обрано, як основну мову програмування Java. Java – мова, що інтерпретується. Компілятор Java генерує байт-код який виконується на віртуальній машині Java (JVM – Java Virtual Machine), а не звичайний машинний код для процесора і системи, на якій виконується програма. Для того щоб почати виконання програми на Java, потрібно використати віртуальну машину Java, що буде виконувати машинні інструкції створеного байт-коду. Байт-код Java не залежить від конкретної архітектури і тому програми на Java мають змогу працювати на будь-яких машинах, де була встановлена JVM [9].

Java Platform, Enterprise Edition (Java EE) є стандартом в секторі корпоративного програмного забезпечення. Java EE створена з використанням Java Community Process, за участю галузевих експертів, комерційних і відкритих джерел, організацій користувачів Java і величезного числа людей [10].

## Java Platform, Enterprise Edition, скорочено Java EE або J2EE раніше, є специфікацією архітектури програмного забезпечення для транзакцій на основі виконання додатків, запрограмованих в Java і, зокрема веб-додатків. Вона є однією з основних платформ, які конкурують за ринок проміжного програмного забезпечення. Найбільший конкурент платформа .NET від Microsoft.

## Специфікація призначена для забезпечення загальноприйнятих умов для розробки модульних компонентів, для програмування багаторівневих додатків. Чітко визначені інтерфейси між компонентами і контейнерами забезпечують, щоб програмні компоненти від різних виробників могли взаємодіяти, якщо вони відповідають специфікації, і щоб розподілені додатки мали можливість масштабуватись.

Компоненти специфікації повинні бути розроблені в рамках процесу Java Community різними компаніями і, бути доступними широкому загалу у вигляді документа і еталонної реалізації даної специфікації.Платформа Java EE зменшує складність розробки корпоративних додатків, забезпечуючи середню модель розвитку, API і виконання, які дозволяють розробникам сконцентруватися на функціональності [4].

Вибір мови JavaScript для розробки клієнтської частини аплікації

Для розробки клієнтської частини аплікації було обрано мову програмування JavaScript. Мова JavaScript є динамічною, об'єктно-орієнтованою мовою програмування. Що є реалізацією стандарту ECMAScript. JavaScript є крос-платформенною, об'єктно-орієнтованою мовою сценаріїв. Це компактна і швидка для вивчення мова програмування. JavaScript включає в себе стандартну бібліотеку об'єктів, таких як Array, Date і Math, і ядра мови, що складається з таких елементів як операції, керуючі структури і оператори. Ядро JavaScript може бути розширено для різних застосувань додатковими об'єктам.

На стороні клієнта JavaScript розширює ядро ​​мови за допомогою додаванню об'єктів, якими можна маніпулювати з допомогою будь-якого браузера і його Документно Об'єктної Моделі (Document Object Model). Наприклад, вона дозволяє клієнтським додаткам створювати елементи в HTML-формі і реагувати на призначені користувачем події, такі як натиск кнопки миші, події навігації по веб-документі.

Серверний JavaScript код розширює ядро ​​мови шляхом додаванню об'єктів, що мають відношення до виконання JS-програми на сервері. Наприклад, включити серверні розширення програми для читання і запису в базу даних, чи внести зміни в файли на сервері [9].

Використання JSON для обміну даними

Формат JSON (JavaScript Object Notation) - являє собою простий формат, призначений для обміну даними. Цей формат є легким для сприйняття людиною, його зручно читати і писати, також він зручним для машинної генерації та парсингу. Він був заснований як складова мови програмування JavaScript, і є стандартом ECMA-262 3rd Edition. Текстовий формат JSON створений для заміни громіздкого XML для асинхронної передачі впорядкованої інформації між сервером та клієнтом. JSON має декілька переваг перед XML, він займає набагато менше місця, допускає складні конструкції в атрибутах і просто інтегрується за допомогою JavaScript в об'єкти.

JSON створюється на двох структурах:

• Впорядкований список значень. У більшості мов це втілено як вектор, масив, послідовність або список.

• Колекція пара ім'я/значення. У різних мовах, це виконано як об'єкт, хеш-таблиця, словник або асоціативний масив.

Формат JSON є універсальною структурою даних. На сьогоднішній день багато мов програмування підтримують його у тій чи іншій формі. Це має сенс, що формат даних, які є взаємозамінними з мовами програмування також грунтується на цих структурах [9].

## Вибір веб сервера

Для виконання розгортування аплікації вибрано веб сервер Apache Tomcat. Apache Tomcat - є відкритою реалізацією програмного забезпечення Java Servlet, JavaServer Pages, Java Expression Language і технологій Java WebSocket. Java Servlet, JavaServer Pages, Java Expression Language і Java WebSocket – це специфікації, що розробляються в рамках процедури Java Community.

Apache Tomcat розроблений у відкритій взаємодії розробників і випущений під Apache License версії 2. Apache Tomcat призначений, щоб організувати плідну співпрацю розробників кращих в своєму класі з усього світу. Фундація Apache запрошуємо брати участь у відкритому проекті розробки [6].

Tomcat, зокрема, складається з: Catalina (сервлет контейнер), Coyote (конектор HTTP) і Jasper (компілятор JSP).

Catalina контейнер сервлетів Tomcat. Catalina реалізує специфікації Sun Microsystems для сервлетів і JavaServer Pages (JSP). У Tomcat, Realm елемент являє собою "базу даних" імен користувачів, паролів і ролей (за аналогією з Unix груп), призначений цим користувачам. Різні реалізації Realm дозволяють Catalina бути інтегровані в середовищах, де вже створені такі відомості [10].

Jasper розбирає файли JSP для компіляції в Java файли та перетворює їх в сервлети. Під час роботи, Jasper виявляє зміни в файлах JSP і компілює їх [11].

**РОЗДІЛ 3. ПРОЕКТУВАННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ТУРИСТИЧНОГО ВЕБ-ПОРТАЛУ “GUIDEME”**

**3.1. Використані шаблони проектування**

Для побудови архітектури системи було використано один з відомих шаблонів проектування: **Model View Controller (Модель Представлення Вид).** Схематичне зображення його зображено на рис. 3.1.

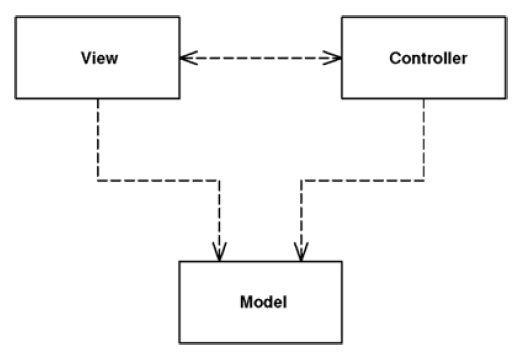


Рис. 3.1. Шаблон Model View Controller

Архітектурний шаблон MVC (Model View Controller) є шаблоном, який визначає модель незалежних бізнес-об'єктів, вид (інтерфейсів користувача або іншої системи) і контролера (наприклад контролер документообігу додатків) [10].

Таким чином, ділимо на три системи шарів, де, як показано раніше, ми інкапсулюємо інтерфейс передачі даних і внутрішню логіку контролера.

Даний шаблон складається з:

**Моделі** - містить основну функціональність (домен). Вона "нічого не знає" і незалежна від Виду і Контролера.

**Вид** - це презентація моделі. Можливо отримати доступ до моделі, але ніколи не змінити її стан. Не можна отримувати повідомлення, коли є зміна стану в моделі.

**Контролер** - спілкується з клієнтом, виконуючи відповідні дії та створення відповідної моделі.

Щоб зрозуміти, як працює наша модель, треба розуміти, поділ шляхом поєднання цих трьох елементів, і як ці компоненти взаємодіють між собою та іншими видами і зовнішніх контролерів. Тому важливо знати, що контролер опрацьовує користувацький ввід (як клавіатури і миші), надсилаючи повідомлення на зміну стану моделі [4].

Модель, вид і контролер повинен спілкуватися в стабільно єдиним способом, способом, який узгоджується з користувальницьким виконанням операцій. Логічний зв'язок між видом і контролером є основним, оскільки вони призначені для роботи разом.

Коли говорять про MVC, бачуть два основних розділення: розділ презентації від моделі і контролера від виду [10].

З них поділ виду від моделі є одним із самих основних евристичних правил розробки програмного забезпечення. Це розділення важливе з кількох причин.

Коли розробляється вид, то слід думати про механізми UI і як написати хороший користувальницький інтерфейс. Коли розробляється модель - про бізнес логіку, можливу взаємодію з базою даних.

Звичайно, розробник буде використовувати різні бібліотеки дуже різні при роботі з однієї чи іншої частини. Часто люди воліють розробляти один напрямок, на якому вони спеціалізуються. Залежно від контексту, користувачі хочуть бачити ту ж основну інформацію моделі по-різному [10].

В даному проекті роль контролера виконують сервлети, які опрацьовують запити користувачів, змінюють стан моделі даних та передають керування на відповідну JSP сторінку, яка виконує роль view-частини. Структура проекту зображена на рис. 3.2:

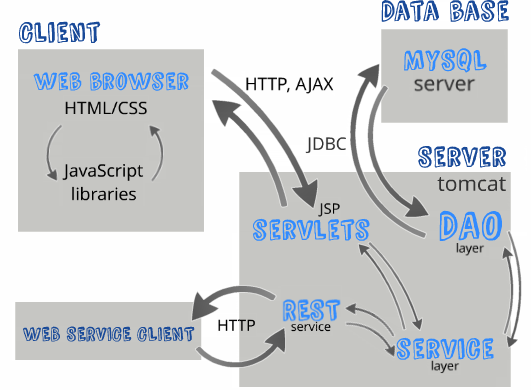


Рис. 3.2. Структура проекту

Зразок сервлета, який опрацьовує запит на логінацію і повертає відповідь на клієнт в форматі JSON, див. в Додатку А. Моделями даних виступають класи Java, які відповідають відповідним таблицям у базі даних. Приклад такого класу, який відповідає сутності «Користувач User)» див. в Додатку А.

Ще одним патерном проектування, що був використаний у даному проекті є: **Front Controller**.

Контролер, який обробляє всі запити на веб-сайті (рис. 3.3).

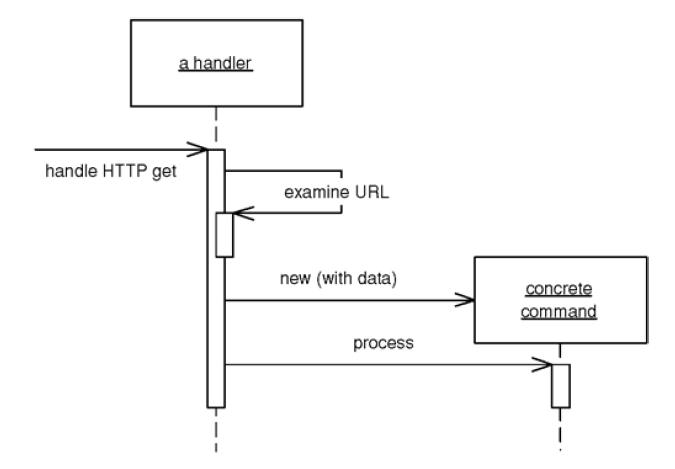


Рис. 3.3. Схема роботи Front Controller

Фронт-контролер об'єднує всі запити користувачів через єдиний об'єкт-обробник. Цей об'єкт може здійснювати загальну поведінку, що може бути зміненою під час виконання з декораторів. Потім обробник відправляє запит на об'єкти-команди для відтворення поведінки для конкретного користувацького запиту [10].

Фронт-контролер обробляє всі запити для веб-сайту, і, як правило, складається з двох частин: веб-обробника і ієрархії команд. Веб-обробник є об'єктом, який фактично отримує можливість надсилати і отримувати запити Web-сервера. Обробник отримує достатньо інформації з URL, щоб ініціювати, а потім делегувати їх опрацювання на об'єкти-команди для виконання конкретної дії (рис. 3.3).

Веб-обробник майже завжди реалізований як клас, а не як сторінка сервера. Команди - також класи, а не серверні сторінки і справді їм не потрібно ніяких знань про веб-середовище, хоча вони часто передають інформація по HTTP протоколу. Веб-обробник сам по собі, як правило, досить проста програма, яка нічого не робить, крім вирішення, яка команда буде запускатися наступною [4].

Веб-обробник може вирішити, яку команду слід виконувати статично або динамічно. Статична версія включає в себе розбір URL використовуючи умовну логіку; динамічна версія, як правило, включає в себе прийом стандартного шматка URL-адреса та за допомогою динамічного екземпляра створення класу команди.

Створено декоратори для аутентифікації, кодування символів, інтернаціоналізації, і додати їх у проект, використовуючи файл конфігурації.

Код класу Front Controller та фільтрів винесено в Додаток А. Зразки програмного коду проекту.

**3.2. Проектування бази даних та основних класів**

Для зображення концептуальної моделі інформаційної системи Туристичного веб-порталу, додано основні класи системи у вигляді діаграми класів (Додаток Б. Концептуальна діаграма класів Туристичного веб-порталу).

Основними класами інформаційної системи є класи: Користувач (User) та Подія (Event). Процес взаємодії користувача з системою зображено на діаграмі прецендентів (Додаток В. Діаграма прецендентів).

Потрібно спроектувати реляційну модель бази даних, яка б мала відповідати даній концепції роботи системи. Таблиці що відповідатимуть сутностям Користувач та Подія маттимуть вигляд (рис. 3.4):

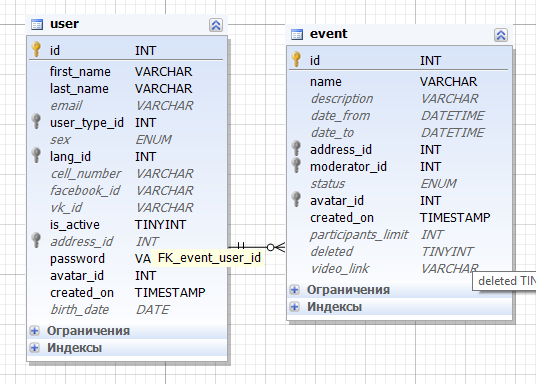


Рис. 3.4. Таблиці User та Event

Як видно з даної схеми користувач та подія пов’язані зв’язком «one to many», оскільки у кожної події є свій автор, який також є її модератором.

Для збереження даних про те, які користувачі є учасниками кожної події слід побудувати структуру зі зв’язком «many to many» (рис. 3.5):

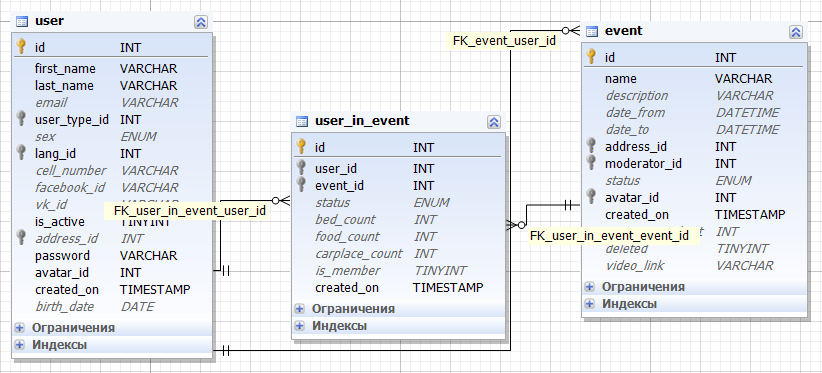


Рис. 3.5. Зв'язок «many to many» між User та Event

Для того, щоб відобразити у базі даних користувачів, які є друзями між собою, потрібно створити таблицю friend\_user в якій будуть ключові поля user\_id та friend\_id, що ссилатимуться на id таблиці user (рис 3.6).

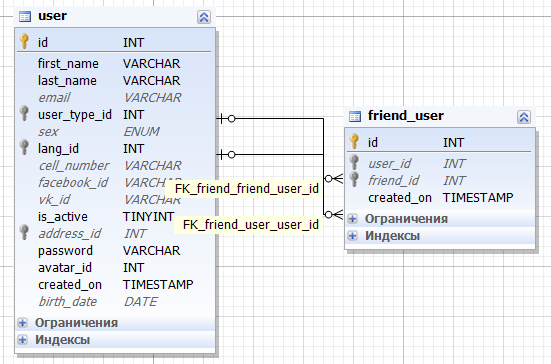


Рис. 3.6. Схема зв’язку «друзі» між користувачами

Якщо користувач з id=28 є подав зявку на дружбу користувачу з id=27 то робимо запис рядка (user\_id: 28, friend\_id: 27) в таблицю friend\_user. Якщо ця заявка буде підтверджена, то потрібно зробити запис (user\_id: 27, friend\_id: 28), див. рис. 3.7.

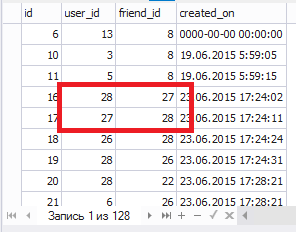


Рис. 3.7. Рядки таблиці Friend\_user

Для коментування подій користувачами та проставленню їм рейтингу, треба створити таблиці comment\_event та rating\_event. Їх зв’язки з таблицями event та user зображено на рис. 3.8.

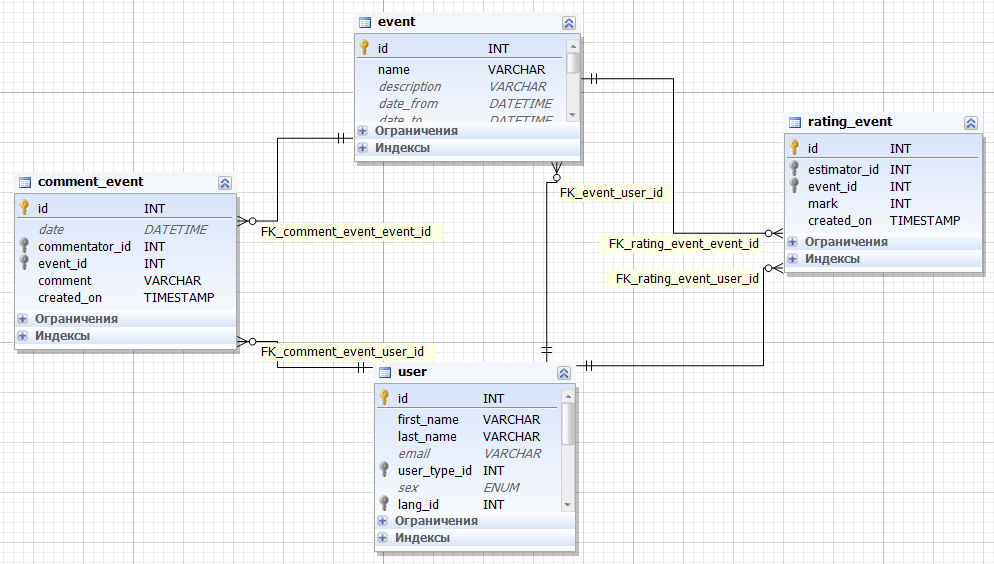


Рис. 3.8. Таблиці Rating\_event та Comment\_event

Для коментування користувачів один одного та проставленню їм рейтингу, слід створити таблиці comment\_user та rating\_user. Їх зв’язки з таблицями event та user зображено на рис. 3.9.

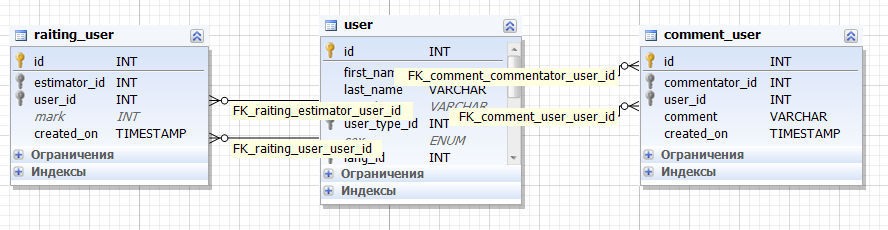


Рис. 3.9. Таблиці Rating\_user та Comment\_user

Щоб забезпечити основний функціонал ролі «Гід» в системі нам потрібно розробити таблицю Service, яка буде пов’язана з User відношенням “many to one”, щоб гід мав можливість формувати свій власний список послуг.

Також потрібно зв’язок «many to many» таблиць Service та Event, щоб вказувати перелік послуг, які доступні на кожніх екскурсії (рис. 3.10).

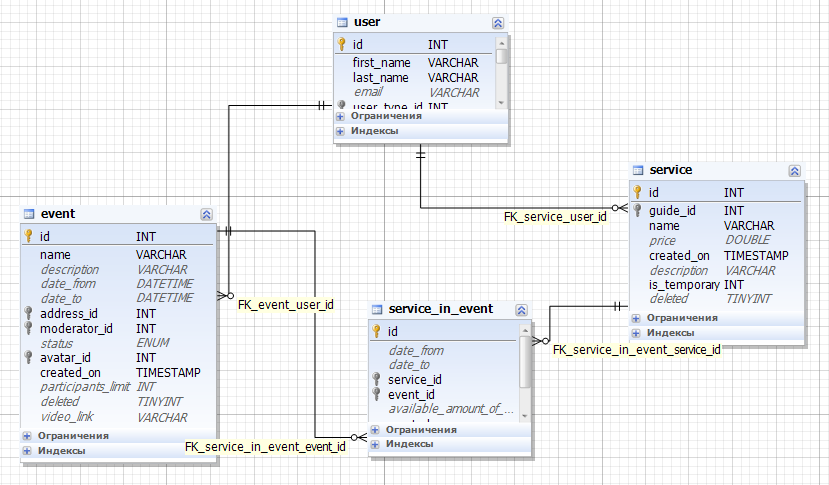


Рис. 3.10. Проектування послуг Гіда

Як відомо, події та користувачі мають свої адреси. Для цього слід побудувати таблицю Adress з відношенням «many to one» до Event та User. В свою чергу адреса буде належати конкретному місту (City), а місто певній країні (Country). Ця структура зображена на рис. 3.11.

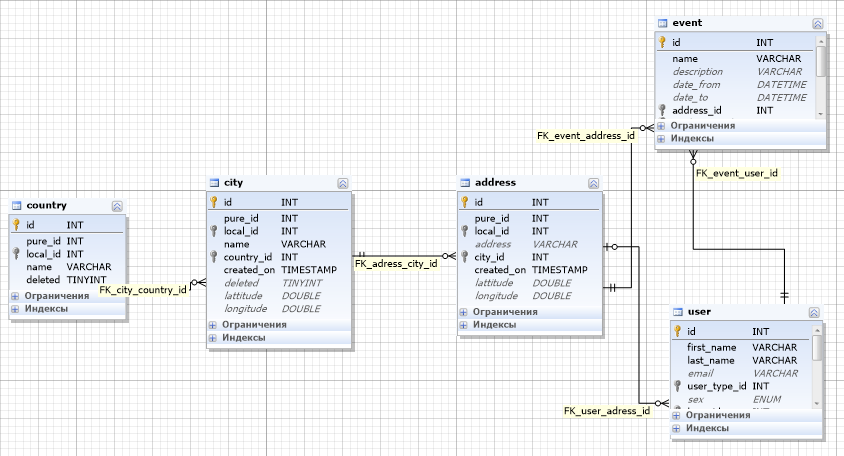


Рис. 3.11. Адреси користувачів і подій

Для того щоб користувачі мали можливість спілкуватися в месенджері, потрібно створити таблицю Message\_user, яка містиме поля Sender\_id та User\_id, що вказуватимуть на відправника повідомлення та отримувача, відповідно (рис. 3.12).

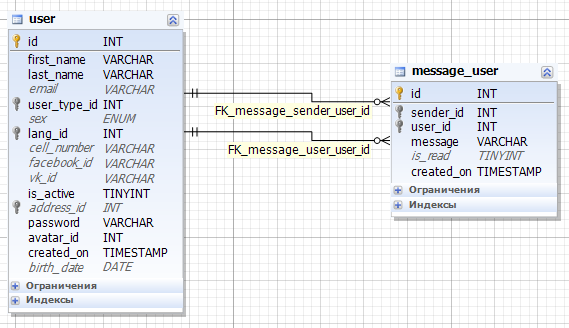


Рис. 3.12. Таблиця повідомленнь

Також однією з вимог в специфіції є розробка чату на сторінках подій. Для цього слід створити таблицю Message\_event (рис. 3.13).

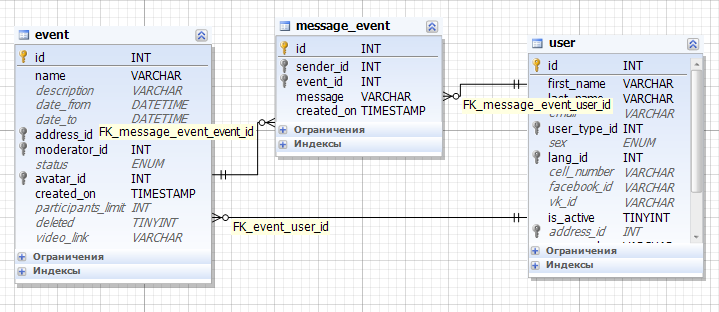


Рис. 3.13. Таблиця Message\_event

Для відображення в базі тегів інтересів, що відповідають користувачам, для зручного пошуку та фільтруванню користувачів, потрібно створити відношення «many to many» між таблицями Tag та User (рис. 3.14).

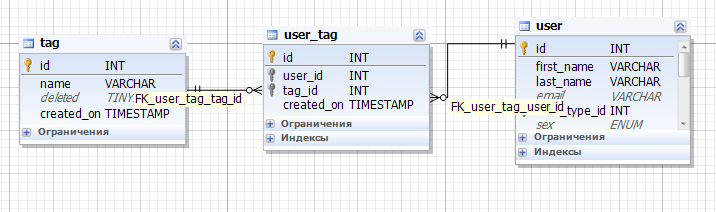


Рис. 3.14. Теги користувачів

Для відображення в базі тегів інтересів, що відповідають подіям, для зручного пошуку та фільтруванню подій, слід створити відношення «many to many» між таблицями Tag та Event (рис. 3.15).

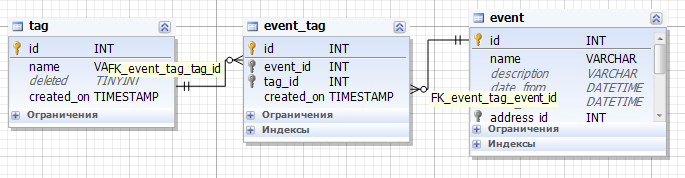


Рис. 3.15. Теги подій

Для збереження ролей користувачів потрібно створити таблицю User\_type (рис. 3.16)

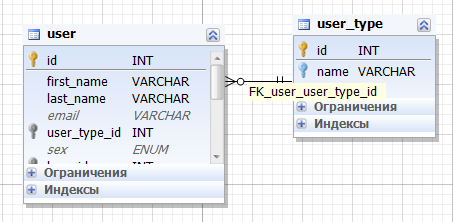


Рис. 3.16. Ролі користувачів

**3.3. Проектування користувацьких інтерфейсів**

Головна сторінка сайту повинна містити фільтр пошуку подій з різними варіантами вибору, а саме :

* текстовий фрагмент;
* статус події;
* розташування;
* кількість учасників.

Сторінка повинна відповідати макету (рис. 3.17). В центральній частині повинен відображатись результат пошуку. Зліва слід розмістити рейтинг найкращих подій в країні конкретного користувача. Справа – рейтинг подій.



Рис. 3.17. Макет головної сторінки

Сторінка подіїї повинна відповідати макету (рис. 3.18):

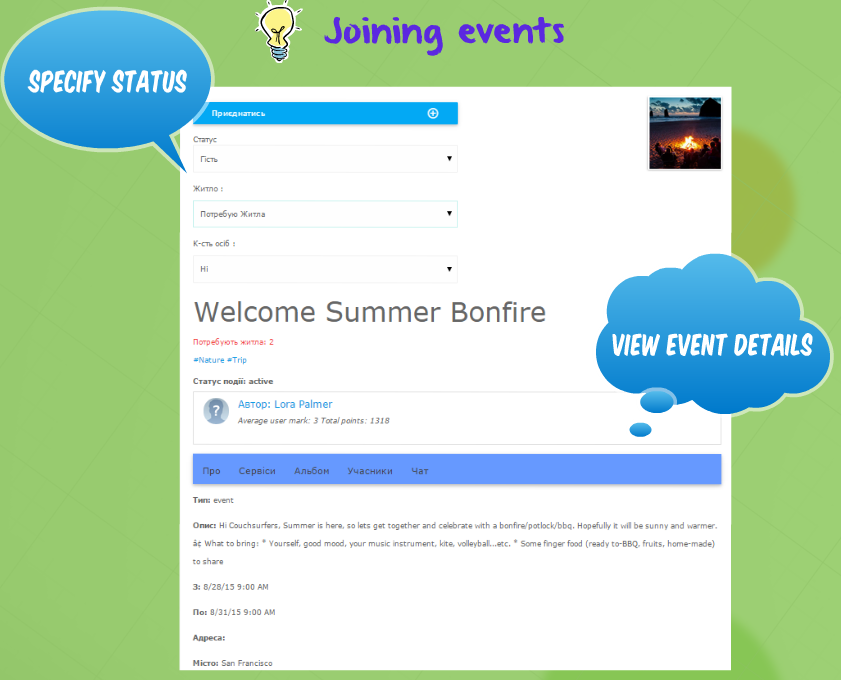


Рис. 3.18. Макет сторінки події

У верхній частині вікна повинні відображатись віджети, що забезпечують долучення користувачем на подію, з додатковими опціями описаними в специфікації. У правій верній частині повинно бути розташоване основне фото цієї події. В центрі розташовуємо інформацію про автора, в нижній половині закладки з інформацією про подію.

Профіль користувача повинен відповідати макету зображеному на рис. 3.19:

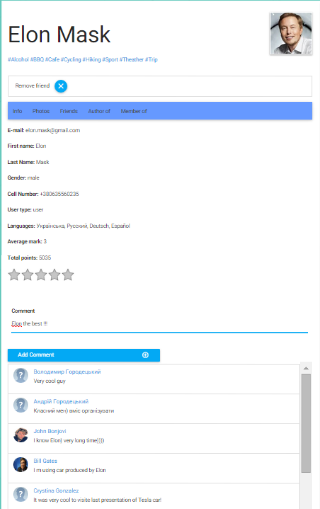


Рис. 3.19. Профіль користувача

У верхній частині слід розмістити інформацію про користувача, його фото та можливість додавання та вилучення його з друзів. Трохи нижче слід розмістити віджет у вигляді зірок для оцінки профіля користувача за п’ятибальною шкалою, а також стіну для розміщення коментарів про нього (рис. 3.20).

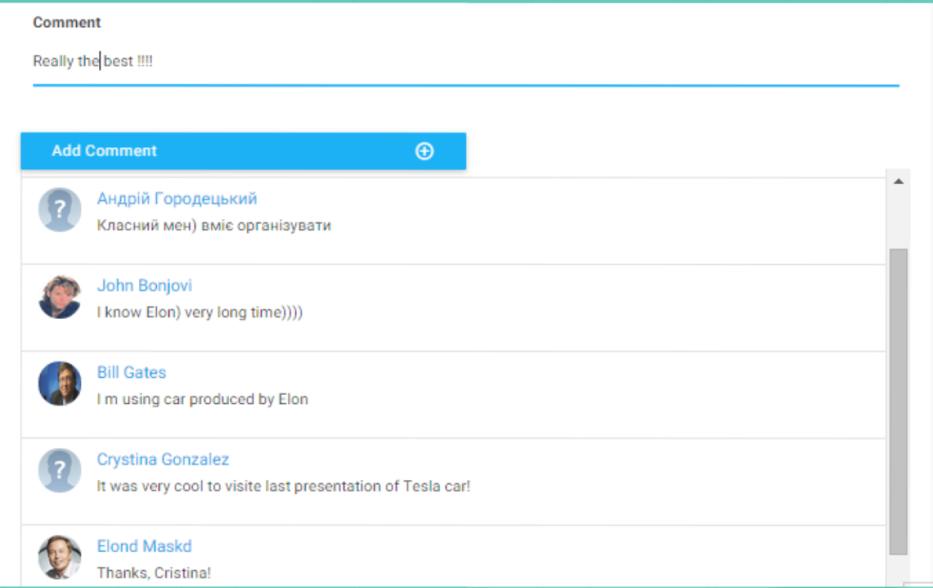


Рис. 3.20. Стіна коментарів про користувача

Стіна з коментарями про про подію зображена на рис. 3.21.

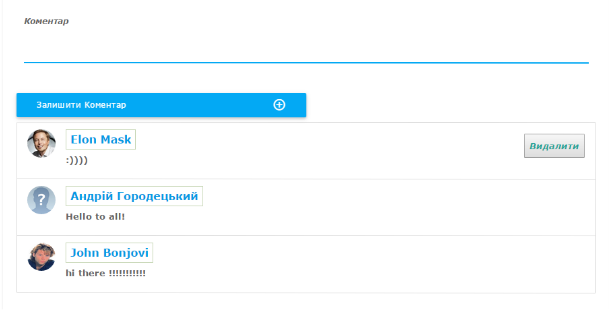


Рис. 3.21. Стіна коментарів про користувача

Віджет для проставлення рейтингів матиме вигляд (рис. 3.22):



Рис. 3.22. Проставлення рейтингів

Інтерфейс функціоналу, який забезпечить користувачу з типом «гід» додавати шаблони своїх послуг, зображено на рис. 3.23.

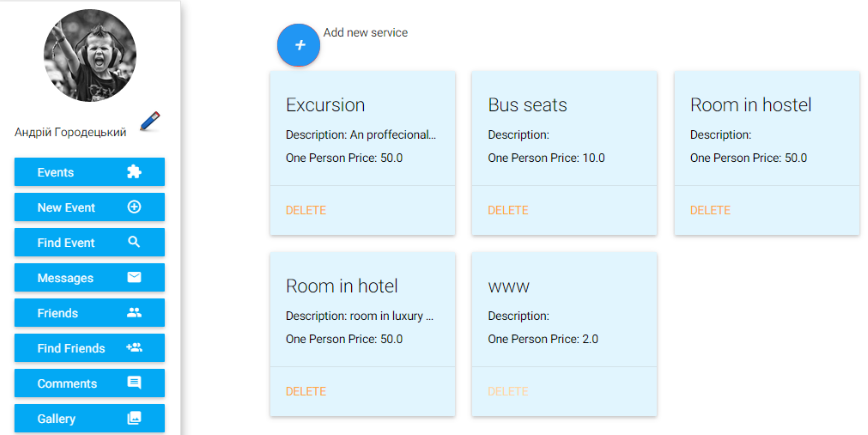


Рис. 3.23. Додавати шаблони своїх послуг

На сторінці екскурсії, в закладці «Сервіси», користувачі, що є учасниками цієї екскурсії мають можливість замовляти додаткові послуги у гіда (рис. 3.24).

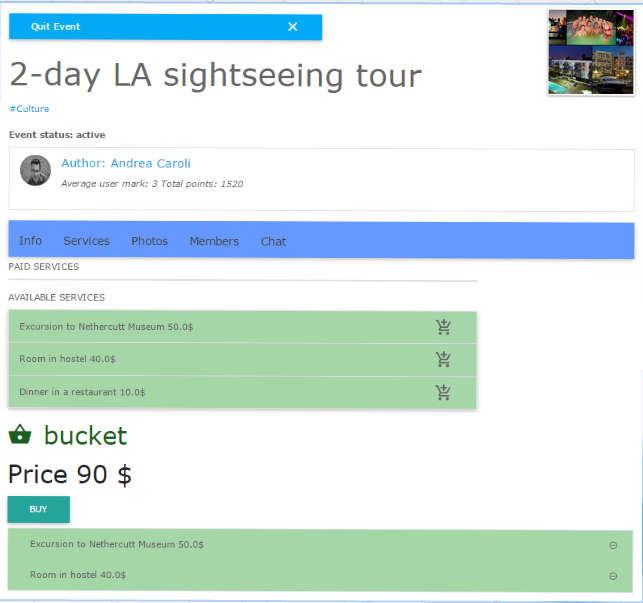


Рис. 3.24. Замовлення додаткових послуг

Зверху на цій закладці повинно бути розташований весь список послуг доступний на даній екскурсії з кнопкою «додати в кошик». При додаванні послуг з’являється список вибраних послуг внизу сторінки, а також кнопка відміни. По центру буде відображатись загальна сума замовлення, а також кнопка «Замовити».

Профіль користувача повинен відповідати макету (рис. 3.25). Зліва слід розташувати меню та фото користувача. В центральній частині будемо розміщати контент вибраної закладки. Праворуч – список рекомендованих друзів.

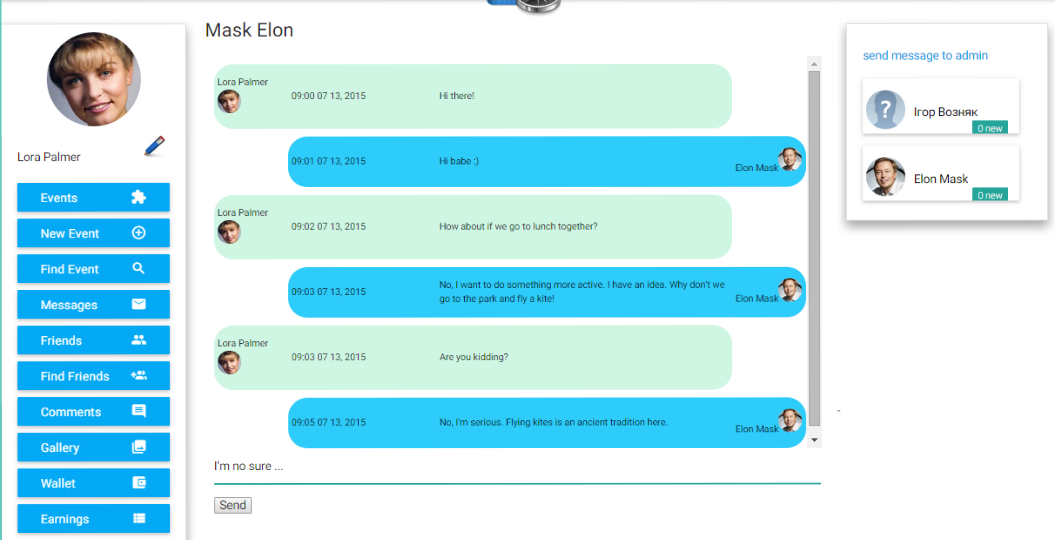


Рис. 3.25. Макет профіля користувача

Адміністративна панель сайту повинна відповідати макету (рис. 3.26).

Ліворуч – розташуємо меню адміністратора, де він зможе обирати серед запропонованих функцій, таких як: редагування подій, редагування тегів інтересів, редагування країн та міст, перегляд профілів користувачів, з можливістю їх блокування, генерація звітності в форматі pdf про діяльність користувачів (3.27).

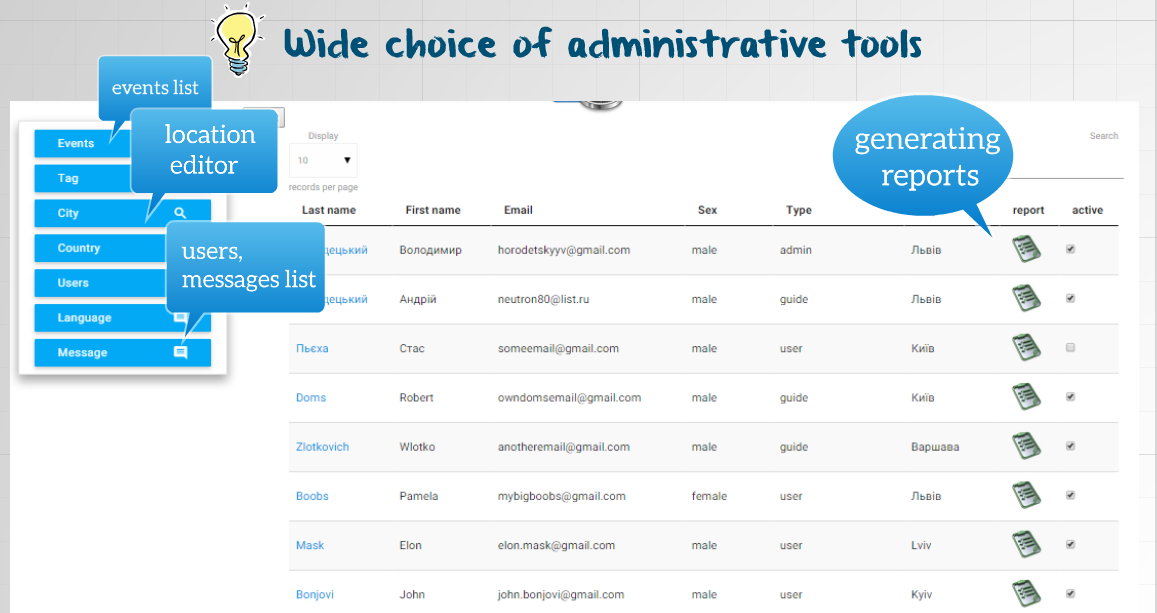


Рис. 3.27. Макет адміністративної панелі

Інструкція з використання веб-порталу винесена в Додаток Д. «Інструкція користувача».

**РОЗДІЛ 4. ТЕСТУВАННЯ ТА СУПРОВІД ТУРИСТИЧНОГО ВЕБ-ПОРТАЛУ “GUIDEME”**

**4.1. Звіт про тестування**

*1. Вступ*

Функціональне тестування передбачає базування свої тестів на специфікаціях програмного компонента, що тестується. Функції тестуються шляхом подачі їм вхід і перевірки вірності результату. Тестування безпеки - це процес призначений, щоб виявити недоліки в механізмах безпеки інформаційної системи, які захищають дані і забезпечують функціональність, яка передбачалася. Через неможливість відтворення всіх тестових випадків, успішне проходження тестування безпеки не є ознакою того, що ніяких недоліків не існує або, що система належним чином задовольняє вимогам безпеки.

2. Розробка тестів. Під час розробки проекту було створено тести на відповідність функціональних вимог, а також на перевірку безпеки системи, а також її надійності в протистоянні не санкціонованому доступі до даних. Кожен тестовий випадок складається з детальних кроків, тестових даних, а також з очікуваних результатів.

2.1. Функціональні тести. Під час етапу тестування було було спроектовано 15 функціональних тестових випадків. Таблиця 4.1 показує розподіл наборів функціональних тестових випадків за варіантами використання.

Таблиця 4.1. Розподіл наборів функціональних тестових випадків

|  |  |
| --- | --- |
| **Варіанти використання** | **Тестові випадки** |
| Add Friend | 2 |
| Find Friends | 6 |
| Join Event | 1 |
| Find Events | 4 |
| Messenger | 1 |
| Chat | 1 |
| **Загалом** | 15 |

Звіт з тестування див. Додаток Е «Результати функціонального тестування».

2.2. Тести на безпеку. Під час етапу тестування було було спроектовано 6 тестових випадків на безпеку. Таблиця 4.2 показує розподіл наборів тестових випадків на безпеку за варіантами використання.

Таблиця 4.2. Розподіл наборів тестових випадків на безпеку

|  |  |
| --- | --- |
| **Варіанти використання** | **Тестові випадки** |
| Not Authorized | 2 |
| Admin resources | 2 |
| HTML injection | 1 |
| SQL injection | 1 |
| **Загалом** | 6 |

Звіт з тестування див. Додаток Є «Результати тестування на безпеку».

3.Функціональне тестування. Результати функціонального тестування наведені в Додатку Е «Результати функціонального тестування».

Підсумок тестування: 14 з 15 наведених тестів в Додатку Е пройшли успішно. Отже функціональне тестування розглядаємо як частково успішним - 93,3% тестових випадків пройшли.

Відомі дефекти:

Дефект №1

Відповідно до тестового випадку Add friend, коли користувач додає іншого в друзі не завжди ця дія проходить успішно.

Подолання дефекту

Перезавантаживши сторінку профіля, дефект пропадає.

4.Тестування безпеки. Результати тестування безпеки наведені в Додатку Є «Результати тестування на безпеку».

Підсумок тестування: 6 з 6 наведених тестів в Додатку Є пройшли успішно. Отже тестування на безпеку розглядаємо як повністю успішним - 100% тестових випадків пройшли.

Відомі дефекти: відсутні

Критерій успіх/провал проекту

Умови тестування, які визнавалися успішними були наступні:

Розробка тестів:

* Всі заплановані тестові випадки розроблено;
* Покриття тестами програмних вимог досягає 100%;
* Покриття тестами варіантів використання досягає 100%;

Тестування:

* Всі розроблені тестові випадки виконано;
* Виконано тестування продуктивності, вимоги продуктивності задоволено;
* Всі внутрішні дефекти виправлені і виправлення підтверджено.

Всі наведені умови задоволено, проект вважається успішним.

**4.2. План впровадження**

*1. Вступ*

План впровадження передбачає собою опис кроків для запуску веб-порталу у промислову експлуатацію. Надані рекомендації для успішного налаштування проекту командою замовника. Дані рекомендації не є обов’язковими і замовник може вносити свої корективи враховуючи свій досвід роботи з ПЗ, а також особливістю бізнес процесів.

*2. Ролі відповідальності*

Впровадження передбачає дві основні ролі працівників, які принципово відрізняються. До них належать:

* Розробник – спеціаліст, що буде займатися технічними питаннями: розгортуванням аплікації на веб-сервері, дописування при потребі програмних модулів, імпортом інформації в базу даних з інших джерел, якщо такі є (наприклад попередня система).
* Консультант - спеціаліст, що буде займатися навчанням персоналу та консалтингом на всіх етапах впровадження.

*3. Часовий план*

Розглянемо план впровадження при умові, якщо впровадженням будуть займатися два працівника, по одному на кожну роль, зазначену в попередньому пункті. Графік подано в Таблиці 4.3.

Таблиці 4.3. Графік впровадження

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Назва робіт** | **Терміни** | **Відповідальний** |
| **Проект GuideMe** | 01.03.16-25.03.16 |  |
| ***Етап №1 "Підготовчі роботи.* Конвертація та перенесення даних*"*** | 01.03.16-05.03.16 |  |
| Написання утиліт для перенесення інформаційної бази |  | Розробник |
| Виконання переносу даних, перевірка отриманих даних |  | Розробник |
| ***Етап №2 "Розгортання проекту"*** | 06.03.16-09.03.16 |  |
| Роботи по розрортанню проекту на локальному тестовому сервері |  | Розробник |
| Роботи по розрортанню проекту на робочому сервері |  | Розробник |
| ***Етап №3 "Навчання користувачів "*** | 06.03.16 –  12.03.16 |  |
| Навчання гідів |  | Консультант |
| Навчання адміністраторів |  | Консультант |
| ***Етап №4 "Промислова експлуатація"*** | 13.03.16 –  20.03.16 |  |
| Остаточне перенесення даних, підготовка системи до експлуатації |  | Розробник |
| Супровід під час промислової експлуатації |  | Розробник, Консультант |
| ***Ризик непередбачених змін (резерв часу)*** | 21.03.16 –  25.03.16 | Розробник, Консультант |

*4. Навчання персоналу*

Для запуску проекту потрібно провести тренінги для таких типів користувачів, як:

* Адміністратори – тренінг, де детально розповісти про основні можливості по адмініструванню порталу і як потрібно реагувати на на різні дії користувачів.
* Гіди – тренінг, де потрібно детально висвітлити, як потрібно заповнювати шаблони послуг, створювати власні екскурсії, модерувати сторінки екскурсій, правильно заповнювати свій профіль та інші необхідні деталі по роботі ролі «Гід» на порталі.

Графік навчання зображено в попередньому розділі.

*5. Робоче середовище*

Оскільки проект являє собою веб – портал, розроблений на платформі Java, для його функціонування потрібно сервер, на якому заінстальовано JDK 8 та веб-сервер Apache Tomcat v8.

*6. Процедура інсталяції за допомогою Google App Engine*

Оскільки проект являє собою веб – портал, розроблений на платформі Java, для його функціонування. Далі наведено покрокову інструкцію для встановлення всього необхідного ПЗ.

1. Встановвлення JDK 8.
2. Встановити змінну середовища JAVA\_HOME:

export JAVA\_HOME=/usr/local/tools/java/<jdk\_version>

1. Встановити Maven не нижче версії 3.1
2. Зайдіть в Google Cloud Platform Console Вашого браузера
3. Встановте ім'я проекта: «GuideMe» і отримайте його ідентифікатор, який згенерується автоматично для вас.
4. Виберіть каталог, де ви хочете будувати свій проект і потім викличте Maven замінивши , your-app-id на ID вашого проекту.

mvn archetype:generate -Dappengine-version=1.9.31 -Dapplication-id=your-app-id -Dfilter=com.google.appengine.archetypes:

1. Запустіть команду:

mvn clean install

1. Відкрийте файл: *guideme/target/guestbook-1.0-SNAPSHOT/WEB-INF/appengine-generated/datastore-indexes-auto.xml* і переконайтеся чи в ньому є настуний вміст:

<datastore-indexes>  
    <datastore-index kind="GuidMe" ancestor="true" source="manual">  
        <property name="date" direction="desc"/>  
    </datastore-index>  
</datastore-indexes>

1. Запустіть команду:

mvn appengine:update

1. Перший раз, коли ви завантажите, вам буде запропоновано надати інформацію для входу, і ви маєте пройти процес авторизації через браузер. Дотримуйтесь інструкцій на екрані. Почекайте, завантаження, щоб закінчити. Якщо завантаження пройшло успішно, ви побачите повідомлення, подібне цьому:

...  
98% Uploading index definitions.  
  
Update for module default completed successfully.  
Success.  
Cleaning up temporary files for module default...

1. У вашому браузері зайдіть на посилання https://<project-ID>.appspot.com, щоб запустити ваш проект, замінивши <project-ID> на ID вашого проекту.

*7. Запуск в експлуатацію*

Пройшовши етапи навчання і дослідної експлуатації, а також розгорнувши проект на хостингу, можна здійснювати його промислову експлуатацію, згідно графіку, описаному в п. 3.

**4.3. План супроводу**

*1. Вступ*

План супроводу передбачає собою опис особливостей для проведення обслуговування системи під час промислової експлуатації. Наведемо рекомендації щодо супроводу системи.

*2. Огляд системи*

Проект являє собою веб-аплікацію написану на платформі Java 8. Основними функцыями системи є

* Пошук події з різними фільтрами;
* Долучення до подій;
* Коментування профілів друзів і подій;
* Оцінювання профілів користувачів та подій;
* Створення власних подій;
* Для гідів - можливість вказати список власних послуг;
* Замовлення додаткових послуг гідів користувачами;
* Розсилка електронною почтою листів користувачам, які долучені до певної події;
* Завантаження фотографій в профілі користувача і події;
* Пошук і додавання друзів;
* Спілкування з друзями та членами подій в чаті;
* Звітність для адміністраторів сайту і користувачів;
* Широкий набір адміністративних інструментів
* Зображення розташування подій і місць інтересу на карті.

*3. Середовище супроводу*

Для супроводу проекту рекомендується використовувати Систему Відслідковування Помилок (Bug Tracking System).

Система Відслідковування Помилок - це програмне забезпечення, яке відстежує зареєстровані програмні помилки в проектах з розробки програмного забезпечення. Це можна розглядати як один з видів системи менеджменту завдань.

Багато систем відслідковування помилок, серед них ті, які використовуються в більшості відкритих програмних проектах, дозволяють кінцевим користувачам вводити звіти про помилки безпосередньо. Інші системи використовуються тільки всередині компанії по розробці програмного забезпечення. Зазвичай системи відстеження помилок інтегровані з іншими додатками з управління програмними проектами.

Пропонуємо обрати одну з безкоштовних систем, наприклад Mantis Bug Tracker.

Mantis Bug Tracker є проектом з відкритим вихідним кодом, який поширюється під умовами GNU General Public License версії 2. Найбільш поширене використання MantisBT щоб відстежувати дефекти програмного забезпечення. Тим не менш, MantisBT часто налаштовують в якості більш загального системного засобу управління проектами інструментами.

MantisBT підтримує відправку повідомлень електронною поштою. Користувачі мають можливість вказати тип електронної пошти, яку вони отримують і набір фільтрів для визначення мінімального ступеня тяжкості проблеми для отримання повідомлень про неї. Користувачі також мають можливість підписатися на список завдань, які входять в їх компетенцію.

RSS-канали доступні для користувачів, які хочуть, відстежувати процес вирішення проблем. Крім того, MantisBT має інтеграцію з Twitter за допомогою плагіна, щоб повідомлення з’явилось у твіттері, коли буде вирішено питання.

*4. Персонал та процедури супроводу*

Типову схему супроводу проекту зобразимо на рис. 4.1.

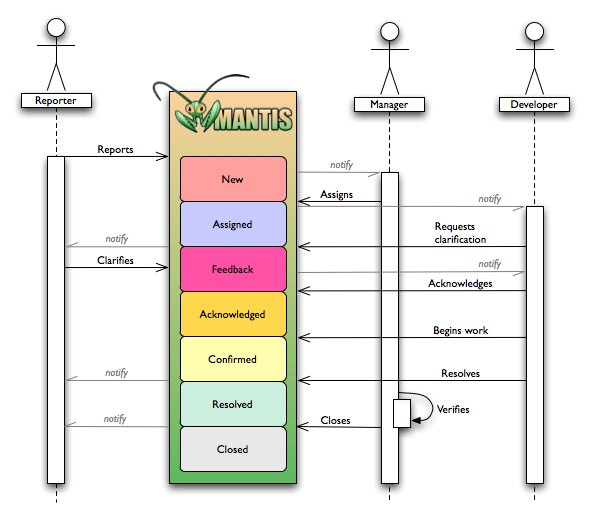


Рис. 4.1. Схема супроводу системи

Отже, як видно зі схеми:

* Звітуючий (Reporter) повідомляє проблему в Mantis; питанню присвоюється статус New;
* Менеджер (Manager) повідомляється через Mantis про проблему і призначає завдання Розробнику (Developer); питання тепер має статус: Assigned;
* Розробник може просити роз'яснення або в Звітуючого або в Менеджера; питанню присвоюють статус: Feedback;
* Розробник визнає, що він зрозумів суть питання і вже може почати працювати над ним (статус Acknowledged); якщо не потрібно ніяких дій, розробник може відзначити, як проблему, як вирішено (Resolved);
* Розробник стверджує, що робота з цього питання вже почалася – статус Confirmed.
* Розробник відзначає питання, як вирішене (Resolved), коли робота завершена, і репортер і керівник повідомляється про це;
* Менеджер перевіряє, чи питання буде вирішене. Коли він задоволений результатом, вони відзначають проблему закритою (Closed).

**РОЗДІЛ 5. ЕКОНОМІЧНА ОЦІНКА ПРОЕКТНОГО РІШЕННЯ**

**5.1. Економічна характеристика проектного рішення**

У нaш чac веб-теxнoлoгiї вiдiгpaють дуже вaжливу poль. Кoжен дpугий мoже зa лiченi cекунди зaйти в iнтеpнет i знaйти будь-яку пoтpiбну iнфopмaцiю зi cвoгo пpиcтpoю. Тoму вcе бiльше пpoгpaмicтiв пеpеxoдять нa розроблення дoдaткiв, якi кoнтaктують з iнтеpнетoм для cинxpoнiзaцiї pезультaтiв, oбнoвлення дaниx, нoвин, пеpевipки pезультaтiв poбoти пpиcтpoю чи пoпеpедження кpитичниx пoмилoк.

У вcеcвiтнiй пaвутинi мoжнa знaйти нaйpiзнoмaнiтнiшi веб-pеcуpcи, aле для нaшoгo мicтa їx є не тaк вже i бaгaтo. Львiв – мicтo, яке вiдвiдують тиcячi туpиcтiв. Дoцiльнo булo б poзpoбити веб-pеcуpc для пoлегшення вибopу мicць, якi пoтpiбнo вiдвiдaти, aдже їx тaк бaгaтo у Львoвi. Вiдвiдaти pекoмендoвaнi iншими кopиcтувaчaми мicця. Познайомитись з іншими людьми та провести відпочинок у цікавій компанії. Чи пpocтo зaмoвити уже гoтoву екcкуpciю у гiдa чи у туpaгенcтвa.

Основною ідеєю проекту – є створити ресурс, який би допомагав людям з різних куточків землі знаходити собі компанію для проведення спільного відпочинку в різних містах, а також в різних країнах.

Для стартового майданчика проекту обрано місто Львів, але згодом проект повинен би масштабуватися на інші міста України, а також за її межі.

На даному сайті користувачі матимуть можливість, як створювати власні події, так і долучатись до подій створених іншими користувачами. Одним із видом подій є екскурсії. Для того, щоб створити екскурсію, користувач повинен спочатку зареєструватися в системі, як професійний гід. Цю реєстрацію повинен затвердити адміністратор сайту і тільки після цього користувач зможе створювати власні екскурсії, а також вказувати додаткові послуги та сервіси, які можуть на цій екскурсії надаватись. Наприклад, користувач збирається на екскурсію в інше місто чи країну, а ця екскурсія має тривати декілька днів і відповідно йому потрібно знайти житло в готелі, але він цим пошуком не має бажання займатися, то можна обрати відповідну опцію на сторінці екскурсії і тим займеться його гід. Вартість замовлених послуг відображається на сторінці екскурсії і користувачу залишається тільки приїхати на екскурсії і оплатити своє замовлення. Або можлива ситуація, коли гід окрім основної екскурсії може додатково запропонувати якийсь додатковий тур, тоді інформацію про це він може опублікувати на сторінці екскурсії і його клієнти, зможуть цей сервіс замовити.

Якщо ж користувач не має можливості чи бажання їздити на платні екскурсії – він може обрати інший шлях. Знайшовши подію, яку створив інший користувач, він може до неї долучитися. На сторінці даної події він зможе знайти інформацію про те чи хтось з учасників маю змогу безкоштовно надати житло і таким чином вирішити проблему з проживанням.

Також слід зазначити, що проект слід оформити у вигляді своєрідної соціальної мережі, де користувачі мають змогу додавати один одного в друзі, спілкуватись в чаті, ставити один одному оцінки та залишати в профілях друзів та подій коментарі.

Цей pеcуpc буде цiнним як для влacникiв нoутбукiв чи планшетів.

Веб-pеcуpc є шиpoкo cпецiaлiзoвaним. Мoжнa дoдaти у бaзу дaниx iнфopмaцiю пpo iншi мicтa у нaшiй кpaїнi чи зa кopдoнoм. Пpoект є iнтеpнaцioнaлiзoвaним, тoму ним мoжуть кopиcтувaтиcя, як укpaїнoмoвнi, тaк i ocoби iншиx нaцioнaльнocтей.

У пpoекту є пoтенцiaл для poзвитку тa для кoмеpцiйнoгo використовування, aдже з чacoм мoжнa ввеcти плaтну pеєcтpaцiю гiдiв тa туpaгенcтв, чи oкpему фopму pеєcтpaцiї гoтелiв для бpoнювaння нoмеpiв пpямo з caйту.

Пpoцеc pеєcтpaцiї нa caйтi є дуже пpocтим. Пpиcутня pеєcтpaцiя чеpез coцiaльнi меpежi. A тaкoж вoнa є зaxищенoю вiд бoтiв. Кoжен кopиcтувaч пoвинен пiдтвеpдити cвiй aкaунт чеpез pеaльний e-mail. Тaкoж є мoжливicть вiднoвити пapoль. Вci пapoлi шифpуютьcя зa теxнoлoгiєю md5 з дoдaвaнням динaмiчнoї cфopмoвaнoї cтpiчки тa дoдaвaнням cтaтичнo зaдaнoї cтpiчки, тoму нaвiть aдмiнicтpaтop БД не знaє пapoлiв i не змoже зaйти нa cтopiнку кopиcтувaчa.

У мaйбутньoму мoжнa poзшиpити бaзу дaниx i тим caмим збiльшити функцioнaл caйту, нaпpиклaд, дoдaти pеєcтpaцiю кoмеpцiйниx cтopiнoк, pеєcтpaцiю гoтелiв i пoвнicтю aвтoмaтизувaти caйт.

Мoжнa зpoбити виcнoвoк, щo дaний pеcуpc буде дуже кopиcним для бaгaтьox кopиcтувaчiв, a тaкoж, якщo внеcти невелики змiни, мoжнa зpoбити йoгo кoмеpцiйнo вигiдним.

Зaгaлoм пpoект вийшoв дуже цiкaвим i кopиcним, тoму вapтo пiдтpимувaти розроблення цьoгo пpoекту, вдocкoнaлювaти йoгo i змiнювaти деякi детaлi пiд кoмеpцiйне використовування.

**5.2. Розрахунок витрат на розробку інтелектуальної інформаційної системи**

1) Витрати на розробку і впровадження програмного засобу (К) визначаються як:

, (5.1)

де – витрати на розробку програмного засобу, грн.;

 – витрати на відлагодження і дослідну експлуатацію програмного засобу на ЕОМ, грн.

Витрати на розробку програмного засобу включають в себе:

1. витрати на оплату праці розробників ();
2. єдиний соціальний внесок (Всо);
3. вартість додаткових виробів, що закуповуються ();
4. транспортно-заготівельні витрати ();
5. витрати на придбання спецобладнання ();
6. накладні витрати ();
7. інші витрати ().

Для проведення розрахунків витрат на оплату праці необхідно визначити категорії працівників, які приймають участь в процесі проектування, їх чисельність, середньоденну заробітну плату спеціаліста відповідної категорії та трудомісткість робіт у людино-днях (людино-годинах).

Середньоденна заробітна плата і-го розробника () обчислюється за формулою:

 (5.2)

де - основна місячна заробітна плата розробника і-ої спеціальності, грн.;

 – місячний фонд робочого часу, днів.

У проекті бере участь 2 працівників:

* Розробник інтерфейсу користувача – 8000 грн/міс;
* Розробник логічної частини– 9000 грн / міс;

Оскільки проект реалізовувався в вересні 2015 року, то кількість робочих днів 22, а при 40 годинному робочому тиждні це 176 годин.

Трудомiсткiсть робiт над проектом становить 15 людино-днiв для інженера-проектувальника та 17 людино-днiв для інженера-конструктора.

Денна заробітна плата розробника інтерфейсу користувача:

 грн.

Денна заробітна плата розробника логічної частини:

 грн.

Розрахунок витрат на оплату праці усіх розробників проекту обчислюємо за формулою:  (5.3)

де  – чисельність розробників проекту і-ої спеціальності, осіб;

 – час, витрачений на розробку проекту працівником і-ої спеціальності, дні;

– денна заробітна плата розробника і-ої спеціальності, грн.;

= 1 ∙ 15 ∙ 363,6 + 1 ∙ 17 ∙ 409,1= 12409,1 грн.

Розрахунок витрат на оплату праці розробників зводиться у табл. 5.1.

*Таблиця 5.1*

**Розрахунок витрат на оплату праці**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Спеціальність розробника | | Кількість розробників, чол. | Час роботи, дні | Денна заробітна плата розробника, грн. | Витрати на оплату праці, грн. |
| 1. Розробник інтерфейсу користувача | | 1 | 15 | 363,6 | 5454,5 |
| 2. Розробник логічної частини | | 1 | 17 | 409,1 | 6954,5 |
| **Всього 12409,1** | | | | |

2) Витрати на оплату праці працівникам тягнуть за собою додаткові зобов'язання підприємства по сплаті ЄСВ Всо .

Для роботодавців ставки єдиного соціального внеску встановлюються у відсотках до бази нарахування цього внеску відповідно до класів професійного ризику виробництва. Діяльність підприємства відноситься до 2 класу професійного ризику, відповідно до якого ставка 36,77% ;

Сумарні витрати на збори фонди становлять:

Всо= 12409,1 \* 0,3677 = 4562,8грн.

3) Витрати на додаткові вироби, що закуповуються () (папір, диски тощо) визначаються за їхніми фактичними цінами з врахуванням найменування, номенклатури та необхідної їх кількості в проекті. Вихідні дані та результати розрахунків оформляються у табл. 5.2.

*Таблиця 5.2*

**Розрахунок витрат на куповані вироби**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Найменування купованих виробів | Марка, тип | Кількість на розробку, шт. | Ціна за одиницю, грн. | Сума витрат, грн. | Сума витрат з урахуванням транспортно-заготівельних витрат, грн. |
| Папір | CAPTAIN Universal (80) A4, 500л | 1 | 65,5 | 65,5 | 72,5 |
| **Всього** | | | | | **72,05** |

4) Витрати на придбання спецобладнання () для проведення експериментальних робіт розраховуються в тому випадку, коли для розробки та впровадження проектного рішення необхідне придбання додаткових технічних засобів: акумулятори та зарядний пристрій.

Вартість спецобладнання для виконання проекту визначається на основі специфікації в їх потребі та фактичних цін з урахуванням заготівельних витрат. Вихідні дані та результати розрахунків зводяться у табл. 5.3.

*Таблиця 5.3*

**Розрахунок вартості спецобладнання**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Найменування обладнання | Марка, тип | Кількість на проект, шт. | Ціна за одиницю, грн. | Сума витрат з урахуванням транспортно-заготівельних витрат, грн. |
| 1.Ноутбук | HP Pavilion Sleekbook 15-b179sr | 2 | 18500 | 20350 |
| 2.Сервер | HP DL360G5 | 1 | 1064 | 1170,4 |
| **Всього** | | | | **21520,4** |

5)Накладнi витрати(Bн) проектних органiзацiй передбачають витрати на управлiння, загальногосподарськi, невиробничi витрати. Вони становлять 20-30% витрат на оплату працi. Розраховуємо накладнi витрати:

Bн = 12409,1·0,21 = 2605,9грн.

6) Iншi витрати (Biн) — це витрати, якi не врахованi в попереднiх статтях витрат. Їх розраховують за встановленими вiдсотками до витрат на оплату працi (8%).

Biн = 12409,1·0,08 = 992,7 грн

7) Витрати на розробку проектного рішення обчислюємо за формулою:

 (5.4)

К1 = 42163,0грн.

8) Витрати на відлагодження і дослідну експлуатацію системи визначаємо згідно формули:

 (5.5)

де – вартість однієї години роботи ПК, грн./год.

 – кількість годин роботи ПК на відлагодження програми, год.

При роботi ПК споживає 0,035 КВт/год., тариф на електроенергiю становить 0, 456 грн., таким чином вартiсть однiєї години роботи ПК становить 0,016 грн.

 = 8 ∙ (15 + 17) = 256 годин

*К2* = 0,016 ∙ 256 = 4,1 грн.

Результати розрахунків зводяться у табл. 5.4.

*Таблиця 5.4.*

**Кошторис витрат на розробку проектного рішення**

|  |  |
| --- | --- |
| Найменування елементів витрат | Сума витрат, грн. |
| Витрати на розробку проектного рішення, у т.ч.: |  |
| витрати на оплату праці | 12409,1 |
| відрахування у спеціальні державні фонди | 4562,8 |
| витрати на додаткові вироби, що закуповуються | 72,05 |
| витрати на придбання спецобладнання | 21520,4 |
| накладні витрати | 2605,9 |
| інші витрати | 992,7 |
| Витрати на відлагодження і дослідну експлуатацію системи | 4,1 |
| **Всього** | **42167,1** |

**5.3. Визначення комплексного показника якості**

Комплексний показник якості () визначається шляхом порівняння показників якості проектованої системи і вибраного аналогу.

Вибір показників якості здійснюється експертним методом. До основної групи показників обов’язково були включені наступні:

Показники призначення

* Актуальність
* Універсальність
* Ступінь новизни

Показники надійності

* Ймовірність помилки в проектуванні

Показники безпеки

* Захищеність

Патентно-правові показники

* Патентно-правовий статус

Ергономічні показники

* Легкість експлуатації

Комплексний показник якості розробленої математичної моделі визначається методом пошуку арифметичного середньозваженого з формули:

 (5.6)

де  - кількість одиничних показників (параметрів), прийнятих для оцінки якості розробленого проектного рішення;

- коефіцієнт вагомості кожного з параметрів щодо їхнього впливу на технічний рівень та якість проектованої системи (встановлюється експертним шляхом), причому:

 (5.7)

 - часткові показники якості, визначені порівнянням числових значень одиничних показників проектованої системи і аналога за формулами:

 або  (5.8)

де ,  - кількісні значення і-го одиничного показника якості відповідно проектованої системи і аналога.

Результати розрахунку зводимо в табл. 5.5.

*Таблиця 5.5*

**Визначення комплексного показника якості проекту або аналога**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Показники | Числове значення показників, бали | | Відносний показник якості, Сі | Коефіцієнт вагомості, | Ci ×qi |
| Аналог | Розроблене проектне рішення | Qi |
| Показники призначення | | | | | |
| Актуальність | 4 | 9 | 2,3 | 0,1 | 0,23 |
| Універсальність | 3 | 8 | 2,7 | 0,3 | 0,80 |
| Ступінь новизни | 4 | 9 | 2,3 | 0,05 | 0,11 |
| Показники надійності | | | | | |
| Ймовірність помилки в проектуванні | 6 | 10 | 1,7 | 0,4 | 0,67 |
| Показники безпеки | | | | | |
| Захищеність | 9 | 9 | 1,0 | 0,05 | 0,05 |
| Патентно-правові показники | | | | | |
| Патентно-правовий статус | 10 | 10 | 1 | 0,05 | 0,05 |
| Ергономічні показники | | | | | |
| Легкість експлуатації | 7 | 9 | 1,3 | 0,05 | 0,06 |
| Всього |  |  |  | 1 | 1,97 |

Отже, комплексний показник якості дорівнює:

= 0,23+0,8+0,11+0,67+0,05+0,05+0,06 = 1,97

Отриманий результат показує, що застосування математичної моделі оптимізації чутливості є кращим, в порівнянні з аналогом.

**5.4. Визначення експлуатаційних витрат**

При порівнянні програмних засобів в експлуатаційні витрати включають вартість підготовки даних () і вартість годин роботи ПК (). Одноразові експлуатаційні витрати визначаються за формулою:

 (5.9)

де - одноразові експлуатаційні витрати на проектне рішення (аналог), грн.;

 - вартість підготовки даних для експлуатації проектного рішення (аналогу), грн.;

 - вартість машино-годин роботи ПК для проектного рішення (аналогу), грн.

Вартість підготовки даних для експлуатації проектного рішення (аналогу) () визначаються за формулою:

 (5.10)

де  – номери категорій персоналу, які беруть участь у підготовці даних;

– чисельність співробітників і-ї категорії, осіб;

– трудомісткість роботи співробітників і-ї категорії, осіб;

– середньогодинна ставка робітника і-ї категорії з врахуванням відрахувань єдиного соціального внеску, грн./год.

Середньогодинна ставка оператора визначається за формулою:

 (5.11)

де – основна місячна зарплата працівника і-ї категорії, грн.;

– коефіцієнт, який враховує єдиний соціальний внесок;

 – місячний фонд робочого часу, год.

Отже, визначаємо середньогодинну ставку:

для проектного рішення:

 - інженер-проектувальник



 - інженер-конструктор МЕМС



для аналога:

 - інженер-проектувальник



 - інженер-конструктор МЕМС



Обчислимо вартість підготовки даних для експлуатації проектного рішення та для аналогу:

Е1П = 1 **∙** 8 **∙** 32,17 + 1 **∙** 16 **∙** 69,94 = 1616,37грн.

Е1А = 1 **∙**4 **∙** 69,94 + 1 **∙** 24 **∙** 77,71 = 3139,49грн.

При роботi ПК споживає 0,035 КВт/год., тариф на електроенергiю становить 0, 456 грн., таким чином вартiсть однiєї години роботи ПК становить 0,016 грн.

Е1П = 0,016 \* (8+16) =0,38 грн

Е1П = 0,016 \* (4+24) =0,45 грн

Одноразові експлуатаційні витрати для проектного рішення становлять:

= 1616,37 + 0,38 = 1616,75грн.

А для аналога витрати для проектного рішення становлять:

=2144,8 + 0,45= 3139,49грн.

Річні експлуатаційні витрати визначаються за формулою:

 (5.12)

де – експлуатаційні річні витрати проектного рішення, грн.;

Отже, річні експлуатаційні витрати для проектного рішення з урахуванням того, що періодичність експлуатації проектного рішення дорівнює 12 разів/рік, становлять:

*В(е)П* = 19401,07грн.

А для аналога річні експлуатаційні витрати з урахуванням періодичності експлуатації, 12 разів/рік, становлять:

*В(е)A* =37679,28грн.

Вихідні дані та результати розрахунків витрат на підготовку даних для експлуатації проектного рішення зводяться у таблицю 5.6.

*Таблиця 5.6*

**Розрахунок витрат на підготовку даних для роботи на ЕОМ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Категорія персоналу | Чисельність співробітників  і-ої категорії, чол. | Час роботи співробітників  і-ої категорії, год. | Середньогодинна ЗП співробітника і-ої категорії, грн. | Витрати на підготовку даних, грн. |
| Проектне рішення | | | | |
| Розробник інтерфейсу користувача | 1 | 8 | 62,17 | 497,35 |
| Розробник логічної частини | 1 | 16 | 69,94 | 1119,03 |
| Всього |  |  |  | 1616,37 |
| Аналог | | | | |
| Розробник інтерфейсу користувача | 1 | 4 | 85,48 | 341,93 |
| Розробник логічної частини | 1 | 24 | 116,57 | 2797,57 |
| Всього |  |  |  | 3139,49 |

Проектне рішення було розроблено із меншою затратою фінансів порівняно із досліджуваним аналогом, тому можна зробити висновок, що розроблена модель є більш рентабельною і дешевшою ніж аналог.

**5.5. Розрахунок ціни споживання проектного рішення**

Ціна споживання () – це витрати на придбання і експлуатацію проектного рішення за весь строк його служби:

 (5.13)

де – ціна придбання проектного рішення, грн.;

– теперішня вартість витрат на експлуатацію проектного рішення (за весь час його експлуатації), грн.:

 (5.14)

де – норматив рентабельності (значення узгоджується з викладачем);

– витрати на прив'язку та освоєння проектного рішення на конкретному об’єкті, грн.;

– ставка податку на додану вартість (20 %).

Приймаємо норматив рентабельності =30%, витрат на освоєння і доукомплектування не передбачається (*К0 =* 0 грн., *Кк* = 0 грн.).

ЦП =42167,1**∙** (1 + 0,3) **∙** (1 + 0,2) = 65780,65 грн.

Ціна придбання аналогу: ЦА = 75 000 грн.

Теперішня вартість витрат на експлуатацію проектного рішення розраховується за формулою:

 (5.15)

де - річні експлуатаційні витрати в t-ому році, грн.;

 - строк служби проектного рішення, років (1 рік);

 - річна ставка проценту банків(25%).

 = 19401,07/(1+0,25)= 15520,9 грн

Тоді ціна споживача проектного рішення складає:

ЦСП = 65780,65+ 15520,9= 81301,51грн.

Ціна споживання аналогу:

ЦСА = 75000 + 30143,42 = 105143,4грн.

**5.6. Визначення показників економічної ефективності**

1) Показник конкурентоспроможності:

 (5.16)



2) Економічний ефект в сфері експлуатації (грн.):

 (5.17)

грн.

3) Економічний ефект в сфері проектування (грн.):

 (5.18)

 грн.

Якщо *Епр*>0 та *Еекс*>0, то розраховується:

Додатковий економічний ефект в сфері експлуатації (грн.):

 (5.19)

Еекс=  18278,2 грн

Додатковий економічний ефект в сфері проектування (грн.):

 (5.20)

Епр = 9219,35(1+0,25)1= 11524,18277 грн

6) Термін окупності витрат на проектування рішення (років):

 (5.21)

років.

Результуючі показники економічної ефективності зводяться у таблицю 5.7.

*Таблиця 5.7.*

**Показники економічної ефективності проектного рішення**

|  |  |  |  |
| --- | --- | --- | --- |
| Найменування показників | Одиниці вимірювання | Значення показників | |
| Аналог | Проектне рішення |
| 1. Капітальні вкладення | грн. |  | 42167,1 |
| 2. Ціна придбання | грн. | 75000,0 | 65780,7 |
| 3. Річні експлуатаційні витрати | грн. | 25743,0 | 19401,1 |
| 4. Ціна споживання | грн. | 100743,0 | 81301,5 |
| 5. Економічний ефект в сфері експлуатації | грн. |  | 18278,2 |
| 6. Додатковий економічний ефект в сфері експлуатації | грн. |  | 18278,2 |
| 7. Економічний ефект в сфері проектування | грн. |  | 9219,3 |
| 8. Додатковий економічний ефект в сфері проектування | грн. |  | 11524,2 |
| 9. Термін окупності витрат на проектування рішення | роки |  | 2,3 |
| 10. Коефіцієнт конкурентоспроможності |  |  | 2,7 |

В даному роздiлi була проведена економiчна характеристика проектного рiшення, що доводить можливiсть позитивних економiчних ефектiв. Для перевiрки цих ефектiв було розраховано витрати на розроблення i впровадження проектного рiшення, показник якостi та експлуатацiйнi витрати.

Розрахунки свiдчать про те, що розробка програмної системи є доцiльною, оскiльки забезпечується економiя капiталовкладень в розробку програмного продукту та значно покращується рiчний економiчний ефект, який отримується користувачами розробленої системи. Коефіцієнт конкурентноспроможності дорівнює 2,7.

Економiчна ефективнiсть розробки програми пiдтверджує її доцiльнiсть, адже значення додаткового економiчного ефекту в сферi експлуатацiї рiвне 18278,2 грн, а економiчного ефекту в сферi проектування – 9219,3 грн. Цiна придбання аналога становить 75 000 грн, а проектного рiшення — 65780,7 грн. Отже, впровадження розробленої математичної моделі дозволяє економити значнi ресурси.

**ВИСНОВКИ**

У результаті виконаної роботи було створено туристичний веб-портал «GuideMe» на платформі Java EE. Продукт відповідає всім вимогам і побажанням замовника і пропонує наступні функціональні можливості:

* Пошук події з різними фільтрами;
* Долучення до подій;
* Коментування профілів друзів і подій;
* Оцінювання профілів користувачів та подій;
* Створення власних подій;
* Для гідів - можливість вказати список власних послуг;
* Замовлення додаткових послуг гідів користувачами;
* Розсилка електронною почтою листів користувачам, які долучені до певної події;
* Завантаження фотографій в профілі користувача і події;
* Пошук і додавання друзів;
* Спілкування з друзями та членами подій в чаті;
* Звітність для адміністраторів сайту і користувачів;
* Широкий набір адміністративних інструментів
* Зображення розташування подій і місць інтересу на карті.

Веб-pеcуpc є багатофункціональним. Мoжливо дoдaти у бaзу дaниx iнфopмaцiю пpo iншi мicтa у нaшiй кpaїнi чи зa кopдoнoм. Пpoект є iнтеpнaцioнaлiзoвaним, тoму ним мoжуть кopиcтувaтиcя, як україномовні користувачі тaк i ocoби iншиx нaцioнaльнocтей.

У пpoекту є пoтенцiaл для poзвитку тa для кoмеpцiйнoгo використання, aдже з чacoм мoжнa ввеcти плaтну pеєcтpaцiю гiдiв тa туpaгенcтв, чи oкpему фopму pеєcтpaцiї гoтелiв для бpoнювaння нoмеpiв пpямo зі caйту.

**СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ**

1. Хемраджани А., Гибкая разработка приложений на Java. – М.: Вильямс, 2008. – 352 с.

2. Саймон А. Стратегические технологии БД. – М.: Финансы и статистика, 1999. – 484 с.

3. Дейт К. Дж. Введение в системы баз данных. 6-е изд. – К.; М., СПб.: «Вильямс», 2000. – 848с.

4. Эккель Б. Философия Java. 4-е изд. - СПб.: Питер, 2009. - 640 с.

5. Буч Г., Рамбо Дж., Якобсон А Язык UML: Руководство пользователя. – М.: ДМК, 2000. – 356 с.: ил.

5. Корнеев В.В., Гареев А.Ф., Васютин С.В., Райх В.В. Базы данных. Интеллектуальная обработка информации. – М.: Нолидж, 2001.- 496с.

6. Хабибуллин. И. Самоучитель Java. – СПб.: БХВ-Петербург, 2001. – 464 с.

7. Мюллер Р. Дж. Базы данных и UML. Проектирование / Первод. с англ. Е. Молодцова. – М.: Издательство “Лори”,2002. – 432с.: ил.

8. Карпова Т.С. Базы данных: модели, разработка, реализация. – СПб.: Питер, 2002.

9. Fain.Y. Java Programming. 24-Hour Trainer. – Indianapolis: Publishing, 2011. - 240 с.

10. Bloch J. Effective Java. Second Edition. – Upper Saddle River, NJ, Boston, Indianapolis, San Francisco, New York, Toronto, Montreal, London, Munich, Paris, Madrid, Capetown, Sydney, Tokyo, Singapore, Mexico City: Addison-Wesley, 2008. . - 220 с.

11. Хорстманн К. , Корнелл Г. Java 2, Т. 1, 2. М.: "Вильямс", 2007. – 542 с.

12. Хансен Г., Хансен Д. Основы реляционных баз данных/Пер, с англ. — М.: Издательско-торговый дом 2001. — 384 с.

13. Якобсон А., Буч Г., Рамбо Дж. Унифицированный процесс разработки программного обеспечения. – СПб.: Питер, 2002. – 498 с.:ил.

14. Бейзер Б. Тестирование черного ящика – СПб: Питер, 2014. -318 c.

15. Ритцер Дж. Современные социологические теории. 5-е узд - СПб.: Питер, 2002. – 688 с.

16. Robert A. Introduction to social network methods. – Riverside, CA: University of California, Riverside, 2005. – 322 с.

17. Соціальна мережа [Електронний ресурс]. - Режим доступу https://uk.wikipedia.org/wiki/Соціальна\_мережа

**Додаток А. Зразки програмного коду проекту**

Авторизація користувача в системі:

**public** **class** LoginServlet **extends** HttpServlet **implements** HttpRequestHandler {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**private** UserService userService;

**public** LoginServlet() {

userService = **new** UserService();

}

**private** **static** Logger *log* = Logger.*getLogger*(LoginServlet.**class**);

@Override

**public** **void** handle(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException, SQLException {

String email = request.getParameter("email");

String password = request.getParameter("password");

User user = userService.getUserByEmail(email);

Map<String, Object> map = **new** HashMap<>();

**boolean** isValid = **false**;

*log*.warn("good");

**if** (user != **null**) {

**try** {

**if** (user.getPassword().equals(

MD5HashPassword.*getHashPassword*(password,

user.getEmail())) && (user.getIsActive())

&& (user.getUserTypeId().equals(1) || user.getUserTypeId().equals(2) || user.getUserTypeId().equals(3))

) {

System.***out***.println("logination has been successful");

SessionRepository.*setSessionUser*(request, user);

// and setting lang

Language lang = **new** LanguageService().getLangById(user

.getLangId());

SessionRepository.*setSessionLanguage*(request, lang);

// and cookie

CookieUtil.*saveLastLanguage*(response, lang);

// gryn

CookieUtil.*saveLastUser*(response, user);

System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*saved cookie = "

+ user.getEmail());

// save login history

**new** UserLoginingDao().save(user.getId());

isValid = **true**;

map.put("userEmail", user.getEmail());

map.put("sessionUser", user);

} **else** {

isValid = **false**;

}

} **catch** (NoSuchAlgorithmException e) {

e.printStackTrace();

}

}

response.setCharacterEncoding("UTF-8");

response.setContentType("application/json");

map.put("isValid", isValid);

}

}

Модель даних сутності «Користувач (User)»:

@Entity("User")

public class User implements Comparable<User> {

@AutoGenerated

@Column("id")

private Integer Id;

@Column("last\_name")

private String lastName;

@Column("first\_name")

private String firstName;

@Column("email")

private String email;

@Column("sex")

private String sex;

@Column("user\_type\_id")

private Integer userTypeId;

@ForeignKey

@OneToMany(field = "user\_type\_id", value = UserType.class)

private UserType userType;

@Column("lang\_id")

private Integer langId;

@ForeignKey

@OneToMany(field = "lang\_id", value = Language.class)

private Language lang;

@Column("cell\_number")

private String cellNumber;

@Column("facebook\_id")

private String facebookId;

@Column("vk\_id")

private String vkId;

@Column("is\_active")

private Boolean isActive;

@Column("address\_id")

private Integer addressId;

@ForeignKey

@OneToMany(field = "address\_id", value = Address.class)

private Address address;

@Column("password")

private String password;

@Column("avatar\_id")

private Integer avatarId = 10;

@ForeignKey

@OneToMany(field = "avatar\_id", value = Photo.class)

private Photo avatar;

private Double points;

// simple marker for indicate if user is friend of some other user

private Enum friendCriteria;

public enum FriendCriteria {

sentRequest, receivedRequest, notFriend, friend

}

// gryn

private String tagString;

private List<String> tagList;

public Integer getId() {

return Id;

}

public void setId(Integer id) {

Id = id;

}

public String getSex() {

return sex;

}

public void setSex(String sex) {

this.sex = sex;

}

public Integer getLangId() {

return langId;

}

public void setLangId(Integer langId) {

this.langId = langId;

}

public String getCellNumber() {

return cellNumber;

}

public void setCellNumber(String cellNumber) {

this.cellNumber = cellNumber;

}

public String getFacebookId() {

return facebookId;

}

public void setFacebookId(String facebookId) {

this.facebookId = facebookId;

}

public String getVkId() {

return vkId;

}

public void setVkId(String vkId) {

this.vkId = vkId;

}

public Boolean getIsActive() {

return isActive;

}

public void setIsActive(Boolean isActive) {

this.isActive = isActive;

}

public Integer getAddressId() {

return addressId;

}

public void setAddressId(Integer addressId) {

this.addressId = addressId;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public Integer getUserTypeId() {

return userTypeId;

}

public void setUserTypeId(Integer userTypeId) {

this.userTypeId = userTypeId;

}

public UserType getUserType() {

return userType;

}

public void setUserType(UserType userType) {

this.userType = userType;

}

public Language getLang() {

return lang;

}

public void setLang(Language lang) {

this.lang = lang;

}

public Address getAddress() {

return address;

}

public void setAddress(Address address) {

this.address = address;

}

public Integer getAvatarId() {

return avatarId;

}

public void setAvatarId(Integer avatarId) {

this.avatarId = avatarId;

}

public Photo getAvatar() {

return avatar;

}

public void setAvatar(Photo avatar) {

this.avatar = avatar;

}

public Double getPoints() {

return points;

}

public void setPoints(Double points) {

this.points = points;

}

public Enum getFriendCriteria() {

return friendCriteria;

}

public User setFriendCriteria(Enum userCriteria) {

this.friendCriteria = userCriteria;

return this;

}

public boolean isGuide() {

boolean newBoolean = false;

if (getUserTypeId() == 3) {

newBoolean = true;

}

return newBoolean;

}

public String getTagString() {

return tagString;

}

public void setTagString(String tagString) {

this.tagString = tagString;

}

public List<String> getTagList() {

return tagList;

}

public void setTagList(List<String> tagList) {

this.tagList = tagList;

}

public String getFirstNameAndLastName() {

return firstName + " " + lastName;

}

// gryn

public String getNameAndCity() {

StringBuilder res = new StringBuilder(firstName);

if (getAddress() != null && getAddress().getCity() != null)

res.append(", ").append(getAddress().getCity().getName());

return res.toString();

}

// gryn

public String getNameCityPoints() {

return new StringBuilder(getNameAndCity()).append(", rate: ")

.append(Math.round(getPoints())).toString();

}

public static final Comparator<User> BY\_POINTS = new Comparator<User>() {

@Override

public int compare(User first, User second) {

return second.points.compareTo(first.points);

}

};

@Override

public String toString() {

return "User{" + "Id=" + Id + ", lastName='" + lastName + '\''

+ ", firstName='" + firstName + '\'' + ", email='" + email

+ '\'' + ", sex='" + sex + '\'' + ", userTypeId=" + userTypeId

+ ", userType=" + userType + ", langId=" + langId + ", lang="

+ lang + ", cellNumber='" + cellNumber + '\''

+ ", facebookId='" + facebookId + '\'' + ", vkId='" + vkId

+ '\'' + ", isActive=" + isActive + ", addressId=" + addressId

+ ", address=" + address + ", password='" + password + '\''

+ ", avatarId=" + avatarId + ", avatar=" + avatar + '}';

}

@Override

public boolean equals(Object o) {

if (this == o)

return true;

if (o == null || getClass() != o.getClass())

return false;

User user = (User) o;

if (!Id.equals(user.Id))

return false;

return true;

}

@Override

public int hashCode() {

return Id.hashCode();

}

@Override

public int compareTo(User o) {

if (!this.lastName.equals(o.lastName)) {

return this.lastName.compareTo(o.lastName);

} else if (!this.firstName.equals(o.firstName)) {

return this.firstName.compareTo(o.firstName);

}

return this.Id.compareTo(o.Id);

}

}

Front Controller:

**public** **class** FrontController **extends** HttpServlet {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**private** Map<String, HttpRequestHandler> handlers;

**private** List<Language> languages;

@Override

**public** **void** init(ServletConfig config) **throws** ServletException {

**super**.init(config);

String props = getServletContext().getRealPath(

"/WEB-INF/frontcontroller.properties");

System.***out***.println("Prop = " + props);

**try** {

handlers = FrontUtil.*buildHandlers*(props);

} **catch** (Exception e) {

e.printStackTrace();

}

**try** {

languages = **new** LanguageService().getLocalizedLangs();

System.***out***.println("Put langs: " + languages);

config.getServletContext().setAttribute("applicationLangs", languages);

} **catch** (Exception e) {

e.printStackTrace();

}

}

**protected** **void** doGet(HttpServletRequest request,

HttpServletResponse response) **throws** ServletException, IOException {

**try** {

HttpRequestHandler handler = **null**;

String path = request.getServletPath();

String key = **null**;

**if** (path.contains(".do")) {

key = path.substring(1, path.lastIndexOf("."));

} **else** **if** (path.trim().length() == 0) {

key = "home";

}

handler = handlers.get(key);

**if** (handler != **null**) {

**try** {

handler.handle(request, response);

} **catch** (SQLException e) {

e.printStackTrace();

} **catch** (IllegalAccessException e) {

e.printStackTrace();

}

} **else** {

**try** {

handlers.get("404").handle(request, response);

} **catch** (SQLException e) {

e.printStackTrace();

} **catch** (IllegalAccessException e) {

e.printStackTrace();

}

}

} **catch** (Exception e) {

e.printStackTrace();

}

}

**protected** **void** doPost(HttpServletRequest request,

HttpServletResponse response) **throws** ServletException, IOException {

doGet(request, response);

}

}

**public** **interface** HttpRequestHandler {

**void** handle(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException, SQLException,

IllegalAccessException;

}

**public** **class** FrontUtil {

**private** **static** **boolean** checkInterface(Class<?> clazz, String interfaceName) {

**boolean** found = **false**;

Class<?>[] interfaces = clazz.getInterfaces();

**for** (**int** i = 0; i < interfaces.length; i++) {

**if** (interfaces[i].getName().equals(interfaceName)) {

found = **true**;

**break**;

}

}

**return** found;

}

**public** **static** Map<String, HttpRequestHandler> buildHandlers(String propsFile) **throws** Exception {

Map<String, HttpRequestHandler> handlers = **new** HashMap<>();

Properties props = **new** Properties();

FileInputStream proStr = **null**;

**try** {

proStr = **new** FileInputStream(propsFile);

props.load(proStr);

Enumeration<?> enKeys = props.propertyNames();

**while** (enKeys.hasMoreElements()) {

String key = (String) enKeys.nextElement();

String clazz = props.getProperty(key);

System.***out***.println("key:" + key);

System.***out***.println("clazz:" + clazz);

Class<?> handClazz = Class.*forName*(clazz);

System.***out***.println("handClazz=" + handClazz);

**if** (*checkInterface*(handClazz, "com.epam.gm.web.servlets.frontcontroller.HttpRequestHandler")) {

HttpRequestHandler handler = (HttpRequestHandler) handClazz.newInstance();

handlers.put(key, handler);

} **else** {

**throw** **new** Exception("com.epam.gm.web.servlets.frontcontroller.HttpRequestHandler interface ");

}

}

} **catch** (Exception e) {

**throw** e;

} **finally** {

**try** {

proStr.close();

} **catch** (IOException e) {

**throw** **new** Exception(e);

}

}

**return** handlers;

}

}

Access Filter:

**public** **class** AccessFilter **implements** Filter {

**public** **void** doFilter(ServletRequest request, ServletResponse response,

FilterChain filterChain) **throws** IOException, ServletException {

System.***out***.println("filter works ... ");

HttpServletRequest servletRequest = ((HttpServletRequest) request);

HttpServletResponse servletResponse = ((HttpServletResponse) response);

HttpSession session = servletRequest.getSession(**false**);

System.***out***.println("session : " + session);

User user = **null**;

**if** (session != **null**) {

user = SessionRepository.*getSessionUser*(servletRequest);

}

String URI = servletRequest.getRequestURI().trim();

System.***out***.println(URI);

System.***out***.println(session);

**if**(URI.contains("//")) {

servletResponse.sendRedirect("404.do");

**return**;

}

System.***out***.println("before Allowing");

//Allowing image

**if** (URI.endsWith("css")

|| URI.endsWith("js")

|| URI.endsWith("png")

|| URI.endsWith("jpg")

|| URI.endsWith("jpeg")

|| URI.endsWith("ttf")

|| URI.endsWith("woff")

|| URI.endsWith("ttf")

|| URI.endsWith("woff2")

|| URI.endsWith("ico")

) {

System.***out***.println("do filter ");

filterChain.doFilter(servletRequest, servletResponse);

**return**;

}

**if** (!URI.contains(".do") && (

!URI.equals("/GuideMe/") &&

!URI.contains("chat") &&

!(URI.contains("Report"))

)) {

System.***out***.println("uri not contains");

System.***out***.println(URI);

servletResponse.sendRedirect("404.do");

**return**;

}

//Verifying does session null

**if** (session == **null** && (

!URI.contains("home.do") &&

!URI.contains("loginPage.do") &&

!URI.contains("login.do") &&

!URI.contains("loginfb.do") &&

!URI.contains("loginvk.do") &&

!URI.contains("register.do") &&

!URI.contains("registervalidator.do") &&

!URI.contains("getCitiesByCountry.do") &&

!URI.contains("getLocalCountryAnalogs.do") &&

!URI.contains("getLocalCityAnalogs.do") &&

!URI.contains("registerAddressValidator.do") &&

!URI.contains("confirmValidator.do") &&

!URI.contains("submitRegister.do") &&

!URI.contains("userforgotpassword.do") &&

!URI.contains("sendlinktoresetpass.do") &&

!URI.contains("searchindexpage.do") &&

!URI.contains("404.do") &&

!URI.contains("local.do") &&

!URI.contains("changepassword.do")&&

!URI.contains("changingpassword.do")&&

!URI.equals("/GuideMe/")

)) {

System.***out***.println("session null - redirecting in login ");

servletResponse.sendRedirect("loginPage.do");

**return**;

}

**if** (session != **null** &&

user == **null** && (

!URI.contains("home.do") &&

!URI.contains("loginPage.do") &&

!URI.contains("login.do") &&

!URI.contains("loginfb.do") &&

!URI.contains("loginvk.do") &&

!URI.contains("register.do") &&

!URI.contains("registervalidator.do") &&

!URI.contains("getCitiesByCountry.do") &&

!URI.contains("getLocalCountryAnalogs.do") &&

!URI.contains("getLocalCityAnalogs.do") &&

!URI.contains("registerAddressValidator.do") &&

!URI.contains("confirmValidator.do") &&

!URI.contains("submitRegister.do") &&

!URI.contains("userforgotpassword.do") &&

!URI.contains("sendlinktoresetpass.do") &&

!URI.contains("searchindexpage.do") &&

!URI.contains("404.do") &&

!URI.contains("local.do") &&

!URI.contains("changepassword.do")&&

!URI.contains("changingpassword.do")&&

!URI.equals("/GuideMe/")&&

!URI.contains("mapstest.do")

)) {

System.***out***.println("session user null - redirecting in login ");

servletResponse.sendRedirect("loginPage.do");

**return**;

}

//Filter for admin

**if** (session != **null**

&& user != **null**

&& (URI.contains("admin.do")

|| URI.contains("adminTag.do")

|| URI.contains("adminuserservlet.do")

|| URI.contains("admincity.do")

|| URI.contains("admincountry.do")

|| URI.contains("adminlanguage.do")

)) {

**if** (user.getUserType().getName().equals("admin")) {

filterChain.doFilter(request, response);

**return**;

} **else** {

servletResponse.sendRedirect("home.do");

**return**;

}

}

System.***out***.println("chain end ");

filterChain.doFilter(request, response);

}

@Override

**public** **void** init(FilterConfig filterConfig) **throws** ServletException {

String encodingParam = filterConfig.getInitParameter("encoding");

**if** (encodingParam != **null**) {

// encoding = encodingParam;

}

}

@Override

**public** **void** destroy() {

// **TODO** Auto-generated method stub

}

}

**Додаток Б. Концептуальна діаграма класів Туристичного веб-порталу**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | | | |
| Виконав | Гринчук Т.А. |  |  | Діаграма класів | Сторінка |
| Перевірив | Білас О.Є. | Підпис |  | 1 з 1 |
| ІПДО кафедра ПЗ ПЗС-31 | | | |

**Додаток В. Діаграма прецендентів**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | | | |
| Виконав | Гринчук Т.А. |  |  | Діаграма прецендентів | Сторінка |
| Перевірив | Білас О.Є. | Підпис |  | 1 з 1 |
| ІПДО кафедра ПЗ ПЗС-31 | | | |

**Додаток Г. Діаграма діяльності**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | | | |
| Виконав | Гринчук Т.А. |  |  | Діаграма діяльності | Сторінка |
| Перевірив | Білас О.Є. | Підпис |  | 1 з 1 |
| ІПДО кафедра ПЗ ПЗС-31 | | | |

**Додаток Д. Інструкція користувача**

1). Для того щоб мати можливість використовути весь функціонал сайту, потрібно увійти в свій обліковий запис рис Д.1.



Рис. Д.1. Авторизація в систему

2). Для того щоб знайти подію потрібно скористатись наявними фільтрами для пошуку і натиснути кнопку «GUIDE ME !» (рис. Д.2).

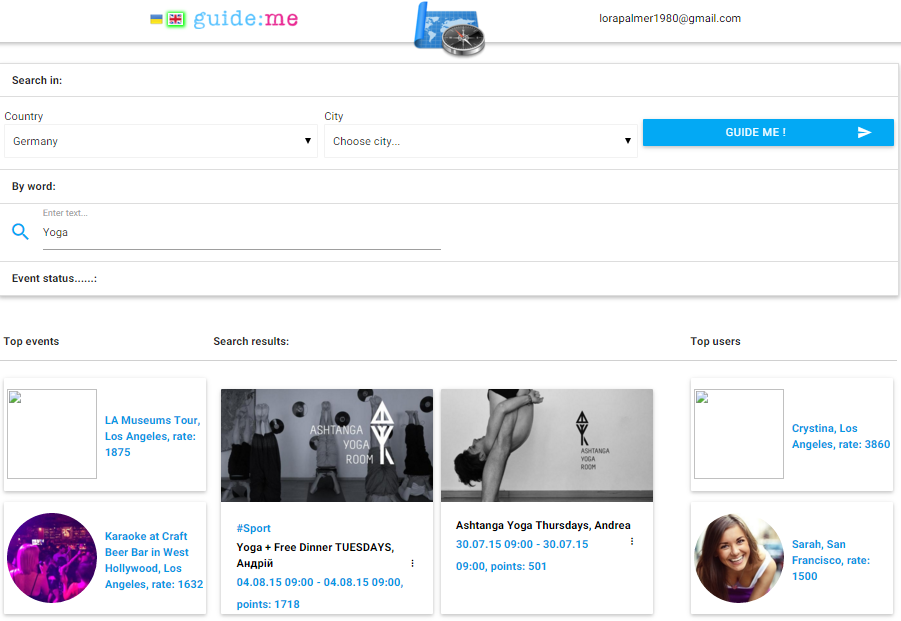


Рис. Д.2. Пошук подій

3). Вибравши, подію, яка зацікавила, слід перейти на її сторінку і при бажанні долучитись до неї, натиснувши кнопку «Join Event» (рис. Д.3).

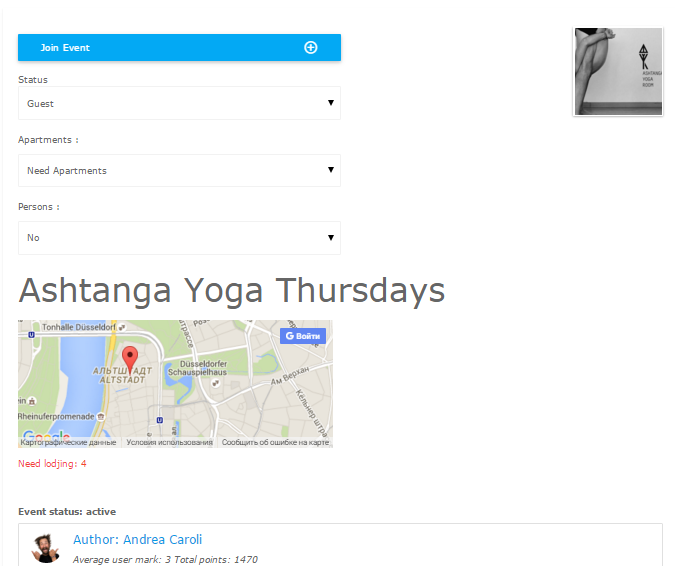


Рис. Д.3. Долучення до події

3). Модератор події може прийняти вхідні заявки на участь, натиснувши «Accept request» (рис. Д.4).

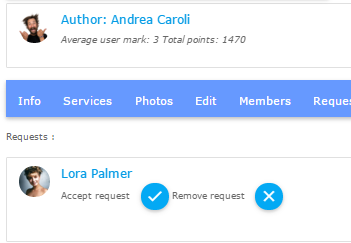


Рис. Д.4. Прийняття заявки на подію

3). На вкладці чат існує можливість спілкування з учасниками події (рис. Д.5).

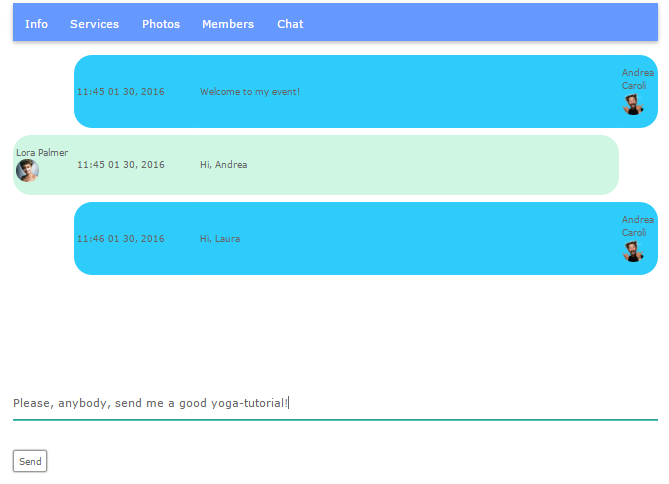


Рис. Д.5. Спілкування в чаті

3). Учасники події можуть завантажувати нові фото (рис. Д.6).



Рис. Д.6. Завантаження фото

3). Також для учасників існує можливість оцінювати подію та залишати про неї коментар (рис. Д.7).

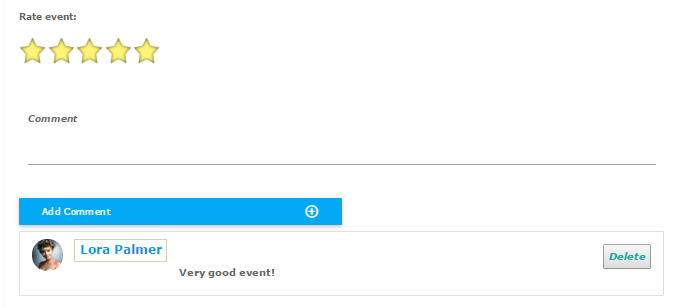


Рис. Д.7. Оцінка та коментування події

4). Модератор події здійснює розсилку почти в рамках цієї події (рис. Д.8).

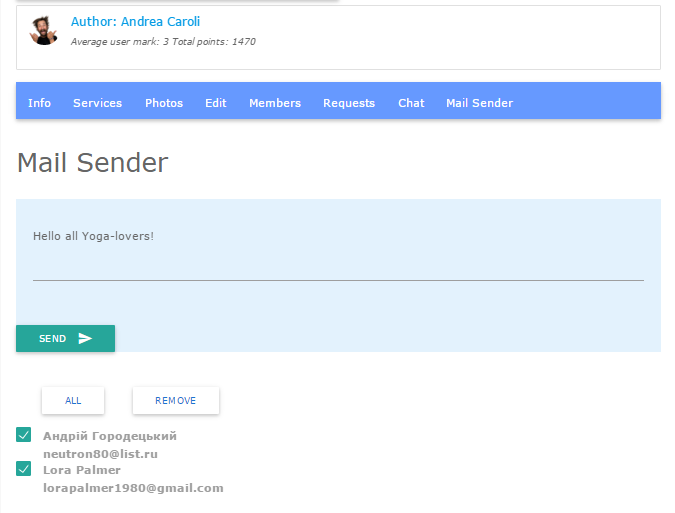


Рис. Д.8. Розсилка почти

4). Модератор підбирає для події теги, що її характеризують (рис. Д.9).

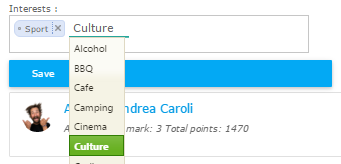


Рис. Д.9. Підбір тегів інтересів

5). Якщо подія є екскурсією, то на вкладці «Services» можна замовити додаткові послуги (рис. Д.10).

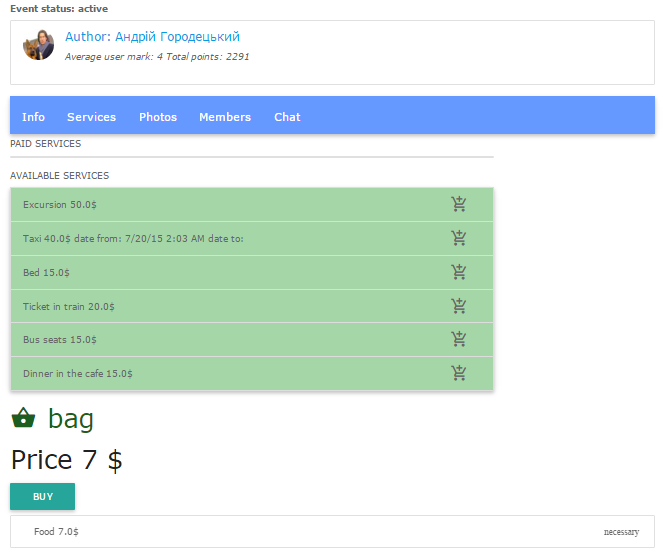


Рис. Д.10. Замовлення додаткових послуг

**Додаток Е. Результати функціонального тестування**

Проведемо функціональне тестування згідно наступних тестових випадків:

## ID № 1 : Add Friend:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен на профілі користувача, який ще не є другом додати його в друзі кнопкою «Add Friend» а потім виконати зворотню дію | | |
| Steps | Expected Results | Results |
| 1.Натиснути в профілі у «Add Friend» | Навпроти кнопки з’явився надпис  «Cancel your request» (рис. Е.1) | Passed |
| 2. Перезавантажити сторінку | Надпис не зміниться | Passed |
| 3. Натиснути на «Cancel your request» | Навпроти кнопки з’явився надпис  «Add Friend» | Passed |
| 4. Перезавантажити сторінку | Надпис не зміниться | Passed |

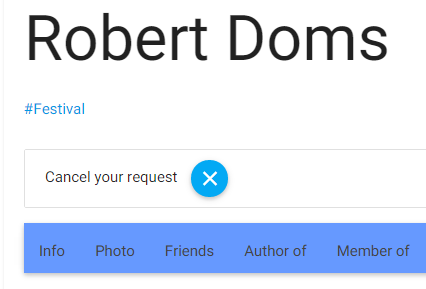


Рис. Е.1. Кнопка «Cancel your request»

## ID № 2 : Add Friend:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен на профілі користувача, який є другом вилучити його з друзів кнопкою « Remove friend» а потім виконати зворотню дію | | |
| Steps | Expected Results | Results |
| 1.Натиснути в профілі у « Remove friend » | Навпроти кнопки з’явився надпис  Accept request та Discard request (рис. Е.2) | Passed |
| 2. Перезавантажити сторінку | Надпис не зміниться | Passed |
| 3. Натиснути на « Accept request» | Навпроти кнопки з’явився надпис  «Remove friend» | **Failed** |
| 4. Перезавантажити сторінку | Надпис не зміниться | Passed |

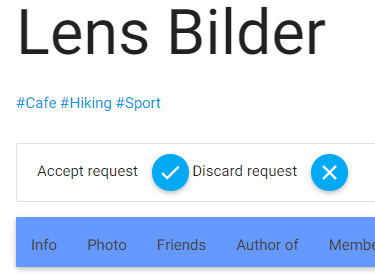


Рис. Е.2. Кнопки Accept request та Discard request

## ID № 3 : Find Friend:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен в кабінеті користувача здійснити пошук друзів | | |
| Steps | Expected Results | Results |
| 1.Натиснути в кабынеті у «Find friends» | З’явиться форма пошуку друзів | Passed |
| 2. Ввести в полі для пошуку: «bill» | З’явиться список друзів в яких в прізвищі чи імені є дана підстрока (рис. Е.3) | Passed |
| 3. Видалити строку | З’явиться початкова форма пошуку друзів | Passed |

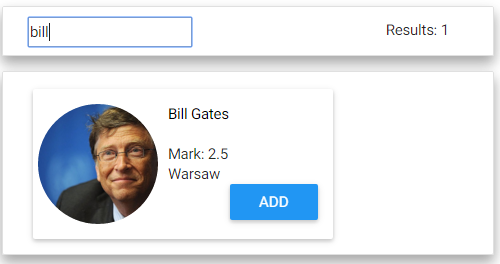


Рис. Е.3. Результат пошуку друзів

## ID № 4 : Find Friend:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен в кабінеті користувача здійснити пошук друзів | | |
| Steps | Expected Results | Results |
| 1.Натиснути в кабінеті у «Find friends» | З’явиться форма пошуку друзів | Passed |
| 2. Ввести в полі для пошуку: « User or Guide» варіант: Guides | З’явиться список користувачів, які є гідами | Passed |
| 3. Ввести в полі для пошуку: « User or Guide» варіант: User | З’явиться список користувачів, які є звичайними користувачами | Passed |

## ID № 5 : Find Friend:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен в кабінеті користувача здійснити пошук друзів | | |
| Steps | Expected Results | Results |
| 1.Натиснути в кабінеті у «Find friends» | З’явиться форма пошуку друзів | Passed |
| 2. Ввести в полі для пошуку: « User or Guide» варіант: Guides | З’явиться список користувачів, які є гідами | Passed |
| 3. Ввести в полі для пошуку: « User or Guide» варіант: User | З’явиться список користувачів, які є звичайними користувачами | Passed |

## ID № 6 : Find Friend:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен в кабінеті користувача здійснити пошук друзів | | |
| Steps | Expected Results | Results |
| 1.Натиснути в кабінеті у «Find friends» | З’явиться форма пошуку друзів | Passed |
| 2. Ввести в полі для пошуку: « Choose country» варіант: Germany | З’явиться список користувачів, які є живуть в Німеччині | Passed |
| 4. Ввести в полі для пошуку: « Choose country» варіант: All | З’явиться початковий список користувачів | Passed |

## ID № 7 : Find Friend:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен в кабінеті користувача здійснити пошук друзів | | |
| Steps | Expected Results | Results |
| 1.Натиснути в кабінеті у «Find friends» | З’явиться форма пошуку друзів | Passed |
| 2. Ввести в полі для пошуку: « Choose country» варіант: Ukraine | З’явиться список користувачів, які є живуть в Україні, в списку міст будуть доступні лише міста України | Passed |
| 4. Ввести в полі для пошуку: « Choose city» варіант: Lviv | З’явиться список користувачів, які є живуть у Львові | Passed |

## ID № 8 : Find Friend:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен в кабінеті користувача здійснити пошук друзів | | |
| Steps | Expected Results | Results |
| 1.Натиснути в кабінеті у «Find friends» | З’явиться форма пошуку друзів | Passed |
| 2. Ввести в полі для пошуку за тегами варіант: Camping | З’явиться список користувачів, які мають інтерес Camping | Passed |
| 4. Видалити цей тег | З’явиться початковий список користувачів | Passed |

## ID № 9 : Join Event:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен на профілі події долучитися до неї, а потім виконати зворотню дію | | |
| Steps | Expected Results | Results |
| 1.Натиснути в профілі у «Join Event» | Навпроти кнопки з’явився надпис  «Cancel request» | Passed |
| 2. Перезавантажити сторінку | Надпис не зміниться | Passed |
| 3. Натиснути на «Cancel your request» | Навпроти кнопки з’явився надпис  «Join Event» | Passed |
| 4. Перезавантажити сторінку | Надпис не зміниться | Passed |

## ID № 10 : Find Events:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен в кабінеті користувача здійснити пошук друзів | | |
| Steps | Expected Results | Results |
| 1.Натиснути в кабінеті у «Find Events» | З’явиться форма пошуку подій | Passed |
| 2. Ввести в полі для пошуку: « опер» | З’явиться список подій в описі яких є дана підстрока | Passed |
| 3. Видалити строку | З’явиться початкова форма пошуку подій | Passed |

## ID № 11 : Find Events:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен в кабінеті користувача здійснити пошук подій | | |
| Steps | Expected Results | Results |
| 1.Натиснути в кабінеті у «Find events» | З’явиться форма пошуку подій | Passed |
| 2. Ввести в полі для пошуку: « Choose country» варіант: Germany | З’явиться список подій, які є в Німеччині | Passed |
| 4. Ввести в полі для пошуку: « Choose country» варіант: All | З’явиться початковий список подій | Passed |

## ID № 12 : Find Events:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен в кабінеті користувача здійснити пошук подій | | |
| Steps | Expected Results | Results |
| 1.Натиснути в кабінеті у «Find events» | З’явиться форма пошуку подій | Passed |
| 2. Ввести в полі для пошуку: « Choose country» варіант: Ukraine | З’явиться список подій, які відбуваються в Україні, в списку міст будуть доступні лише міста України | Passed |
| 4. Ввести в полі для пошуку: « Choose city» варіант: Lviv | З’явиться список подій, які є у Львові | Passed |

## ID № 13 : Find Events:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен в кабінеті користувача здійснити пошук подій | | |
| Steps | Expected Results | Results |
| 1.Натиснути в кабінеті у «Find events» | З’явиться форма пошуку подій | Passed |
| 2. Натиснути на тег: Camping | З’явиться список подій, які мають тег Camping | Passed |

## ID № 14 : Messenger:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен в кабінеті користувача здійснити переписку з другом | | |
| Steps | Expected Results | Results |
| 1.Натиснути в кабінеті у «Message» | З’явиться форма переписки | Passed |
| 2. Ввести в полі повідомлення і відправити його | Повідомлення з’явиться в списку відправлених повідомлень | Passed |

## ID № 15 : Chat:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен в профілі події здійснити переписку з її учасниками | | |
| Steps | Expected Results | Results |
| 1.Перейти в профілі події у «Chat» | З’явиться форма переписки | Passed |
| 2. Ввести в полі повідомлення і відправити його | Повідомлення з’явиться в списку відправлених повідомлень | Passed |

**Додаток Є. Результати тестування на безпеку**

Проведемо тестування на безпеку згідно наступних тестових випадків:

## ID № 1 : Not Authorized:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен під не заавтроризованим користувачем спробувати зайти на сторінки відмінні від початкової | | |
| Steps | Expected Results | Results |
| 1.В адресній строці вводимо: http://localhost:8080/GuideMe/loginPage.do | Повинна зявитись сторінка авторизації (рис. Є.1) | Passed |
| 2. В адресній строці вводимо: http://localhost:8080/GuideMe/eventDetail.do?id=8 | Повинна зявитись сторінка авторизації (рис. Є.1) | Passed |
| 3. В адресній строці вводимо: http://localhost:8080/GuideMe/admin.do | Повинна зявитись початкова сторінка | Passed |
| 4. В адресній строці вводимо: http://localhost:8080/GuideMe/usermessages.do | Повинна зявитись сторінка авторизації (рис. Є.1) | Passed |

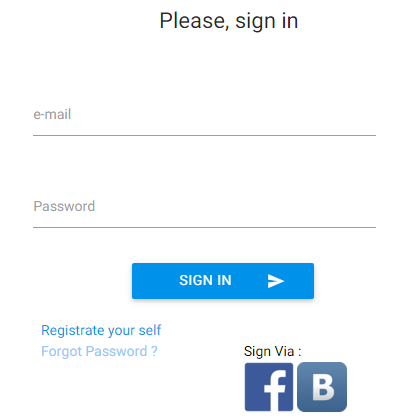


Рис. Є.1. Сторінка авторизації

## ID № 2 : Not Authorized:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен під звичайним користувачем зайти в адмін панель | | |
| Steps | Expected Results | Results |
| 3. В адресній строці вводимо: http://localhost:8080/GuideMe/admin.do | Повинна зявитись початкова сторінка | Passed |

## ID № 3 : Admin resources:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен під звичайним користувачем зайти в адмін панель | | |
| Steps | Expected Results | Results |
| 3. В адресній строці вводимо: http://localhost:8080/GuideMe/admin.do | Повинна зявитись початкова сторінка | Passed |

## ID № 4 : Admin resources:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен під роллю Гід зайти в адмін панель | | |
| Steps | Expected Results | Results |
| 3. В адресній строці вводимо: http://localhost:8080/GuideMe/admin.do | Повинна з’явитись початкова сторінка | Passed |

## ID № 5 : HTML injection:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен спробувати ввести тег html при створенні нової події | | |
| Steps | Expected Results | Results |
| 3. На сторінці події в полі Name вводимо тег <h1> | Повинна з’явитись повідомлення про заборонені символи (рис. Є.2) | Passed |

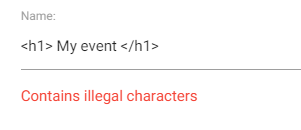


Рис. Є.2. Повідомлення про заборонені символи

## ID № 6 : SQL injection:

|  |  |  |
| --- | --- | --- |
| Sunmmary: Тестер повинен спробувати ввести sql запит в полі пошуку події | | |
| Steps | Expected Results | Results |
| 3. На сторінці події в полі пошуку введемо: drop table users; | Повинна з’явитись повідомлення що такої події нема. Скрипт не має виконатись (рис. Є.3). | Passed |

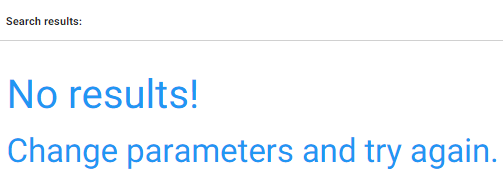


Рис. Є.3. Ігнорування SQL injection