НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ННК "Інститут прикладного системного аналізу"
(повна назва інституту/факультету)

Кафедра Системного проектування
(повна назва кафедри)

«До захисту допущено»

Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)

"___"_____2015 р.

Дипломна робота

першого (бакалаврського) **рівня вищої освіти**
(першого (бакалаврського), другого (магістерського))

зі спеціальності 7.050102, 8.050102 Інформаційні технології проектування
7.050103, 8.050103 Системне проектування
(код та назва спеціальності)

на тему: Розробка порталу для якісного перекладу текстів

Виконав : студент 4 курсу, групи ДА-12
(шифр групи)

Романчук Володимир Ігорович _____ _____
(прізвище, ім'я, по батькові) (підпис)

Керівник доцент, к.т.н. Корначевський Я.І. _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант Java EE доцент, к.т.н. Смаковський Д.С. _____
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент професор, д.т.н. професор Бідюк П.І. _____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Нормоконтроль ст. викладач Британ О.А. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі
немає запозичень з праць інших авторів
без відповідних посилань.
Студент _____
(підпис)

Київ – 2015 року

Національний технічний університет України
«Київський політехнічний інститут»

Факультет (інститут)_____ ННК “ Інститут прикладного системного аналізу”
(повна назва)

Кафедра_____Системного проектування_____
(повна назва)

Рівень вищої освіти _____Перший (Бакалаврський)_____
(перший(бакалаврський), другий (магістерський) або спеціаліста)

Спеціальність 7.050102, 8.050102 Інформаційні технології проектування
_____7.050103, 8.050103 Системне проектування_____.
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)
«___»_____2015 р.

ЗАВДАННЯ
на дипломну роботу студенту
_____Романчуку Володимиру Ігоровичу_____
(прізвище, ім’я, по батькові)

1. Тема проекту (роботи) Розробка порталу для якісного перекладу текстів

керівник проекту (роботи) Корначевський Ярослав Ілліч, к.т.н.
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «02» квітня 2015 р. № 30/1-ст

2. Строк подання студентом проекту (роботи) 19.06.2015

3. Вихідні дані до проекту (роботи)

Портал, який забезпечуватиме можливість реєстрації на ньому перекладачів, зі специфікацією мови та галузі перекладу, а також з можливістю завантаження документів для перекладу, і можливістю електронної оплати.

4. Зміст розрахунково-пояснювальної записки (перелік завдань, які потрібно розробити)

1. Дослідити засоби розробки веб-порталів.
2. Провести порівняльний аналіз мов веб-програмування.

3. Обрати базу даних для обраного завдання.
4. Окреслити структуру порталу.
5. Розробити комерційний веб-проект.
6. Провести медіа маркетинг порталу.
7. Описати ключові структурні елементи обраної мови.
8. Продемонструвати інтерфейс користувача.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

1. Landing page version 1.0 – плакат.
2. Project Translation version 1.0 – плакат.
3. MVP– плакат.
4. Structure part – плакат.

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Гусєв А.М. , доцент		
Основна частина	Корначевський Я.І., доцент		
Java EE	Смаковський Д.С., доцент		

7. Дата видачі завдання 01.02.2015

Календарний план

№ з/п	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Отримання завдання	01.02.2015	
2	Збір інформації	15.02.2015	
3	Вивчення варіантів реалізації та вибір варіанту для розробки	28.02.2015	
4	Розробка алгоритму та структури веб-порталу	10.03.2015	
5	Розробка плану тестування	15.03.2015	
6	Розробка програмної моделі	25.03.2015	
7	Розробка опису та інтерфейсу	25.04.2015	
8	Тестування моделі	30.04.2015	
9	Оформлення дипломної роботи	31.05.2015	
10	Отримання допуску до захисту та подача роботи в ДЕК	19.06.2015	

Студент _____ В.І. Романчук
(підпис) (ініціали, прізвище)

Керівник роботи _____ Я.І. Корначевський
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

до бакалаврської дипломної роботи Романчука Володимира Ігоровича на тему:
«Розробка порталу для якісного перекладу текстів»

Дипломна робота присвячена розробці порталу для виконання якісного перекладу, так як проблема стоїть досить давно, потрібно враховувати структуру, стилі та прикладне застосування документів, що супроводжують виробництво та з'являються при укладанні юридичних угод.

Однак в Україні всі компанії, які можуть надати такий переклад працюють по-старому. Часові діапазони виконання таких завдань коливаються до 3 робочих днів. Знайти їх викликає труднощі. Ще більші проблеми виникають у якості, яку вони надають.

Розроблений портал пропонує вирішення проблеми швидкого та якісного перекладу для підприємств, забезпечуючи прозору роботу з різними форматами файлів користувачів, а також зручну оплату.

Загальний об'єм роботи 93 сторінки, 29 рисунків, 13 таблиці, 8 бібліографічних найменувань.

Ключові слова: веб-портал, розробка, переклад, MVP, якість, швидкість, оплата, угоди, бази даних, мови програмування, порівняльна характеристика.

АННОТАЦИЯ

к бакалаврской дипломной роботе Романчука Владимира Игоревича на тему:
«Разработка портала для качественного перевода текстов»

Работа посвящена разработке портала для выполнения качественного перевода, поскольку проблема эта стоит достаточно давно, так как нужно учесть структуру, стили и прикладное применение документов, которые сопровождаются в производстве и появляются при составлении юридических соглашений.

Однако в Украине все компании, которые могут предоставить такой перевод, работают по-старому. Временные диапазоны выполнения таких задачи колеблются до 3 рабочих дней. Найти их вызывает трудности. Еще большие проблемы возникают в качестве, которое они предоставляют. Разработанный портал предлагает решение проблемы быстрого и качественного перевода для предприятий, обеспечивая прозрачную работу с различными форматами файлов пользователей, а также удобную оплату.

Общий объем работы 93 страницы, 29 рисунков, 13 таблицы, 8 библиографических наименований.

Ключевые слова: веб-портал, разработка, перевод, MVP, качество, скорость, оплата, договора, базы данных, языки программирования, сравнительная характеристика.

ANNOTATION

for bachelor diploma level of Romanchuk Volodymyr Igorovuch on
"Designing portal for high quality text translation"

The work is devoted to the development of a portal to perform high-quality translation, because high quality translation is quite a problem: you have to take into account the structure, style and practical application of documents, that are used in the development, in some other legal transactions.

But in Ukraine, all companies that provides translation work as in old times. Translation takes up to 3 working days, and even such companies are difficult to find. Even greater problems arise when it comes to the quality. Designed portal offers a solution to the problem of fast and adequate translations for companies, providing transparent operation with different file formats and convenient payment means as well.

Total amount of work 93 pages, 29 images, 13 tables, 8 bibliographic reference

Key words: web-portal, development, translation, MVP, quality, speed, payment, agreement, databases, language of programming, comparable characteristics.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ

Термін, скорочення	Значення
Клієнти	замовники письмового перекладу одного із 4-ох тематик (технічна, економічна, юридична та економічна). Замовниками виступають співробітники юридичних, транспортних, торгівельних та ін. компаній.
Виконавці	бюро-перекладів, які мають вагомий досвід роботи та провірену репутацію, виконують письмовий переклад із даних тематик.
Компанія QTExS	виступає як суб’єкт прийому коштів від клієнтів, перерозподілу коштів відповідним виконавцям та стягнення комісії за користування ресурсом. Виступає як контакт – центр для координації та консультації клієнтів та виконавців.
QTExS	Quality text experience
IT (англ. Informationtechnology)	нещодавно створена компанія (можливо, ще не зареєстрована офіційно, але серйозно планує стати офіційною), що будує свій бізнес на основі інновацій або інноваційних технологій,

Landing page	Інформаційна сторінка для виявлення зацікавленості клієнтів та представленням перед публікою

ВСТУП

Проблема якісного перекладу стоїть досить давно, оскільки потрібно врахувати структуру, стилі та прикладне застосування документів, що супроводжуються у виробництві та укладанням юридичних угод. Якщо порівнювати google translator по шкалі перекладу від 0 до 10. Цей перекладач буде на рівні з 0, а професійний переклад починається з відмітки 7+. Саме на такий переклад орієнтується команда Qtexs.

Однак з в Україні всі компанії, які можуть надати такий переклад працюють по-старому. Часові діапазони виконання таких завдання коливаються до 3 робочих днів. Знайти їх викликає труднощі, а тим паче пересвідчитись у якості, яку вони надають.

Існують портали на подобі Qtexs, такі як rev.com, onehourtranslation.com, gengo.com. Проте, вони мають низку проблем, які вимагають дослідницького вирішення.

До прикладу візьмемо, onehourtranslation. Цей портал є лідером ринку перекладу у цілому світі. Якщо надавати переклад з української, то він можливий тільки на Англійську та Російську мови, що явно не забезпечить потреби перекладів українців.

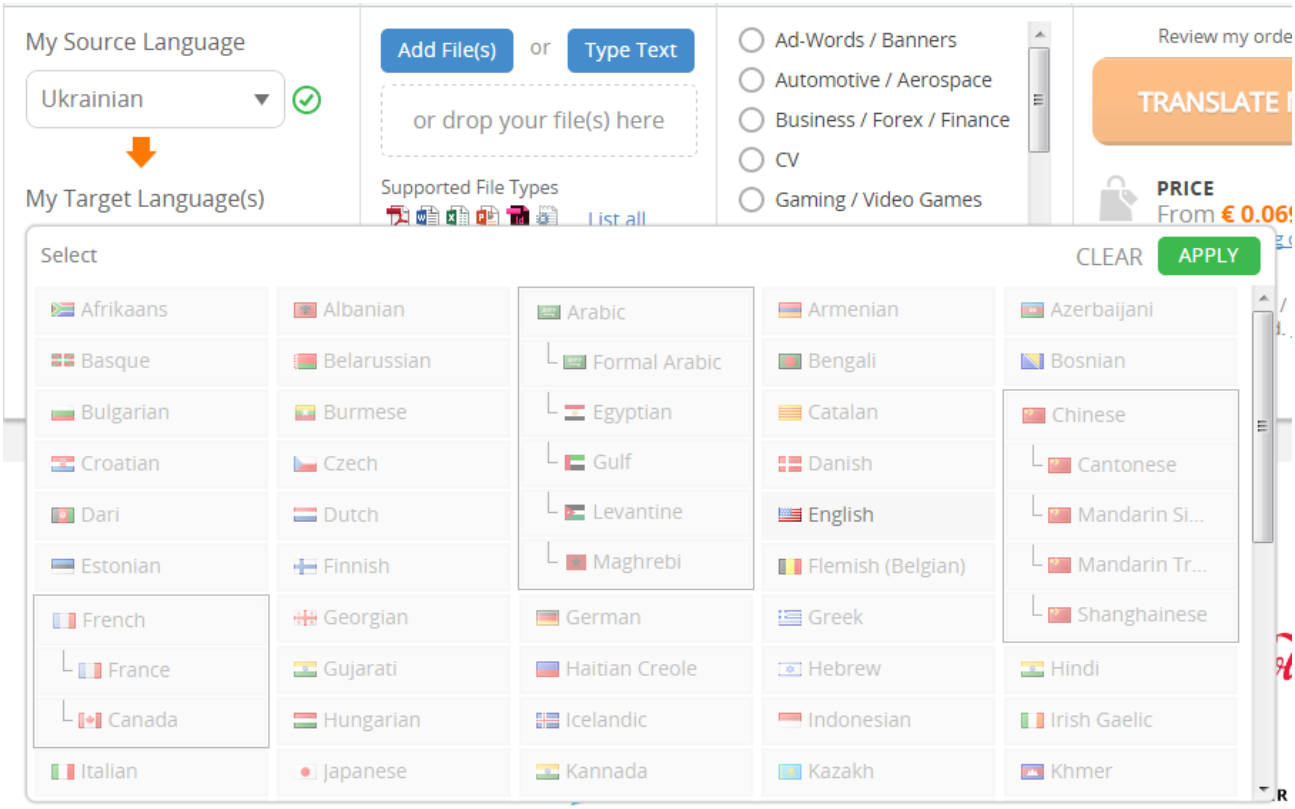

Рисунок 1 - Список наданих перекладів

Візьмемо процес замовлення як процес. Для звичайного користувача він може бути не зрозумілим та занадто накидам. Розглянемо завантаження самого файлу:

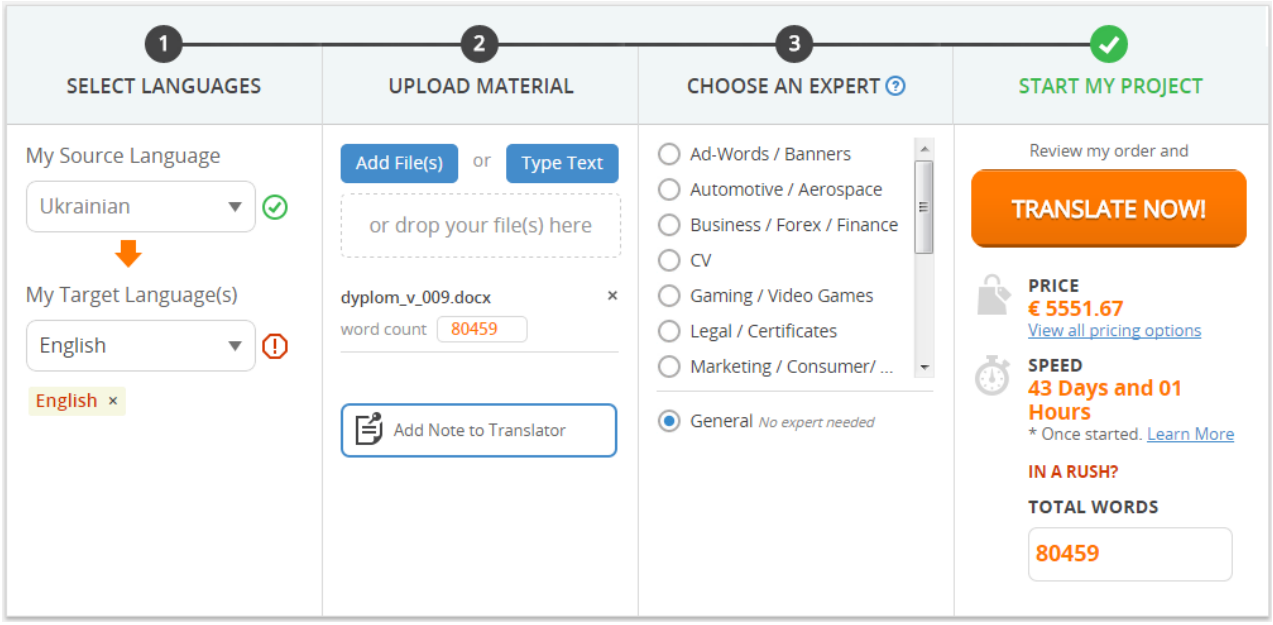

Рисунок 2 - Завантаження файлу

Автоматична система даного порталу обчислює кількість слів 80 459 . Перевіримо у Microsoft Word статисти стику цього ж фалу:

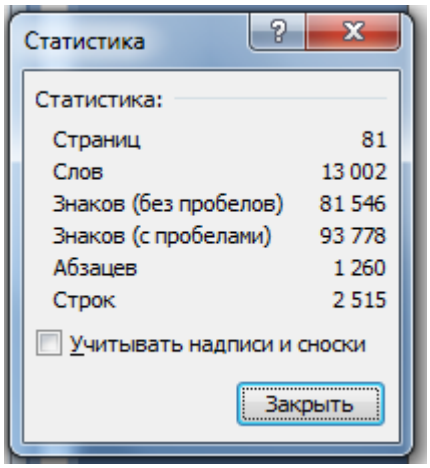

Рисунок 3 - Статистика Microsoft Word

Кількість слів 13 002, це явно не 80 459. Ціна 5551.67 € явно не є дозволеною пересічному бізнесмену.

Стоїть проблема уніфікації результатів для формування єдиного цінового діапазону перекладацьких компаній.

З політ. розвитком України та економічних взаємовідносин значно зросте потреба у якісному перекладі, де норми і правила написання тих чи інших документів будуть встановлюватись людиною. Тому у таких послугах завжди буде потреба. Як на мене, то машинний переклад не здатний повністю витіснити такі послуги. За даними джерелами ринок послуг перекладу є досить стійким до світових рецесій. Відповідно до даних Common Sense Advisory, розмір світового ринку послуг перекладу оцінюють у $33,5 млрд. у 2012 році. Згідно доповіді IbisWorld, даний сегмент ринку очікує приріст аж до позначки $37 млрд. у 2018 році. США є найбільшим ринком за ним слідує Європа, а ринок Азії є найбільш динамічно зростаючим.

Відповідно до Статистичного Бюро США, індустрію перекладу очікує ріст на 42% у часовому проміжку 2010-2020рр. Очікується розвиток перекладу даних тематик:

1. Технічний переклад, через зростаючий попит перекладу документації автомобільної промисловості та ринку електроніки.

2. Медичний переклад, через наступаючий вплив дрібних/середніх компаній на глобальний ринок. Зокрема через зростаючий вплив фармацевтичного ринку Азії, темп приросту – 14%. [1]

3. Економічний переклад, зокрема через зростаючий вплив банківського та фінансового сектору згідно із даними LT –innovate.com. Ринок програмного забезпечення перекладу у США оцінюється у $575,5 млн. у 2010 році та очікується ріст до $3 млрд. у 2017 році. [2]

Згідно із даними Translation Bureau Benchmarking and Comparative Analysis, близько 80% світового перекладу обслуговується за ціною менше $0,15 за 1 слово [3]. Для порівняння в Україні середня ціна за 1 слово перекладу = $0,018 (навіть при курсі USD/UAH = 8,14) [4].

Найбільший попит щодо високоякісного перекладу спостерігається щодо перекладу юридичної документації, як і потреби у швидкості даної послуги

У світі налічується близько 26 тис. бюро-перекладів [5].

Згідно даних СПАРК-Интерфакс, в Україні було зареєстровано 604 юридичних одиниці, що надають послуги письмового перекладу(2012рік) [6]. Об’єм ринку послуг перекладу в Україні - $80-100 млн., Росії - $627 — $1000 млн. В Україні цей ринок на даний момент є досить закритим, згідно інформації UTIC[4]. Оцінка діяльності прямого конкурента “gengo.com” (2012рік) [7]: Кількість перекладених слів: 40+ млн. перекладених слів (при перекладі $0.15 за слово, об’єм перекладу = $ 6+ млн.) 7,500 перекладачів.

Дана платформа пропонує вирішення проблеми швидкого та якісного перекладу для підприємств..

Прийом файлів від користувачів (.txt,.doc, .docx, .pdf, .ppt, .pptx, .odt) та графічні файли (.png, .gpeg та ін.)

Відбувається зчитування файлів на предмет кількості символів та сторінок (1800 символів =1 сторінка), графічні файли не зчитуються, клієнту надається опція ввести самостійно кількість сторінок для перекладу (пр. «1», «3» та ін.), де вказується: «Будь ласка, введіть кількість сторінок для перекладу. 1 сторінка = 1800 символів і менше.»

Прийом оплати через систему liqpay для зручнішого здійснення платежів.

Надання якісного перекладу згідно із стандартною схемою (5 ст. перекладу за 24 год.) та можливістю пришвидшити виконання замовлення (15 ст. перекладу за 24 год. – збільшення ціни замовлення на 100%) через широку мережу доступу бюро-перекладу до даної платформи. Якість в обох випадках залишається стабільно високою.

Повернення коштів у разі неналежного виконання замовлення. Отримання замовлення – оцінка виконавця клієнтом – у разі невдоволення клієнту надаються контакти для зворотного зв’язку, якщо виконавець неякісно виконав замовлення чи не у встановлений час, кошти повернуться клієнту у повному обсязі через банківський переказ.

1. ЗАСОБИ РОЗРОБКИ WEB-ПОРТАЛІВ

1.1 Розробка у Web-середовищі

Web-середовище – це величезна бібліотека людських надбань, це неабиякий архів корисної та оригінальної інформації. І ця інформація з кожним роком поповнюється новими знаннями.

Сучасна людина живе у динамічному, непередбачуваному світі. Все, що оточує сучасну людину швидко змінюється. Web-середовище – це середовище, яке відображає сучасне життя. Здається, що нові розробки у цій галузі з'являються мало не щодня.

Що ж собою являє web-середовище? Якщо казати коротко, то це набір web-сторінок, наповнених інформацією. Людині для того, щоб отримати певну інформацію, необхідно набрати у браузері адрес сторінки, яка її цікавить. У випадках коли потрібно знайти інформацію, а точний адрес необхідної сторінки невідомо, то тоді за допомогою звертаються до пошукових машин, створюють запит і у відповідь отримують набір web-сторінок, які можуть містити дані, що цікавлять людину. Web-сторінки також називають html-документами. Для того щоб створити html-документ, потрібно у текстовому редакторі за допомогою html-тегів розмітити документ. Для цього використовують мову гіпертекстової розмітки HTML. Існує консорціум Всесвітньої павутини (W3C – World Wide Web Consortium), створений у 1994 році Тімом Бернерс-Лі, який займається розробкою вільно розповсюджуваних технологій для Всесвітньої павутини. Також W3C встановлює стандарти на web-технології. W3C вдосконалює і стандартизує HTML.

Коли web-сторінку створено, потрібно дати їй ім'я. Основна сторінка повинна бути названа index, тому що по замовчуванню web-сервер повертає сторінку index. Розширенням для веб-сторінки може бути html, php, htm, shtml, jsp в залежності від змісту. W3C рекомендує на web-сторінках розмежовувати вид та розмітку. HTML використовується для розмітки, для створення зовнішнього вигляду використовують CSS.

Web-сторінка зберігається на сервері. Коли клієнт створює запит web-сторінки, то сервер відправляє відповідні дані на комп'ютер клієнта, де ці дані оброблюються і відображуються за допомогою браузера. Існують технології, за допомогою яких можна динамічно змінювати вигляд web-сторінки. Сьогоднішні web-сторінки – це не статичні документи, як це було раніше, а динамічні, як і наш світ.

Для динамічного наповнення web-сторінки використовуються технології, які можуть бути виконані як на стороні клієнта, так і на стороні сервера. На стороні клієнта зазвичай використовують JavaScript. Ця мова підтримується всіма виробниками, які створюють браузери – це перевага JavaScript, а недолік – це відсутність єдиного стандарту. Як правило, для того щоб створити скрипт, який працював би на всіх браузерах, необхідно спочатку перевірити браузер, за допомогою якого клієнт дивиться web-сторінку, а потім використовувати специфічні функції для цього браузера. Існують інші технології, які можна використовувати на стороні клієнта, але вони не отримали широкого розповсюдження, оскільки є окремим рішеннями для того чи іншого браузера (VBScript) або потребують додаткових програм для свого виконання (Java-аплети, Macromedia Flash).

Для динамічного наповнення web-сторінки на стороні сервера, на сьогоднішній день, можна обрати з більшої кількості технологій. Тут перед розробником постає питання, яку з технологій ліпше використовувати. Адже потрібно, щоб з одного боку поставлена задача була вирішена в якомога найкоротший термін, а з іншого щоб рішення задачі було правильним та надійним. Розробник може обрати технології Python, Ruby, PHP, Perl, JSP, ASP, ASP.NET, Cold Fusion, CGI, Java-сервлети, тощо. Крім добровільного вибору технології, розробник повинен ще виходити з того, якими засобами володіє комп'ютер, на якому буде розміщена його web-сторінка.

Так, спроби надати функціональні можливості через інтернет зв'язані з масою перешкод. Але ці перешкоди долаються тими перевагами, які надають web-додатки в порівнянні з звичайними додатками:

- встановлення web-додатків простіше та дешевше;
- оновлення web-додатків простіше та дешевше;
- web-додатки висувають більш гнучкі вимоги до кінцевого користувача;
- web-додатки полегшують організацію централізованого зберігання даних.

Крім вище згаданих технологій, існують ще інше технології, які полегшують життя web-розробнику. Всі ці технології переплітаються між собою та створюють web-середовище, яке надає нам ті можливості, про які раніше ми лише мріяли. І, більш за все, це лише початок того, що може бути зроблено у web-середовищі, оскільки можна вважати, що web-середовище – це поки що доросла дитина.

На сьогодні web-сторінку можна повноцінно вважати представником компанії, організації в інтернеті. І вже сьогодні кожна web-сторінка, що створюється, повинна містити такий невід'ємний атрибут, як система адміністрування. Це в свою чергу робить сторінку не просто рекламно-інформаційним ресурсом, а й достатньо динамічним джерелом інформації. По своїй суті вона стає ще одним інструментом для відображення певної динамічної інформації про компанію. В такому випадку web-сторінку можна розглядати як набір певної бізнес-логіки, що забезпечує обробку і представлення певної інформації.

Тобто web-сторінка стає достатньо вагомим інструментом комерції компанії. Тому перед створенням web-сторінки варто чітко визначитись щодо її мети, функціонального наповнення, інформаційного наповнення, зовнішнього вигляду, планом розкрутки та підтримки.

Основні фактори, від яких залежить успіх web-сторінки:

- чітко сформована стратегія;
- легка для запам'ятовування назва і адреса;
- добре продумана концепція кольорів і логотипу компанії;

- візуальне оформлення, професійно виконане з максимальною відповідністю до попередніх пунктів;
- продумана навігація по розділах (зручність в користуванні, інтуїтивність інтерфейсу);
- цікаве і динамічне інформаційне наповнення;
- звичайно персонал, що займається підтримкою web-сторінки;
- правильно виконана розкрутка.

Етапи розробки і експлуатації сайту:

- Розробити концепції. Неправильно чи неповно сформована концепція як правило призводить до негативного результату.
- Створення технічного завдання (ТЗ), і дизайну web-сторінки. Наперед описаний функціонал дозволить уникнути багатьох проблем при самій розробці. Як правило під час створення ТЗ чітко описуються всі функціональні частини проекту і алгоритми їх роботи. Чим більш деталізоване ТЗ, тим менша ймовірність виникнення негараздів у функціонуванні web-сайту. Щодо дизайну, то він повинен бути максимально професійним і близьким до концепції. Також при розробці дизайну варто врахувати сферу діяльності компанії, і тип інформаційного наповнення. На етапі розробки важливо правильно розділити всі роботи на етапи керуючись функціональними сутностями, описаними в ТЗ.
- Після завершення розробки web-сторінки, необхідно створити тестове середовище і провести детальне тестування на основі ТЗ.
- Одночасно в цей час треба підготувати адресу і площадку для розміщення сайту.
- Після тестування і виправлення всіх дефектів, що були виявлені, проект викладається в інтернет і наповнюється.
- Просування web-сторінки – реклама, популяризація проекту в інтернеті. Цей етап варто починати лише після завершення наповнення інформацією.

- Підтримка – один з найважливіших моментів. При поганій підтримці навіть найкраще виконаний сайт може зазнати невдачі. Інформація на сайті повинна оновлюватися динамічно і регулярно.

Можливо, на перший погляд всі ці кроки здаються занадто складними і непотрібними. Але на практиці тільки перші кроки в створенні web-сторінки можуть бути складними, а вже далі все йде, як правило, без особливих проблем. І навпаки, якщо чітко не розпланувати, чи підійти до створення web-сторінки не професійно – можна бути впевненим, що всі проблеми ще попереду.

Основними етапами при створенні веб-порталу є:

- мова програмування;
- база даних;
- структура порталу.

1.2 Мови веб-програмування

Мови веб-програмування - це мови, які в основному призначені для роботи з інтернет-технологіями, а окремі з них створювалися лише задля роботи з будь-яким ресурсом, і тривалий час до них приходила популярність і загальне визнання (наприклад, PHP).

Мови веб-програмування діляться на дві групи: клієнтські та серверні.

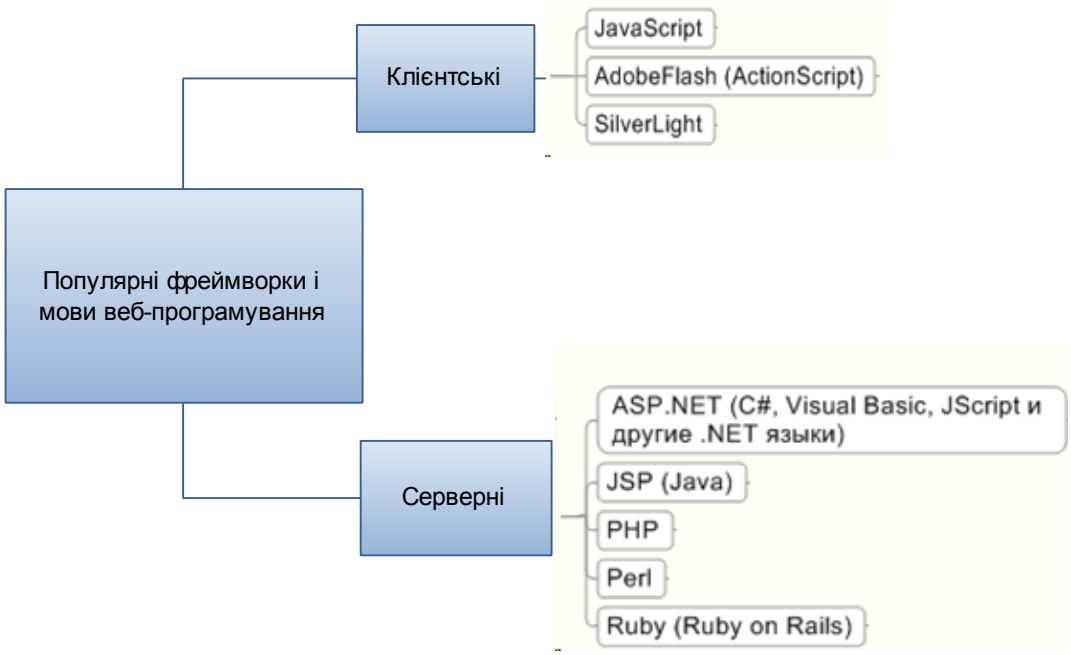

Рисунок 4- Групи мов веб-програмування

Клієнтські мови обробляються на стороні клієнта-користувача, а якщо простіше - програми на клієнтській мові обробляться браузером. Звідси випливає і недолік - це те, що обробка скрипту залежить від браузера користувача, і користувач має повноваження налаштувати свій браузер так, щоб він взагалі ігнорував написані скрипти. При цьому, якщо браузер старий, він може не підтримувати ту чи іншу мову або версію мови, на яку ви спирайтеся. З сучасними браузерами таких проблем виникати не повинно, до того ж мови програмування не так вже й часто кардинально оновлюються (раз на кілька років), і кращі з них давно відомі.

Перевага ж клієнтської мови полягає в тому, що обробка скриптів може виконуватися без відправлення документа на сервер. Це легше пояснити на прикладі: припустимо, вам потрібно перевірити чи правильно користувач ввів E-Mail (тобто, наприклад, перевірити в ньому наявність "@"), щоб це зробити користувачеві, треба було б відправити форму з заповненими даними, потім дочекатися, поки вона опрацюється, і лише після цього отримати повідомлення про помилку (якщо вона, звісно, є). Процес занадто довгий. З клієнтськими мовами програма відразу перевірить правильне заповнення форми перед відправленням, і, якщо необхідно, виведе помилку. Звідси ж випливає і те обмеження, що за допомогою клієнтської мови програмування ніщо не може бути записано на сервер.

Найпоширенішою з клієнтських мов є JavaScript, розробниками якої є компанія Netscape спільно з компанією SunMicrosystems. Інший варіант клієнтської мови це, наприклад, VisualBasicScript (VBS).

Тепер розберемося із серверними мовами програмування. Для початку розглянемо наступну схему:

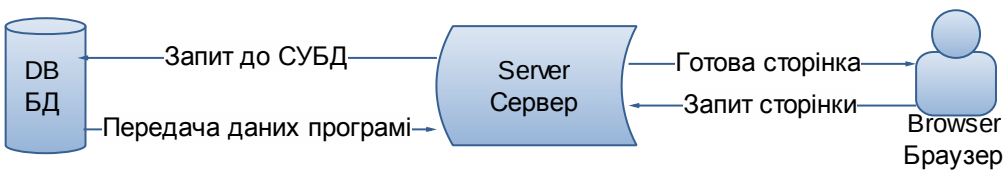

Рисунок 5 - Схема запиту веб-сторінки

Коли користувач дає запит на будь-яку сторінку (переходить на неї за посиланням або вводить адресу в адресному рядку свого браузера), то викликана сторінка спочатку обробляється на сервері, тобто виконуються всі скрипти, пов'язані зі сторінкою, і тільки потім повертається до відвідувача у вигляді простого HTML-документа (тобто відвідувач вже ніяк не зможе побачити код Вашого скрипта). Але робота ваших скриптів вже повністю залежна від сервера, на якому розташований ваш сайт, і від того, яка версія тої чи іншої мови, підтримується хостингом.

Серверні мови програмування відкривають перед програмістом великі простори в діяльності, проте, скільки б не писали люди, які просувають мову, що їх мова дуже легка для навчання, без визначеного багажу знань освоїти її досить важко. Розглянемо детальніше такі основні серверні мови програмування як :

- PHP
- ASP
- Perl
- Java

1.2.1 PHP

Мова, що виконується на стороні веб-сервера, написана мовою C ++, тому містить багато спільного з PHP. Виникнення мови PHP пов’язане з людини на ім’я Рас мус Лердорф (Rasmus Lerdorf) в 1995 році, коли він створює просту програму на Perl, яка представляє собою скрипт по підрахунку відвідування його резюме. Завоювавши велику популярність, скрипт вимагав свого доопрацювання, і тоді з'являється перша версія PHP, написана на C - PHP / FI (Personal Home Page / Forms Interpreter), це як би модифікація Perl для роботи з формами. PHP / FI проіснувала до версії 2.0 (випуск - 1997 р.). Після цього на горизонті з'явилися два студента Ізраїльського університету: Енді Гутменс (Andi Gutmans) та Зів Сураскі (Zeev Suraski), вони почали детально вивчати

першоджерела (Sources) мови PHP / FI і визнали її непридатною для створення великих проектів. Тоді вони створили першу офіційну (сучасну) версію PHP - PHP 3.0, відому під назвою PHP: Hypertext Preprocessor. Згодом з'явилися нові завдання, з якими 3.0 версія PHP не справлялася (достатньо подивитися на кількість нових функцій, які з'явилися в PHP 4.0, без яких тяжко уявити сьогодні можливість ефективно працювати з веб-додатками). Розробники почали ретельно працювати над ядром (Kernel) PHP і незабаром з'являється перша стабільна версія PHP - PHP 4.0 (сенсаційна знахідка для веб-програмістів, повністю перероблене ядро)[8].

1.2.2 Perl

Мову Perl сміливо можна вважати нащадком PHP. Дійсно, при першому вивченні мови в очі впадає різка схожість з PHP. Але це тільки на початку, тому що розробників Perl не особливо турбує питання про спрощення мови. На початку Perl розроблявся для ОС сімейства UNIX. Батьком Perl вважається Ларі Вол (Larry Wall), а розробив він мову, спочатку як систему для звітів в Unix в Usenet-конференціях у 1986 році. Користувачам це сподобалося, і вони захотіли більше можливостей, яких Perl в той час не мав. Основна задача Perl - полегшення команд для Shell, але оскільки ми обговорюємо створення веб-станцій, то далі мова піде про застосування Perl для створення веб-сторінок. Питання ж про те, яку мову вибрати: Perl або PHP настільки філософське, як, наприклад, питання про виникнення людини на землі[9]. Тому підбір тої чи іншої мови є суто індивідуальним.

1.2.3 ASP

ASP – це технологія від Microsoft, що дозволяє легко розробляти програми для World Wide Web. Вона працює на платформі ліній операційних систем Windows NT та на веб-сервері IIS. ASP не є мовою програмування – це лише технологія попередньої обробки, що дозволяє підключати програмні модулі під час процесу формування Web-сторінки. Відносна популярність ASP

заснована на простоті використовуваних мов сценаріїв (VBScript або JScript) і можливості використання зовнішніх COM-компонентів[10].

Технологія ASP отримала свій розвиток у вигляді ASP.NET - нової технології створення веб-додатків, заснованої на платформі Microsoft. NET.

1.2.4 Ruby

Ruby – це скриптова мова високого рівня для швидкого і зручного об'єктно-орієнтованого програмування. Вона має незалежно від операційної системи реалізацію multi-, сувору динамічну типізацію, «збирач сміття» та багато інших можливостей. Ruby близька за особливостями синтаксису до мови Perl, та за об'єктно-орієнтованим підході до Smalltalk. Також деякі риси мови взяті з Python, Dylan та CLU.

Кросплатформність реалізації інтерпретатора мови є повністю вільною, поширюється з відкритими вихідними текстами, можливістю копіювання та модифікації. Останньою є версія 1.8.5, що вийшла 28 серпня 2006року.

Ruby on Rails – об'єктно-орієнтований програмний каркас для створення веб-додатків, написаний мовою програмування Ruby. Ruby on Rails надає каркас модель-представлення-контролер (Model-View-Controller) для веб-додатків, а також забезпечує їх інтеграцію з веб-сервером і сервером бази даних.

Ruby on Rails є відкритим програмним забезпеченням і розповсюджується під ліцензією MIT.

1.2.5 Java

Java – об'єктно-орієнтована мова програмування, що розроблялася компанією Sun Microsystems з 1991 року і офіційно випущена 23 травня 1995 року. Спочатку нову мову програмування називали Oak (James Gosling) і розробили для побутової електроніки[11], але згодом вона була перейменована в Java і почала використовуватися для написання апплетів, додатків і серверного програмного забезпечення.

Програми на Java можуть транслюватися в байт-код, що виконується на віртуальній java-машині (JVM-програмі, що обробляє байт-код), яка передає інструкцію обладнанню, як інтерпретатор, але з відмінністю у тому, що байт-код на відміну від тексту, обробляється значно швидше.

Переваги подібного способу виконання програм - у повній незалежності байт-коду від ОС і устаткування, що дозволяє виконувати Java програми на будь-якому пристрої, який підтримує віртуальну машину. Іншою важливою особливістю технології Java є гнучка система безпеки, завдяки тому, що виконання програми повністю контролюється віртуальною машиною. Будь-які операції, що перевищують встановлені повноваження програми (наприклад, спроба несанкціонованого доступу до даних або з'єднання з іншим комп'ютером) викликають негайне їх переривання. Це дозволяє користувачам завантажувати програми, написані на Java, на їх комп'ютери (або інші пристрої, наприклад, мобільні телефони) з невідомих джерел, при цьому не побоюючись зараження вірусами, втрати цінної інформації і т. п.

Часто до недоліків цього підходу відносять те, що виконання байт-коду віртуальною машиною може знижувати продуктивність програм і алгоритмів, реалізованих на мові Java. Дане твердження справедливе для перших версій віртуальної машини Java, однак останнім часом воно практично втратило актуальність. Цьому сприяли ряд удосконалень: застосування технології JITs (Just-In-Time compiler), що дозволяє переводити байт-код в машинний код під час виконання програми з можливістю збереження версій класу в машинному коді, широке використання Native-коду в стандартних бібліотеках, а також апаратні засоби, що забезпечують прискорену обробку байт-коду (наприклад, технологія Jazelle, підтримувана деякими процесорами фірми ARM)[12].

Всередині Java існує 3 основних сімейства технологій:

– J2EE – Java Enterprise Editon, для створення програмного забезпечення рівня підприємства;

– J2SE – Java Standard Editon, для створення додатків користувача, в першу чергу для настільних систем;

– J2ME – Java Micro Edition, для використання в пристроях, обмежених у обчислювальній потужності, в тому числі мобільні телефони, PDA, вбудованої системи.

1.2.6 Порівняльна характеристика мов веб-програмування.

Характеристики можливостей мов програмування зведені в табл. 1.1.

Таблиця 1.1 - Можливості мов програмування

Можливість\Мова	**Java**	**JavaScript**	**Perl**	**PHP**	**Ruby**
Шаблони/Generics	+	X	x	-	X
Створення об’єктів на стр.	+	?	-	-	-
Підтримка Unicode в ідентифікаторах	+	+	+ /-	- /+	-
Контроль виходу за межі масиву	+	?	x	x	?
Збір сміття	+	+	+	+	+
Цілі числа довільної довжини	+	?	+	?	+
Вивід типів-аргументів при виклиці методу	+	X	x	x	X
Аліаси типів	-	X	x	x	X
Параметричний поліморфізм	+	-	x	+	X
Параметричний поліморфізм із коваріантністю	-	-	?	?	X
Параметричний поліморфізм вищих порядків	-	-	x	?	X
Перегрузка функцій	+	+/-	-	-	-
Іменовані параметри	-	+/-	+	-	+
Значення параметрів за замовчуванням	-	-	+ /-	+	+
Локальні функції	+/-	+	+	+	-

Можливість\Мова	**Java**	**JavaScript**	**Perl**	**PHP**	**Ruby**
			/-		
Лексичні замкнення	-	+	+	?	+
Вивід сигнатури для локальних функцій	-	X	x	?	X
Кортежиі	-	-	-	-	+
Співставлення зі зразком	-	-	+	?	-
Цикл foreach	+	+	+	+	+
Інформація про типи в runtime	+	+	+	+	+
Інформація про типи-параметри в runtime	-	+	?	+	?
Інструкція goto	-	-	+	-	-/+
Інструкції break без мітки	+	+	+	+	+
Інструкція break з міткою	+	+	+	+	+
Підтримка try/catch	+	+	+	+	+
Блок finally	+	+	-	-	+
Блок else (виключення)	+	?	+	-	+
Контрактне програмування	+/-	?	?	?	+/-
Мультиметоди	-	-	-	-	-
Перейменування членів при наслідуванні	-	?	-/+	-	?
Множинне наслідування	-	?	+	-	?
Рішення конфлікту імен при множинному наслідуванні	x	?	+	x	?
Інтерфейси	+	?	?	+	?

Шаблони / Generics – наявність в даній статично типізованій мові інструменту для узагальненого програмування, на зразок Templates в C ++ або generics в C #.

Об'єкти на стеці – можливість створювати екземпляри об'єктів не в купі, а на стеці. Підтримка Unicode в маркер – можливість включення Unicode-символів (наприклад, літер національних алфавітів) в ідентифікатор.

Збірка сміття – можливість використовувати автоматичний процес збирання сміття (звільнення пам'яті, зайнятої об'єктами, що не використовуються).

Цілі числа довільної довжини – підтримка цілих чисел необмеженої розрядності. Повинна бути можливість записати як завгодно велике ціле число за допомогою літерала.

Висновок типів-аргументів при виклику методу – можливість не вказувати явно типи-аргументи при виклику Generic-методу, якщо вони можуть бути виведені з типів звичайних аргументів.

Аліаси типів – можливість визначити видимий глобально (за межами одиниці компіляції) аліас типу, повністю еквівалентний вихідному типу. Наприклад, typedef в Ci. Директива Using в C # не підходить під цей критерій через локальну область дії.

Параметричний поліморфізм – наявність тілобезпечного параметричного поліморфізму (aka generic types). Допускає можливість вказувати Constraints або type classes для типів-параметрів.

Параметричний поліморфізм з коваріантністю – наявність контраваріантних type parameters. У деяких мовах може бути лише часткова підтримка (наприклад, тільки в інтерфейсах делегатів). У такому випадку, відзначено + / -.

Параметричний поліморфізм вищих порядків – можливість створювати type constructors вищих порядків.

Перевантаження функцій – можливість перевантаження функцій / методів за кількістю та типами параметрів.

Іменовані параметри – можливість при виклику функції / методу вказувати імена параметрів і міняти їх місцями.

Значення параметрів за замовчуванням – можливість при виклику функції / методу опускати деякі параметри, щоб при цьому підставлялось значення за замовчуванням, вказане при визначенні функції.

Локальні функції – можливість визначати локальну функцію всередині іншої функції / методу. Мається на увазі можливість використовувати всередині локальної функції локальні змінні з зовнішнього блоку.

Лексичні замикання – можливість використовувати локальну або лямбда-функцію (анонімний делегат) за межами функції-контейнера з автоматичним збереженням контексту (локальних змінних) функції-контейнера.

Висновок сигнатури для локальних функцій – зазначає чи може сигнатура локальної функції бути виведена з використання. Не застосовується для мов з динамічною типізацією. Ставимо «-», якщо мова не підтримує локальних функцій.

Цикл foreach – наявність можливості перебрати всі елементи колекції з допомогою циклу foreach. Якщо в мові є еквівалентна або більш сильна можливість (на зразок list comprehensions), ставимо +.

Інформація про типи в Runtime – можливість дізнатися точний тип об'єкта в Runtime.

Інформація про типи-параметрів в Runtime – можливість дізнатися в Runtime інформацію про тип, з якого інстанціірований Generic-тип. Якщо мова не підтримує Generic-типи, то ставимо «-» . Якщо інформація про типи стирається в Runtime (використовується erasure) також - «-».

Інструкція Break без позначки – підтримка інструкції Break без позначки (безумовний вихід з найближчого циклу) і відповідної інструкції Continue. Наявність у мові інструкції Break, що відноситься до Switch чи іншої конструкції, не впливає на це поле.

Інструкція Break з міткою – підтримка інструкції Break з міткою (безумовний вихід з циклу, помічені міткою) і відповідної інструкції Continue.

Наявність в мові інструкції Break, що відноситься до Switch чи іншої конструкції, не впливає на це поле.

Підтримка Try / Catch – підтримка обробки винятків з допомогою Try / Catch або еквівалентної конструкції[13].

Блок Finally – підтримка блоку Finally при обробці винятків або еквівалентної конструкції.

Блок Else (виняток) – підтримка блоку else при обробці винятків (дії, які виконуються при завершенні блоку Try без винятку).

Контрактне програмування – можливість ставити перед- і постумови для методів та інваріанти для класів; множинне успадкування. Можливість успадкувати клас відразу від декількох класів (не інтерфейсів).

Мультиметоди – динамічна (run time) диспетчеризація функції залежно від типів декількох аргументів. У мовах з "message passing" ООП схожий функціонал реалізується за допомогою патернів "Visitor".

Рішення конфлікту імен при множинному успадкуванні – рішення для випадку ромбовидного спадкування (B нащадок A, C нащадок A, D нащадок B і C). Рішення може прийматися як для всього класу, так і для кожного поля / методу окремо.

Інтерфейси – семантична і синтаксична конструкція в коді програми, яка використовується для специфікації послуг, що надаються класом[14].

Для порівняння мов програмування розглянемо також взаємодію їх типів даних подану у табл. 1.2.

Таблиця 1.2 - Типізація мов-програмування.

Можливість\Мова	**Java**	**Perl**	**PHP**	**Ruby**
Статична типізація	+	+/-	-	-
Динамічна типізація	-	+	+	+
Явна типізація	+	-/+	+/-	-
Неявна типізація	-	+	+	+
Явне наведення типів	+	?	+	+

Можливість\Мова	**Java**	**Perl**	**PHP**	**Ruby**
Неявне наведення типів без втрати даних	+	+	+	+
Неявне наведення типів із втратою	-	+	+	-
Неявне наведення типів в неоднозначних ситуаціях	-	+	+	-

Статична типізація – змінні і параметри методів / опцій зв'язуються з типами в момент оголошення і не можуть бути змінені пізніше.

Динамічна типізація – змінні і параметри методів / опцій зв'язуються з типами в момент присвоювання значення (або передачі параметра в метод / функцію), а не в момент оголошення змінної або параметра. Одна і та ж змінна в різні моменти може зберігати значення різних типів.

Явна типізація – типи змінних і параметрів вказуються явно.

Неявна типізація – типи змінних і параметрів не вказуються явно. Неявна типізація може бути і статичною, у такому випадку типи змінних і параметрів обчислюються компілятором[15].

Явне наведення типів – для використання змінної такого типу там, де передбачається використання змінної іншого типу, потрібно (можливо) явно виконати перетворення типу.

Неявне наведення типів без втрати даних – неявне наведення типів в таких ситуаціях, де не відбувається втрати даних, наприклад, використання цілого числа там, де передбачалося використання числа з плаваючою точкою.

Неявне наведення типів з втратою даних – неявне наведення типів в таких ситуаціях, де може відбутися втрата даних - наприклад, використання числа з плаваючою точкою там, де передбачалося використання цілого числа. Неявне наведення типів у неоднозначних ситуаціях, наприклад, використання рядка там, де передбачалося число або навпаки. Класичний приклад: скласти число 1

з рядком "2" - результат може бути як число 3, так і рядок "12". Інший приклад - використання цілого числа там, де очікується логічне значення (Boolean).

Важливою частиною кожної мови є підтримка засобів компіляції і інтерпретації. Тому розглянемо кожну мову згідно цих параметрів табл.1.3.

Таблиця 1.3 - Характеристики копіляції/інтерпретації.

Можливість\Мова	**Java**	**Perl**	**PHP**	**Ruby**
Open-source компілятор (інтерпретатор)	+	+	+	+
Можливість компіляції	+	+	+	+
Bootstrapping компілятор	+	?	x	+
Багатопоточна компіляція	+	?	?	X
Інтерпретатор командної строки	-	+	+	+
Умовна компіляція	- /+	+	?	X

Open-Source компілятор (інтерпретатор) – наявність повноцінного Open-Source компілятора (для різних мов - інтерпретатора). Якщо існує Open-Source компілятор, але він підтримує не всі можливості мови, то стоїть «- / +».

Можливість компіляції – можливість компіляції в Byte-код з можливістю JIT-компіляції. Якщо мову компілюють в код на іншій мові (наприклад, C), який потім компілюють в нативний код, то стоїть «+».

Bootstrapping компілятор – наявність повноцінного Open-source bootstrapping компілятора (тобто, компілятора, написаного тою ж мовою, яку він компілює, і успішно компілює самого себе). Якщо існує Open-source bootstrapping компілятор, але він підтримує не всі можливості мови, то стоїть «- / +».

Багатопоточна компіляція – можливість компілятора на багатопроцесорних системах використовувати декілька потоків для

прискорення компіляції. Якщо мова не підтримує компіляцію, то стоїть «х» (не застосовується).

Інтерпретатор командного рядка – можливість вводити інструкції мови рядок за рядком з їх негайним виконанням. Може використовуватися в якості калькулятора.

Умовна компіляція – можливість вмикати / вимикати частини коду залежно від значення символів умовної компіляції (наприклад, за допомогою # If ... # endif в C ++)

Загальні характеристики зведені в табл. 1.4.

Таблиця 1.4. - Порівняльна характеристика мов веб– програмування.

Мова веб – програмування	**Недоліки**	**Переваги**
PHP	- незручний синтаксис; - відносно низька продуктивність праці.	- є на переважній більшості хостингів; - легко поєднується з html; - легка у використанні; - володіє багатьма бібліотеками та класами; - наявна різноманітна документація та підручники.
Ruby	- незначне розповсюдження на хостингах; - дефіцит хороших паперових підручників.	- гарно продуманий синтаксис; - найбільш «високорівнева мова» з тих, що розглядаються; - маленький об’єм коду; - легка у освоєнні та «швидкому програмуванні».

Мова веб – програмування	**Недоліки**	**Переваги**
Perl	- застаріла технологія; - недоліки ті ж, що і в PHP	- є на переважній більшості хостингів; - легка у використанні; - володіє багатьма бібліотеками та класами; - наявна різноманітна документація та підручники; - швидша за PHP.
Мова веб – програмування	Недоліки	Переваги
Java	- тяжка у освоєнні; - має велику кількість досить запутаних реалізацій; - мало розповсюджена на безкоштовних хостингах.	- Найшвидша, найгнучкіша та найрозповсюдженіша мова з тих, що розглядаються; - легко поєднується з html, «маскуючись» під JSP сторінки.

1.3 Огляд баз даних і СУБД

Традиційно використовуються для WEB-розробок мови програмування (Perl, PHP, ASP та інші), що дозволяють реалізовувати практично будь-які завдання. Але обробляти з їх допомогою великі обсяги даних, що мають до того ж складну структуру, досить важко. Розробка подібних програм вимагає зростаючих затрат праці програмістів, у геометричній прогресії зростає обсяг

програмного коду та кількість помилок, знижується надійність програмного забезпечення.

У такій ситуації на допомогу програмісту приходять бази даних. Відповідно до класичного визначення, база даних - це впорядкована сукупність інформації, що зберігається у вигляді множин, кожна з яких містить записи уніфікованого вигляду. Системи управління базами даних (СУБД) надають програмісту найпотужніший інструментарій для створення, оновлення та обробки великих обсягів інформації, що має складну структуру.

У класичній теорії виділяють три типи, три структури баз даних: ієрархічну, мережеву та реляційну. В даний час домінуюче становище займають реляційні бази даних.

Реляційна модель вимагає від СУБД більш високого рівня складності. В ній робиться спроба позбавити програміста виконання рутинних операцій з управління даними, настільки характерних для ієрархічної і мережевої моделей.

Реляційна модель бази даних являє собою централізоване сховище таблиць, що забезпечує безпечний одночасний доступ до інформації з боку багатьох користувачів. У рядках таблиць частина полів містить дані, що відносяться безпосередньо до запису, а частина - посилання на записи інших таблиць. Таким чином, зв'язки між записами є невід'ємною властивістю реляційної моделі[16].

Кожен запис таблиці має однакову структуру. Наприклад, в таблиці, яка містить описи автомобілів, у всіх записів буде один і той же набір полів: виробник модель, рік випуску, пробіг і т.д. Такі таблиці легко зображувати в графічному вигляді.

У реляційній моделі досягається інформаційна та структурна незалежність. Записи не пов'язані між собою настільки, щоб зміна одного з них торкнулося інші, а зміна структури бази даних не обов'язково призводить до перекомпіляції працюючих з нею програм.

У реляційних СУБД застосовується мова SQL, що дозволяє формулювати довільні, нерегламентовані запити. Це мова четвертого покоління, тому будь-

який користувач може швидко навчитися складати запити. До того ж, існує безліч програм, які дозволяють будувати логічні схеми запитів у графічному вигляді. Все це відбувається за рахунок посилення вимог до продуктивності комп'ютерів. На щастя, сучасні обчислювальні потужності більш ніж адекватні.

Реляційні бази даних страждають від відмінностей у реалізації мови SQL, хоча це і не проблема реляційної моделі. Кожна реляційна СУБД реалізує якусь підмножину стандарту SQL плюс набір унікальних команд, що ускладнює завдання програмістам, які намагаються перейти від однієї СУБД до іншої. Доводиться робити нелегкий вибір між максимальною переносимістю і максимальною продуктивністю. У першому випадку потрібно дотримуватися мінімального загального набору команд, які підтримуються в кожній СУБД. У другому випадку програміст просто зосереджується на роботі в даній конкретній СУБД, використовуючи переваги її унікальних команд і функцій[9].

MySQL - це найпоширеніша реляційна СУБД. Але теорія баз даних не стоїть на місці. З'являються нові технології, які розширюють реляційними моделями.

1.3.1 MySQL

MySQL виникла як спроба застосувати mSQL до власних розробок компанії: таблиць, для яких використовувалися ISAM - підпрограми низького рівня. В результаті був вироблений новий SQL-інтерфейс, але API-інтерфейс залишився у спадок від mSQL. Звідки походить назва «MySQL» - достеменно не відомо. Розробники дають два варіанти: або тому, що практично всі напрацювання компанії починалися з префікса My, або на честь дівчинки на ім'я My, дочки Майкла Монті Віденіуса, одного з розробників системи.

Логотип MySQL у вигляді дельфіна носить ім'я «Sakila». Він був обраний з великого списку запропонованих користувачами «імен дельфина». Ім'я «Sakila» було відправлено Open Source-розробником Ambrose Twebaze[17].

Клієнтська програма MySQL являє собою утиліту командного рядка. Ця програма підключається до сервера по мережі. Команди, що виконуються сервером, зазвичай пов'язані з читанням і записом даних на жорсткому диску.

Клієнтські програми можуть працювати не тільки в режимі командного рядка. Є й графічні клієнти, наприклад MySQL GUI, PhpMyAdmin та ін., але вони - тема окремого курсу.

MySQL взаємодіє з базою даних на мові, що зветься SQL (Structured Query Language - мова структурованих запитів).

SQL призначена для маніпуляції даними, які зберігаються в системах управління реляційними базами даних (RDBMS). SQL має команди, з допомогою яких дані можна вилучати, сортувати, оновлювати, вилучати та додавати. Стандарти мови SQL визначає ANSI (American National Standards Institute). В даний час діє стандарт, прийнятий у 2003 році (SQL-3).

SQL можна використовувати з такими RDBMS як MySQL, mSQL, PostgreSQL, Oracle, Microsoft SQL Server, Access, Sybase, Ingres. Ці системи RDBMS підтримують всі важливі і загальноприйняті оператори SQL, однак кожна з них має безліч своїх власних патентованих операторів і розширень.

SQL є загальною мовою запитів для декількох баз даних різних типів. Даний курс розглядає систему MySQL, яка є RDBMS з відкритим вихідним кодом, і доступна для завантаження на сайті MySQL.com.

Ось як характеризують MySQL її розробники: MySQL - це система управління базами даних.

База даних являє собою структуровану сукупність даних. Ці дані можуть бути будь-якими - від простого списку майбутніх покупок до переліку експонатів картинної галереї або величезної кількості інформації в корпоративній мережі. Для запису, вибірки та обробки даних, що зберігаються в комп'ютерній базі даних, необхідна система управління базою даних, яка і є ПО MySQL. Оскільки комп'ютери чудово справляються з обробкою великих обсягів даних, управління базами даних відіграє центральну роль в обчисленнях. Реалізовано таке управління може бути по-різному - як у вигляді окремих утиліт, так і у вигляді коду, що входить до складу інших програм.

– MySQL - це система управління реляційними базами даних.

У реляційній базі дані зберігаються в окремих таблицях, завдяки чому досягається виграш у швидкості та гнучкості. Таблиці поєднані між собою за допомогою зв’язків, завдяки чому забезпечується можливість об'єднувати при виконанні запиту дані з декількох таблиць. SQL як частину системи MySQL можна охарактеризувати як мову структурованих запитів плюс найбільш поширена стандартна мова, що використовується для доступу до баз даних.

– Програмне забезпечення MySQL - це ПЗ з відкритим кодом.

ПЗ з відкритим кодом означає, що застосовувати і модифікувати його може будь-який бажаючий. Таке ПЗ можна отримувати за допомогою Internet і використовувати безкоштовно. При цьому кожен користувач може вивчити вихідний код і змінити його відповідно до своїх потреб.

– Технічні можливості СУБД MySQL

ПО MySQL є системою клієнт-сервер, що містить багатопоточний SQL-сервер, який забезпечує підтримку різних обчислювальних машин баз даних, а також кілька різних клієнтських програм і бібліотек, засоби адміністрування та широкий спектр програмних інтерфейсів (API).

– Безпека

Система безпеки заснована на привілеї та паролі з можливістю верифікації з віддаленого комп'ютера, за рахунок чого забезпечується гнучкість і безпека. Паролі при передачі по мережі під час з'єднання з сервером шифруються. Клієнти можуть з'єднуватися з MySQL, використовуючи сокета TCP / IP, сокета Unix або іменовані канали (named pipes, під NT)

– Місткість даних

Починаючи з MySQL версії 3.23, де використовується новий тип таблиць, максимальний розмір таблиці доведений до 8 мільйонів терабайт (263 Bytes). Однак слід зауважити, що операційні системи мають свої власні обмеження за розміром файлів. Нижче наведено кілька прикладів:

- 32-розрядна Linux-Intel - розмір таблиці 4 Гб.
- Solaris 2.7 Intel - 4 Гб
- Solaris 2.7 UltraSPARC - 512 Гб

- WindowsXP - 4 Гб

Як можна побачити, розмір таблиці в базі даних MySQL звичайно лімітується операційною системою. За замовчуванням MySQL-таблиці мають максимальний розмір близько 4 Гб. Для будь-якої таблиці можна перевірити / визначити її максимальний розмір за допомогою команд SHOW TABLE STATUS або myisamchk-dv table_name. Якщо велика таблиця призначена тільки для читання, можна скористатися myisampack, щоб об'єднати декілька таблиць в одну і стиснути її. Зазвичай myisampack стискує таблицю принаймі на 50%, тому в результаті можна отримати дуже великі таблиці[10].

MySQL функціонує на більшості платформ:

- AIX,
- BSDi,
- FreeBSD,
- HP-UX,
- GNU/Linux,
- Mac OS X,
- NetBSD,
- OpenBSD,
- OS/2 Warp,
- SGIIRIX,
- Solaris,
- SunOS,
- SCO OpenServer,
- SCO UnixWare,
- Tru64,
- Windows 95,
- Windows 98,
- Windows NT,
- Windows 2000,

- Windows XP,
- Windows Server 2003
- Windows Vista.

MySQL має API для мов C, C++, Ейфель, Java, Лисп, Perl, PHP, Python, Ruby, Smalltalk та Tcl, бібліотеки для мов платформи .NET, а також забезпечує підтримку для ODBC за допомогою засобів ODBC-драйвера MyODBC.

1.3.2 PostgreSQL

PostgreSQL починає свій «родовід» від некомерційної СУБД Postgres, розробленою, як і багато Open Source-проектів, в Каліфорнійському університеті в Берклі. До розробки Postgres, що почалася в 1986-му році, мав безпосереднє відношення Майкл Стоунбрейкер, керівник більш раннього проекту Ingres, на той момент уже придбаного компанією Computer Accociates. Сама назва «Postgres» розшифровується як «Post Ingres», відповідно, при створенні Postgres були застосовані багато ким вже раніше зроблені напрацювання.

Стоунбрейкер та студенти розробляли нову СУБД протягом восьми років, з 1986 по 1994 рік. За цей період в синтаксис були введені процедури, правила, нестандартні типи і багато інших компонентів. Робота не пройшла даремно - в 1995 розробка знову розділилася: Стоунбрейкер використовував набутий досвід у створенні комерційної СУБД Illustra, просуваючи його власною однойменною компанією (придбаною внаслідок компанією Informix), а його студенти розробили нову версію Postgres - Postgres95, в якій мова запитів POSTQUEL - спадщина Ingres - була замінений на SQL.

У цей момент розробка Postgres95 була виведена за межі університету і передана команді ентузіастів. З цього моменту ця СУБД отримала ім'я, під яким вона відома і розвивається в цей момент - PostgreSQL.

PostgreSQL базується на мові SQL і підтримує багато з можливостей стандарту SQL: 2003 (ISO / IEC 9075). На даний момент (версія 8.3.5), в PostgreSQL є наступні обмеження, що приведені в табл 1.5.

Таблиця 1.5 - Обмеження PostgreSQL

Максимальний розмір бази даних	Не має обмежень
Максимальний розмір таблиці	32 ТБайт
Максимальний розмір запису	1,6 ТБайт
Максимальний розмір поля	1 ГБайт
Максимум записів у таблиці	Не має обмежень
Максимум полів у таблиці	250—1600, залежно від типу полів
Максимум індексів у таблиці	Не має обмежень

Сильними сторонами PostgreSQL вважаються:

- підтримка БД практично необмеженого розміру;

- потужні і надійні механізми транзакцій і реплікацій;

- наслідування;

- легке масштабування.

Згідно з результатами автоматизованого дослідження різного ПЗ на предмет помилок, у вихідному коді PostgreSQL було знайдено 20 проблемних місць на 775 000 рядків вихідного коду (в середньому, одна помилка на 39 000 рядків коду). Для порівняння: MySQL - 97 проблем, одна помилка на 4 000 рядків коду; FreeBSD (повністю) - 306 проблем, одна помилка на 4 000 рядків коду; Linux (тільки ядро) - 950 проблем, одна помилка на 10 000 рядків коду.

На базі PostgreSQL компанією EnterpriseDB створені більш потужні варіанти цієї СУБД, що є платними для комерційного використання - Postgres Plus (складається цілком тільки з продуктів з відкритими вихідними кодами; плата потрібна тільки при необхідності придбання комерційної підтримки

продукту) і Postgres Plus Advanced Server (розширення PostgreSQL спеціальними можливостями для забезпечення сумісності з Oracle Database). У комплекті поставки даних продуктів міститься великий набір ПЗ для розробників і DBA:

- Postgres Studio - більш потужний аналог pgAdmin;
- Postgres Plus Debugger - відладчик для коду на PL / pgSQL, інтегрований з попереднім пакетом;
- Migration Studio - інструмент для автоматичного перетворення баз даних з MySQL / Oracle в PostgreSQL

1.3.3 Oracle

СУБД Oracle широко відома як в нашій країні, так і у світі. На її основі успішно функціонує безліч інформаційних систем. Сьогодні вже нікого не треба переконувати в її продуктивності, функціональній потужності, надійності, масштабованості, захищеності, відкритості, можливості підтримувати системи з найвищими вимогами щодо захисту даних, безперервності роботи, системи, що працюють в інтернеті і інтранету.

Більш цікаво подивитися, куди ж рухається СУБД Oracle, а разом з нею і вся індустрія СУБД, Oracle дуже часто був законодавцем в області нових напрямків розвитку СУБД. За останні кілька років на наших очах з'явилися такі платформи для побудови клієнт-серверних і інтернет додатків, як Oracle 8i (8.0 та 8.1), Oracle 9i, а в найближчі місяці виходить у продаж нова версія Oracle 9i реліз 2 (9.2). А в надрах корпорації Oracle вже зріють ідеї про Oracle 10i[18].

Кожна з цих версій містить кілька сотень нових можливостей у порівнянні з попередньою версією. На основі аналізу цих нових можливостей ми і спробуємо в цій роботі виділити найбільш цікаві та важливі тенденції розвитку СУБД кінця 20 і початку 21 століття.

За більш ніж 20 років свого розвитку корпорація Oracle була піонером у багатьох областях побудови СУБД. До недавнього часу далеко не повний список таких "піонерських" досягнень виглядав наступним чином:

- Перша комерційна SQL СУБД

- Підтримка безлічі обчислювальних платформ
- Підтримка архітектури клієнт / сервер
- Підтримка моделі багатоверсійного запису (Multi-version Read Consistency)
- Підтримка кластерної і MPP архітектури
- Підтримка розподілених транзакцій
- Підтримка активних бізнес правил
- Підтримка паралельної обробки
- Оптимізація роботи з сховищами даних
- Підтримка всього спектру Multimedia
- Підтримка об'єктно-реляційної моделі
- Підтримка Messaging

Таким чином, почавши з першої комерційної реляційної СУБД, Oracle реалізував системи підвищеної надійності, системи для підтримки сховищ даних і аналітичних систем, розподілених БД. Почавши з чисто реляційної моделі, Oracle послідовно реалізував зберігання та обробку таких мультимедійних даних, як текст, зображення, відео, аудіо, просторова інформація. Потім СУБД стала підтримувати і об'єктну модель. Наступним кроком стало встроювання все в ту ж СУБД засобів підтримки багатовимірної моделі (OLAP) і засобів для Data Mining, засобів підтримки спеціальних моделей, типових для сховищ даних. І, нарешті, останнім бурхливо розвиваються напрямком стало встроювання в СУБД Oracle підтримки XML моделі.

Оскільки напрямків розвитку досить багато, поговоримо про найбільш цікаві з них, серед яких хотілося б особливо виділити наступні[19]:

- встроювання в єдину СУБД засобів ефективного створення та підтримки роботи дуже великими (до 512 Pb) БД (VLDB) сховищами даних, засобів підтримки багатовимірних OLAP технологій та алгоритмів Data Mining (автоматичне дослідження даних) із збереженням всіх переваг комерційної

СУБД, засобів проектування і виконання процедур вилучення, погодження, очищення, передачі і завантаження даних (ETL), засобів персоналізації;

- розвиток в СУБД інтернет технологій, таких як інтернет файлова система (IFS), віртуальна машина Java (підтримка Java 1.3), робота з динамічними Web сервісами, засоби проектування та реалізації порталів та портлетов;

- реалізація нових засобів розподілу інформації між різними БД, серверами, додатками (можливо від різних виробників). Почавши з підтримки розподілених БД і реплікації, Oracle реалізував підтримку систем роботи з чергами повідомлень, Workflow, автоматичне підтримання логічної і фізичної резервної БД, завантаження даних у сховища і Data Store і, нарешті, єдину технологію, що об'єднує всі вище перераховані - Oracle Stream;

- багато змін останнім часом було зроблено в галузі удосконалення захисту даних. Найбільш цікавими можна вважати реалізацію концепції приватної (персональної) БД (Private Database) на основі механізму Fine Grain доступу, коробкове рішення по захисту даних з використанням міток секретності строк даних (Label Security), нових засобів кодування даних у БД і при передачі;

- дуже багато зусиль було в останні роки витрачено на перетворення СУБД Oracle в безперервно працюючу і завжди доступну для додатків і користувачів платформу. Комплекс рішень, таких як Real Application Cluster, логічна і фізична Standby БД, виконання адміністрування БД без її зупинки і уповільнення роботи з об'єктами БД дозволяють реалізувати на основі Oracle системи з часом простою 5 - 15 хвилин на рік. А нова можливість Flash Back дозволяє користувачам легко подорожувати в минуле і працювати на експлуатаційній системі зі своїми вчорашніми, позавчорашній і т. д. даними. Крім того, підключення на льоту все нових і нових вузлів кластеру дозволяє плавно збільшувати потужність обчислювальної системи;

- ну і звичайно XML DB. Тепер поряд зі звичайними реляційними даними ми можемо зберігати в той же БД XML документи і швидко працювати

з ними. При цьому документи зберігаються в СУБД і використовують всі переваги такого зберігання. Спеціальні механізми зберігання, індексування, побудови XML View тощо дозволяють не тільки ефективно зберігати, але й запитувати і змінювати ці дані та їх частини. Причому традиційні SQL операції вміють працювати як з реляційними даними, так і з XML файлами, і навпаки, за допомогою XML операцій можна працювати з SQL даними. А крім цього, XML DB дозволяє розкласти XML документи по ієрархічним папках, встановити для них додаткові засоби контролю доступу (ACL), здійснити пошук потрібних XML документів по контексту і т д;

- постійно удосконалюються і спрощуються засоби управління СУБД. Багато операції з адміністрування БД, раніше вимагали участі адміністратора БД, тепер виконуються автоматично. А графічний інструмент адміністратора БД - Oracle Enterprise Manager дозволяє, кинувши швидкий погляд на всю прикладну систему в цілому, побачити "вузькі місця" і далі допомагає деталізувати проблеми і підказує методи їх усунення. Інтелектуальна експертна система допоможе налаштувати Вашу БД.

Якщо глобально подивитися на напрямок всіх основних змін за останні роки, то їх можна об'єднати в наступні групи:

- висока доступність
- масштабованість і продуктивність
- захист даних
- розвиток засобів розробки
- керованість
- робота з інтернет контентом і мультимедіа
- Business Intelligence
- підтримка хостингу

Технічні характеристики зведені в табл. 1.6.

Таблиця 1.6 - Технічні характеристики БД Oracle

Об’єм бази даних	До 65535 таблиць кожна об’ємом до 12Тб
Кількість записів в одній таблиці	до 1 млрд.
Розмір запису	до 64К
Кількість полів в запису	до 250
Мінімальний об’єм пам’яті, що займає ядро СУБД	3Мб (для спеціалізованих версій – від 800К)
Захист даних	2 -й клас захисту даних від несанкціонованого доступу і 2-й рівень контролю за відсутністю недекларованих можливостей
Формати для повнотекстової індексації	PDF, DOC, TXT, XLS, XML, PS, PPT
Взаємозв’язок з задачами користувача	низькорівневий (CALL) и високорівневий(LinApi) програмні інтерфейси
Програмні інтерфейси	ODBC 3.x, JDBC(1,2,3), DBExpress, Embedded SQL, OLEDB, PERL, PERL/DBI, TCL/TK, PHP, Python, OCI, ADO.NET
Адміністрування	псевдографічні і графічні утиліти для Windows і UNIX – робочий стіл, архіватор БД, конвертор БД, тестування і відновлення БД, міграція БД, налагоджування збережених процедур та тригерів
Архівування	повне, вибіркове, інкрементне, за розкладом, відповідно до скрипту, можливість архівування на стрічку
Реплікація	асинхронна (у тому числі й двонаправлена), можлива реплікація з іншими БД через ODBC

Продовження таблиці 1.6

Синхронізація	з різними СУБД через ODBC за допомогою Online протоколів TCP / IP (в т.ч. і через SSL), HTTP, HTTPS і Offline транспортом - ActiveSync, FTP, E-Mail і т.д.
Засоби розробки	будь-які засоби розробки, що підтримують ODBC, JDBC, DBExpress та ін
Підтримувані платформи	Linux (різні версії та апаратні платформи включаючи Embedded Linux), MCBC, Solaris (різні версії і платформи), Mac OS X, BSD (OpenBSD, FreeBSD, BSDI, NetBSD різних версій), UnixWare, IRIX, AIX, SINIX, QNX, USIX , VxWorks, OS-9, OS-9000, ОС РВ, ІНТРОС, VMS, Windows (95, 98, Me, 2000, XP, NT4), WindowsCE
Типи даних	Char, Varchar, Nchar, Nchar Varying, Byte, Varbyte, Boolean, Smallint, Integer, Bigint, Real, Double, Numeric, Date, Blob, Extfile
Геометричні типи даних	По специфікації OpenGIS: POINT, LINESTRING, POLYGON, MULTIPOINT, MULTILINESTRING, MULTIPOLYGON, GEOMETRYCOLLECTION Для сумісності з PostgreSQL: BOX, LINE, CIRCLE
Геометричні функції	За специфікацією OpenGIS: • функції для створення значень геометричних типів з допомогою текстового та бінарного представлення (GeomFromText, GeomFromWKB і безліч інших);

	• функції для аналізу властивостей геометричних даних (як спільні - Dimension, Envelope, Boundary та ін, так і спеціалізовані для кожного з геометричних типів даних - Length, Area, Centroid та ін); • геометричні оператори (Union, Intersection та ін); функції, що описують відносини між двома значеннями геометричних типів (Distance, Equals, Intersects та ін.)
Підтримувані мережеві протоколи	TCP/IP(в том числі и SSL), SPX, NetBios, Named Pipes

1.3.4 Порівняльна характеристика БД

Для зручності порівняння розглянутих БД всі основні дані зведені в табл. 1.7.

Таблиця 1.7 - Основні характеристики БД.

	Максимальний розмір БД	Максимальний розмір табл.	Максимальний розмір поля	Макс. кількість стовпців в рядку	Макс. розмір типу CHAR	Макс. розмір типу NUMBER	Мін. значення типу DATE	Макс. значення типу DATE
MySQL 5	Необме - жений	2 GB (Win32 FAT32) to 16 TB (Solaris)	64 KB	3398	64 KB (text)	64 bits	1000	9999

Oracle	Необме - жений (4 GB * block size per tablespac e)	4 GB * block size (with BIGFILE tablespace)	Необмежений	1000	4000 B	126 bits	-4712	9999
Postgre SQL	Необме - жений	32 TB	1.6 TB	250-1600 (залеж но від типу)	1 GB	Необмежений	-4713	5874 897

Підтримування відповідними БД операційних систем приведено в табл.1.8.

Таблиця 1.8 - Підтримка ОС базами даних.

Підтримка ОС	Windows	Mac OS X	Linux	BSD	UNIX	AmigaO S	Symbia n
MySQL	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Oracle	Yes	Yes	Yes	No	Yes	No	No
PostgreSQL	Yes	Yes	Yes	Yes	Yes	Yes	No

1.4 Структура порталу

У цьому розділі описано основні принципи роботи сайту, його структура, формується загальне уявлення про подальшу роботу над проектом. Також на

цьому етапі необхідно придумати назви розділів сайту, заголовки сторінок, визначити переходи між ними, тобто продумати логічну структуру розміщення інформації.

Тут же треба врахувати, якщо необхідно, способи спілкування з користувачами в рамках сайту - коментарі до статей, форум, чат, гостьову книгу. В результаті третього етапу має бути сформована ясна і логічна структура розміщення інформації на сайті - ніщо так не втомлює при пошуку потрібної інформації, як погано структуровані сайти.

Проектування інформаційної структури сайту-це ядро процесу створення сайту. Від цього етапу багато в чому залежить і результат. При цьому мається на увазі не лише зовнішній вигляд сторінок сайту, але і якість змісту-і зручність використання. Цей етап набуває особливої важливості при розробці гіпертекстових систем, які являють собою не просто текстовий масив, а інформаційну систему, організовану особливим чином[20].

Проектування сайту - це інженерна задача, в процесі розв'язання якої у неспеціалістів в цій галузі виникає багато питань і проблем.

Коли ми беремо в руки книгу, то в першу чергу звертаємо увагу на обкладинку, а потім продовжуємо знайомитися з нею послідовно, з першої до останньої сторінки. Ще більше значення має обкладинка журналу, але основною його відмінністю від книги є інформаційна структура - журнал можна читати з середини, з початку, з кінця, в довільній послідовності. Він надає більшу свободу в плані подання інформації та подальшого її сприйняття. Інформаційна структура Web-сайту виходить на якісно новий рівень. Його головна відмінність від матеріальних носіїв інформації - нелінійність. Зміст сайту зазвичай представляє собою складну "об'ємну" композицію з складових його об'єктів. Причому складові частини сайту, на відміну від друкованих матеріалів, пов'язані один з одним не фізично, а віртуально.

Ідеологія гіпертексту, покладена в основу "всесвітньої павутини", отже, будь-якого Web-сайту, передбачає перегляд сторінок у довільній послідовності. Web-сторінка не має фіксованого положення усередині сайту, так як автор

може протягати нитки гіпертекстових зв'язків від однієї, розташованої в просторі сторінки, до будь-якої іншої. По гіпертекстових посиланнях користувач може відразу потрапити на будь-яку сторінку, що знаходиться в глибині сайту, не побачивши при цьому ні обкладинки, ні змісту. Все це передбачає особливий підхід до створення гіпертекстових документів та їх об'єднання в організаційну структуру.

На перший погляд, з огляду на сказане, може здатися, що в просторі Web-сайту панує повний хаос. У деяких інтернет-ресурсах це саме так, однак, всередині добре спроектованого сайту основні (магістральні) зв'язку між сторінками завжди складаються в певну структуру. При створенні Web-сайтів використовують декілька типів базових структур (рис. 6):

- послідовна (лінійна) структура;
- ієрархічна (деревоподібна) структура;
- структура системи координат;
- структура мережі (павутина).

При використанні послідовної структури (рис. 6а) елементи шикуються в логічний ланцюжок. Така послідовність зазвичай має яскраво виражені початок і кінець, причому початок знайомства з нею з одного з проміжних елементів, як правило, не має сенсу. Така структура добре підходить для такого матеріалу, як глави книги, розділи віртуальної екскурсії чи подорожі, ланцюжки тестових завдань.

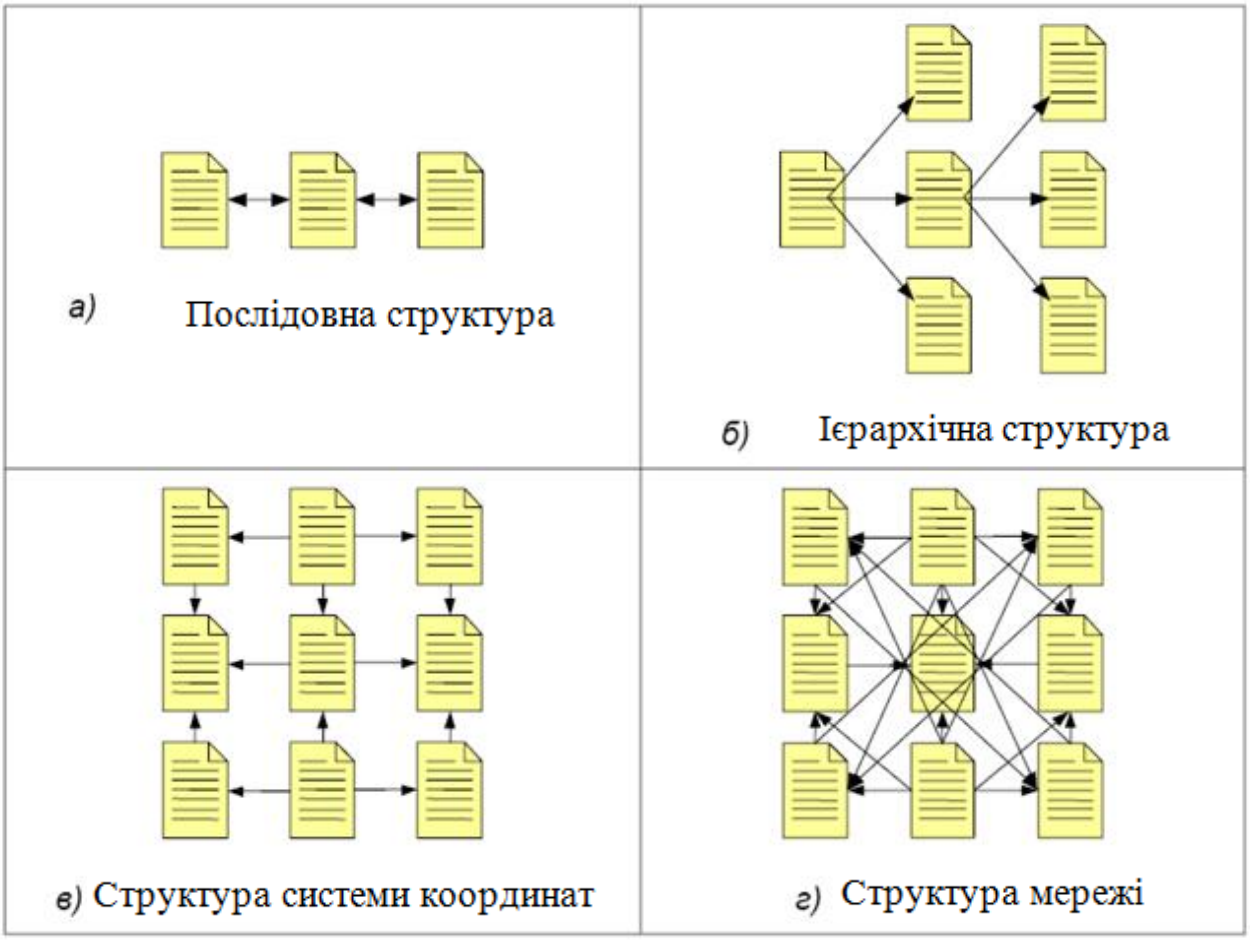

Рисунок 6 - Типів базових структур

Ієрархічна структура (рис. 6б) передбачає, що кожен її елемент (за винятком першого) є підрозділом елемента більш високого рівня. Така структура має чітко виражене початок (“ корінь дерева ”), але не має кінця. Вона передбачає можливість переходу з рівня на рівень, а також переміщення по горизонталі. Деревовидна структура найкраще підходить для організації різнорідної, але добре структурованого матеріалу.

Структура системи координат (рис. 6в) передбачає однорідність складових її інформаційних одиниць і відсутність очевидної ієрархії. Елементи структури є осередками матриці, існує багато шляхів, за якими може переміщатися користувач.

Структура мережі (рис. 6г) наслідує асоціативне мислення, вільного потоку думки, що передбачає ще більшу кількість варіантів маршрутів переміщення по ній. Послідовна і ієрархічна структури в силу своєї простоти

частіше використовуються починаючими авторами і розраховані на найширшу аудиторію. Структури системи координат і мережі дозволяють створювати об'ємні специфічні ресурси, розраховані на досвідчених користувачів.

Виділені варіанти базових структур сайту в чистому вигляді використовуються дуже рідко, переважно у випадку невеликих за обсягом ресурсів. Більшість реальних сайтів має змішану структуру, що представляє собою певну комбінацію базових. На сайтах найчастіше використовують одночасно деревовидну і лінійну структури. Наприклад, на головній сторінці розташовується зміст сайту, що дозволяє переміщуватися по його розділах, організованих в ієрархічну структуру, а всередині розділів, особливо якщо вони великі, документи організовані в послідовну структуру.

Таким чином, перш ніж приступати до створення Web-сторінок сайту, необхідно добре продумати матеріал, відповідно до нього вибрати організаційну структуру сайту в цілому, а потім переходити до проектування і розробки системи переходів між сторінками (системи навігації).

1.4.1 Технологія AJAX

Назва AJAX - це акронім, що розкривається як **Asynchronous JavaScript and XML** і означає **асинхронний JavaScript і XML.**

Простіше кажучи, можна вважати, що AJAX - це «JavaScript **з** розширеними правами», оскільки по своїй суті ця технологія являє собою сценарії на мові JavaScript, які в міру необхідності у фоновому режимі виконують запити до сервера й одержують додаткові дані, обновлюючи окремі частини сторінки, тим самим виключаючи необхідність її повторного завантаження цілком. На мал. 1 наочно показано, що відбувається, коли відвідувач запитує веб-сторінку, створену із застосуванням технології AJAX.

З точки зору перспективи AJAX має кращу збалансованість між функціональністю, реалізованою на стороні клієнта, і функціональністю, реалізованою на стороні сервера, при виконанні дій, викликаних користувачем. До появи AJAX функціональність клієнта й функціональність сервера

розглядалися як окремі частини, які працюють незалежно один від одного у відповідь на дії користувача. AJAX пропонує нове рішення - розподілити навантаження між клієнтом і сервером, дозволивши їм спілкуватися між собою, поки користувач працює із сторінкою.

Для того щоб пояснити вищесказане на простому прикладі, розглянемо веб-форму, куди користувач повинен внести деякі відомості (наприклад, ім'я, адреса електронної пошти, пароль, номер кредитної картки та інше). Перш ніж ця форма надійде в розпорядження вашого додатка, вона повинна бути перевірена. У випадку відмови від застосування AJAX у нашому розпорядженні є два способи перевірки. Перший з них полягає в тому, щоб дозволити користувачу заповнити форму й *відправити* її, після чого додаток перевірить правильність заповнення на стороні сервера. У цьому випадку користувач буде *гаяти час,* очікуючи завантаження нової сторінки. Інший варіант - виконати перевірку на стороні клієнта, але це не завжди можливо, якщо для цього буде потрібно передати клієнтові занадто великий обсяг даних (тільки уявіть собі, що необхідно перевірити правильність уведення назви міста відповідно до введеної назви країни).

Якщо ж застосовується AJAX, то веб-додаток може перевіряти дані, відправляючи запити серверу у фоновому режимі, у міру того як користувач уводить їх. Наприклад, після того як користувач вибере назву країни, веб-браузер може запросити в сервера список міст цієї країни, не відволікаючи користувача від його заняття.

Технологія AJAX слугує для створення більше гнучких і інтерактивних веб-додатків. Вона дозволяє виконувати асинхронні звертання до сервера, не перериваючи роботу користувача й непомітно для нього. **AJAX** - це інструмент, що може застосовуватися розробниками для створення веб-додатків, більш інтелектуально взаємодіючих з людиною.

Технології, з яких складається AJAX, уже реалізовані у всіх сучасних веб-браузерах. Таким чином, від клієнта не вимагається встановлювати які-небудь

додаткові модулі, щоб мати можливість взаємодії з веб-сайтами, побудованими на основі AJAX. До складу AJAX входять наступні компоненти:

- JavaScript - основний інгредієнт AJAX, що дозволяє реалізувати функціональність на стороні клієнта. У функціях JavaScript для маніпулювання окремими частинами сторінки HTML часто задіяна **об'єктна модель документа (Document Object Model - DOM).**
- Об'єкт **XMLHttpRequest** дозволяє JavaScript організувати асинхронний доступ до сервера, завдяки чому користувач має можливість продовжувати роботу зі сторінкою, у той час як **вона** виконує деякі дії. Під доступом до сервера маються на увазі прості запити HTTP на одержання файлів або сценаріїв, розміщених на сервері. Запити HTTP прості у виконанні й не викликають яких-небудь труднощів у випадку застосування брандмауерів.
- Серверні технології, які необхідні для обслуговування запитів, що надходять від JavaScript з боку клієнта. Однією з таки технологій є PHP.

Для організації взаємодії клієнт-сервер необхідно мати можливість передавати дані й розуміти, що за дані були передані. Передача даних - це найпростіше. Сценарій на стороні клієнта, що володіє доступом до сервера (за допомогою об'єкта XMLHttpRequest), може передавати серверу пари ім'я-значення за допомогою методів **GET** або **POST.** Ці дані легко можуть бути прочитані за допомогою будь-якого сценарію на стороні сервера.

Сценарій на стороні сервера просто відправляє свою відповідь за протоколом HTTP, але, на відміну від звичайного веб-сервера, відповідь повинна мати такий формат, що легко може бути розібрана кодом JavaScript на стороні клієнта. Для цього рекомендують формат XML, що має свої переваги, які полягають у тім, що, по-перше, він одержав широке поширення й, по-друге, існує велика кількість бібліотек, що полегшують роботу з XML документами. Але при бажанні можна вибрати будь-який інший формат (дані можуть передаватися навіть у вигляді простого тексту). Одна з відомих альтернатив XML - **JavaScript Object Notation (JSON** - подання об'єктів в JavaScript).

1.4.2 Розвиток AJAX

В AJAX немає жодного нового або революційного компонента, як можна було б подумати через галас, піднятий навколо цієї технології, - усі компоненти AJAX існують, принаймні, з 1998 року. Назва AJAX народилася в 2005 році в статті Джессі Джеймса Гаррета, (*http://www.adaptivepath.com/publications/essays/archives/000385.php)* і одержала популярність після того, як AJAX став застосовуватися компанією Google у багатьох власних додатках.

Зате з AJAX зв'язана та нова обставина, що вперше ринок акумулював досить енергії, щоб підтримати стандартизацію й сфокусувати цю енергію на подальшому розвитку. Як наслідок, було розроблено безліч бібліотек AJAX і з'явилося багато веб-сайтів з підтримкою AJAX. Завдяки проекту Atlas, Microsoft також сприяє розвитку AJAX.

Переваги та недоліки AJAX

Застосування технології AJAX для створення нових веб-додатків дає нам наступні переваги:

- Вона дозволяє створювати більш динамічні й більш якісні веб-сайти й веб-додатки.
- Висока популярність сприяє появі шаблонних рішень, які допоможуть розробникам не винаходити велосипед при рішенні найпоширеніших завдань.
- Дозволяє розробникам застосовувати напрацьовані навички.
- Функціональні можливості AJAX прекрасно інтегруються з функціональністю, надаваною веб-браузерами (наприклад навігацією по сторінці, приведенням розмірів сторінки до певних значень і т.д.).

Найбільш загальні випадки застосування AJAX:

- Перевірка правильності заповнення форми із залученням можливостей сервера, що дуже зручно, коли неможливо заздалегідь передати клієнтові всі дані, які можуть знадобитися в процесі перевірки.

- Розробка простих чатів, які не вимагають наявності зовнішніх бібліотек, таких як віртуальна Java-машина або Flash.
- Додавання функціональності, аналогічної підказкам Google
- Більш ефективне використання інших технологій.
- Створення динамічних таблиць даних, які на лету обновлюють бази даних на сервері.
- Розробка додатків, які вимагають відновлення інформації в режимі реального часу, зчитуючи її з різних зовнішніх джерел.

З AJAX зв'язані наступні потенційні труднощі:

- Оскільки адреса сторінки в процесі її роботи не змінюється, додати в закладки посилання на сторінку AJAX буде не так просто. У випадку додатків AJAX установка закладки має інше значення, що залежить від конкретного додатка, тобто звичайно потрібно зберегти поточний стан (уявіть собі, що працюєте зі звичайною програмою, посилання на яку не можна помістити в закладки).
- Пошукові системи можуть виявитися не в змозі проіндексувати всі частини вашого сайту, створеного на основі AJAX.
- Натискання на кнопку «Назад» у браузерах не приводить до того ж результату, як у класичних веб-додатках, оскільки всі дії користувач виконує в одній і тій же сторінці.
- На стороні клієнта JavaScript може бути відключений, що зробить додаток AJAX не функціональним, тому непогано було б передбачити на своєму сайті альтернативні варіанти сторінок, щоб не втратити потенційних клієнтів.

Ініціалізація запитів до сервера за допомогою об'єкта XMLHttpRequest.

Створивши об'єкт XMLHttpRequest, можна робити досить цікаві маніпуляції. Екземпляр цього об'єкта може бути створений різними способами, залежно від версії броузера, однак всі екземпляри XMLHttpRequest, як передбачається, мають ідентичний прикладний програмний інтерфейс (Application Programming Interface - API) і мають ідентичну функціональність.

(Насправді це не гарантується, оскільки кожний браузер має свою власну реалізацію.)

Таблиця 1.9 - Коротка довідка властивостей і методів об'єкта XMLHttpRequest:

Метод/властивість	**Опис**
abort()	зупиняє виконання поточного запиту
getAllResponseHeaders()	повертає заголовки відповідей у вигляді рядка
getResponseHeader("headerLabel")	повертає заголовок однієї відповіді у вигляді рядка
open("metbod", "URL"[, async-Flag[,"userName"[, "password"]]])	ініціалізує параметри запиту.
send(content)	виконує запит **HTTP**
setRequestHeader("label","value")	призначає пари label/value у заголовку запросу
Onreadystatechange	використається для установки функції зворотного виклику, що буде обслуговувати зміну стану запиту
readyState	повертає стан запиту: 0 = не ініціалізовано 1 = іде відправлення запиту 2 = запит відправлений 3 = іде обмін 4 = завершений
responseText	повертає відповідь сервера у вигляді рядка.
responseXML	повертає відповідь сервера у вигляді документа XML
Status	повертає код стану запиту
statusText	повертає повідомлення про стан запиту

При кожному звертанні до сервера беруть участь методи open і send. Метод open задає конфігурацію запиту, визначаючи його параметри, метод send відправляє запит (забезпечує доступ до сервера). Якщо запит виконується в асинхронному режимі, то перед викликом методу send треба також визначити властивість onreadystatechange, вказавши функцію зворотного виклику, що буде виконана при зміні стану запиту, забезпечивши, таким чином, підтримку механізму AJAX.

Метод open призначений для ініціалізації запиту. Йому передаються два обов'язкових і кілька необов'язкових аргументів. Метод open не створює з'єднання із сервером, він лише визначає параметри з'єднання. Перший аргумент вказує метод, що буде застосують для передачі даних серверу, і може приймати значення GET, POSTабо PUT. Другий аргумент URL, він визначає адресу, по якій буде відправлений запит. URL може бути як повним, так і відносним. Якщо URL говорить про те, що доступ до ресурсу буде здійснений не через HTTP, то значення першого аргументу ігнорується.

Третій аргумент, async, визначає режим виконання запиту - синхронний або асинхронний. Якщо він дорівнює true, то після виклику методу send() керування негайно буде повернуто сценарію, не чекаючи одержання відповіді від сервера, а якщо false, те метод send будеочікувати відповіді від сервера, перш ніж повернути керування, припиняючи роботу веб-сторінки. Щоб установити режим асинхронної роботи, необхідно передати в аргументі async значення true і встановити оброблювач події onreadystatechange, що буде займатися обробкою відповіді сервера.

У випадку застосування методу GET параметри передаються серверу в рядку URL запиту приблизно так: *http://localhost/ajax/test.php?param1=x&param2=y*. У даному прикладі серверу передаються два параметри: перший з ім'ям paraml і значенням x і другий з ім'ям param2 і значенням y.

// звернутися до сервера, щоб виконати операцію на стороні сервера

```
xmlHttp.open ("GET",  "http://localhost/ajax/test.php?param1=x&param2=y", true);
xmlHttp.onreadystatechange = handleRequestStateChange;
xmlHttp.send(null);
```

Якщо параметри передаються методом POST, то вони не приєднуються до рядка URL запиту, а передаються методу send() у вигляді рядка:

```
// звернутися до сервера, щоб виконати операцію на стороні сервера
xmlHttp.open ("POST",  "http://localhost/ajax/test.php?param1=x&param2=y", true);
xmlHttp.onreadystatechange = handleRequestStateChange;
xmlHttp.send("param1=x&param2=y");
```

Ці два уривки коду виконують ідентичні дії. На практиці метод GET найчастіше застосовується для налагодження, оскільки його не тяжко емулювати за допомогою веб-браузера й побачити своїми очима що саме повертає сценарій, що виконується на стороні сервера. Метод POST необхідний, коли обсяг переданих даних перевищує 512 байт, оскільки цей обсяг не можна відправити методом GET.

Яку технологію обрати?

Сучасний WEB-портал – це обов’язково динамічна, а не статична інформація. Статичні html-сторінки майже стовідсотково відійшли в минуле, на долю незмінної інформації залишилися лише медіа-документи.

Для побудови динамічного сайту розробнику надається багато засобів, різних за складністю і можливостям, ефективністю роботи і простотою підтримки. Тому, починаючи розробку чергового WEB-проекту, часто постає запитання – яку технологію обрати?

Таблиця 1.10 - Технології для створення WEB-порталів

Хар-ка	PHP	ASP.NET	Ruby	Perl (CGI)	Python	Cold Fusion	Java сервлети	JSP
Безкоштовність	+	-	+	+	+	+-	+	+
Мультиплатформенність	+	-	+	+	+	+	+	+
тип виконання - інтерпретатор	+	-	+	+	+	+	+-	+
об'єктно орієнтованість	+	+	+	+	+	-	+	+
перевантаження функцій	-	+	+	-	+	-	+	+
простір імен	+	+	+	+	+	-	+	+
Розширюваний	+	+	+	+	+	+	-	+
Розвивається	+	+	+	+-	+	+	+	+
зрозумілий код	+	+	-	-	-	+-	+	+

У цьому розділі були розглянуті основні етапи та засоби їх реалізації при розробці порталу. Проаналізувавши розглянуті технології за мову веб-програмування взята Java, бо:

1) знаходиться на більшості хостингів;

2) легка у використанні;

3) легко встроюється в html-код;

4) досить розповсюджена у веб-програмуванні.

Невід’ємною частиною кожного сайту, в нашому випадку порталу, є база даних. При виборі бази даних потрібно чітко знати які функції вона має виконувати, тому перевагу віддано MySQL.

Вибрана БД :

1) має зручні засоби управління;

2) використовується на більшості хостингів;

3) підтримується всіма основними ОС;

4) є безплатним продуктом.

Для розроблюваного порталу вибрана ієрархічна структура побудови інформаційних блоків.

Коли вибрані основні моменти, можна приступати до розробки самого порталу

2. ПОРТАЛ QTEXS

2.1. Реалізація комерційних web-проектів

Портал Qtexs, спочатку називався KuGo це ініціали замовників. Проте основною проблемою у продовженні розробки цього порталу була відсутність мотивації. Коло осіб, що розробляли цей проект то збільшувалось то зменшувалось. В період з 1.09.2014-10.01.2014 був застій проекту, що зумовлювалось нестачею досвіду та практики.

Першим нашим проривом було проектування Landing Page та купівля домена qtexs.com на термін у 2 роки, що обійшлося нам на той час 200грн.

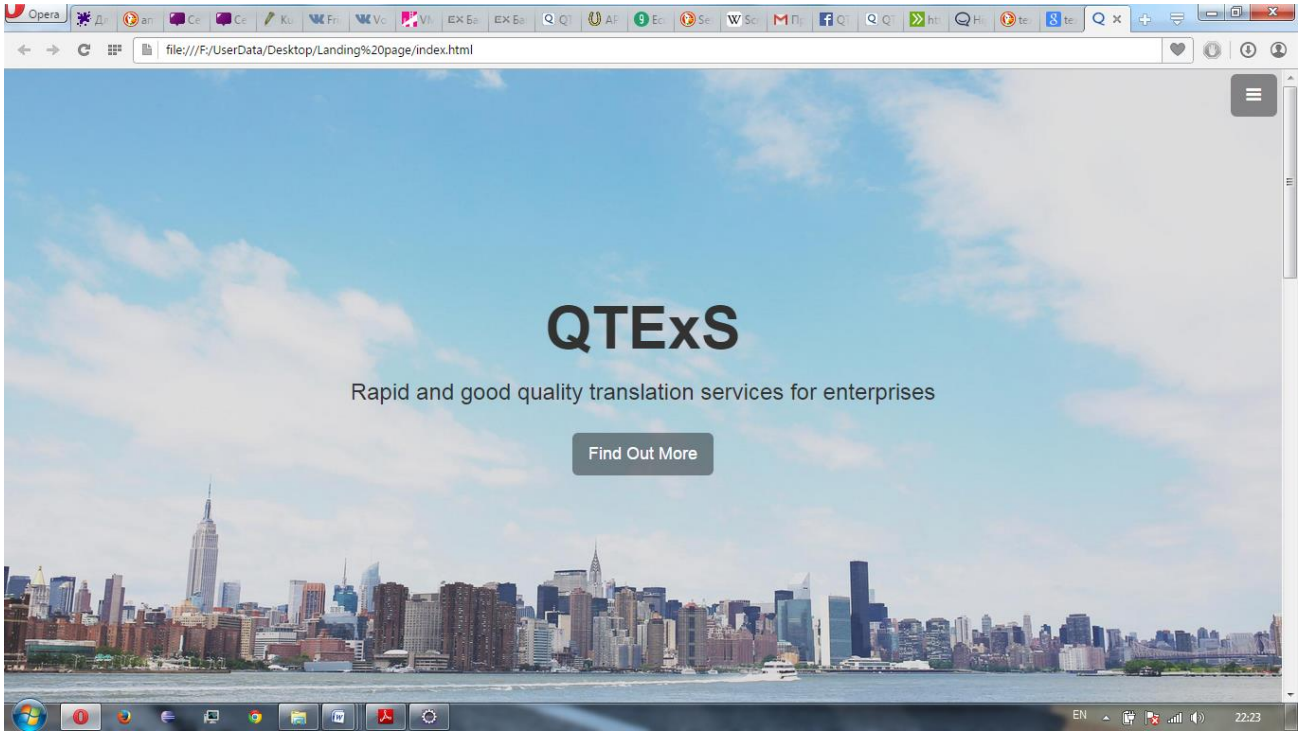

Рисунок 7- Перша інформаційна сторінка Landing Page для представлення аудиторії[2]

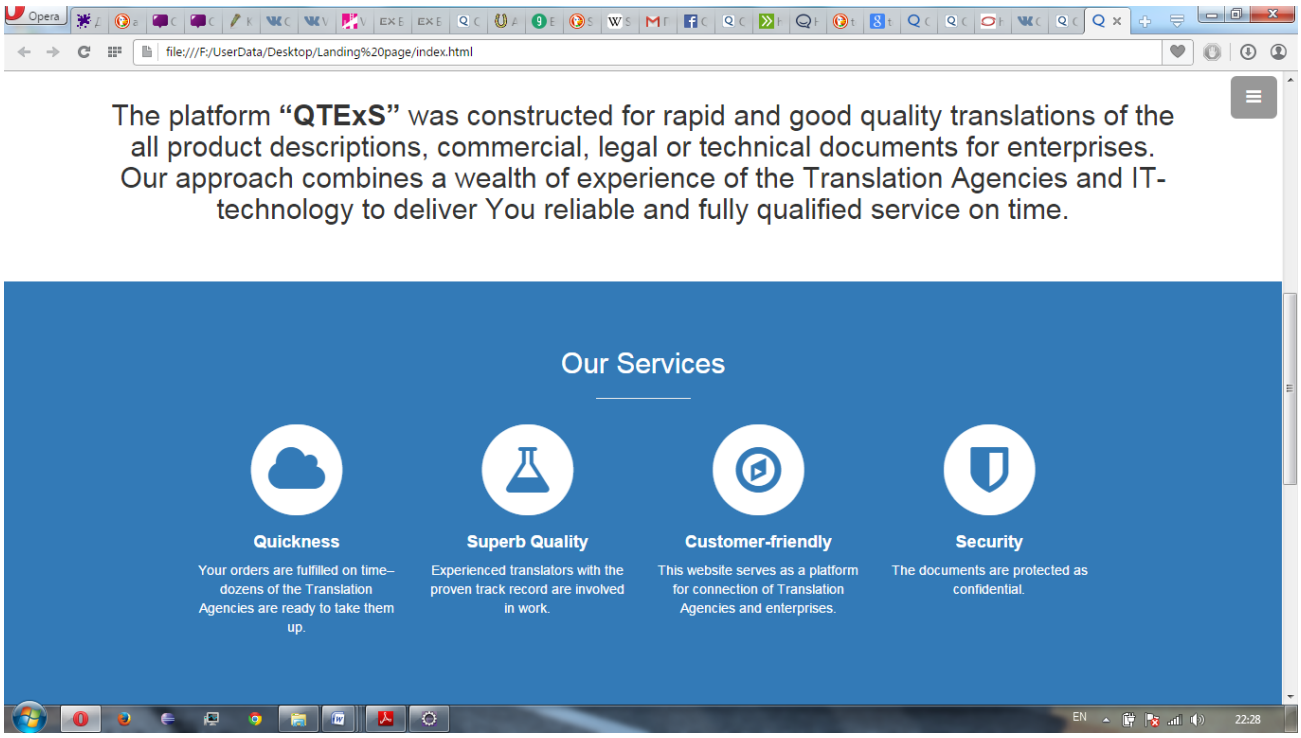

Рисунок 8- Друга інформаційна сторінка Landing Page

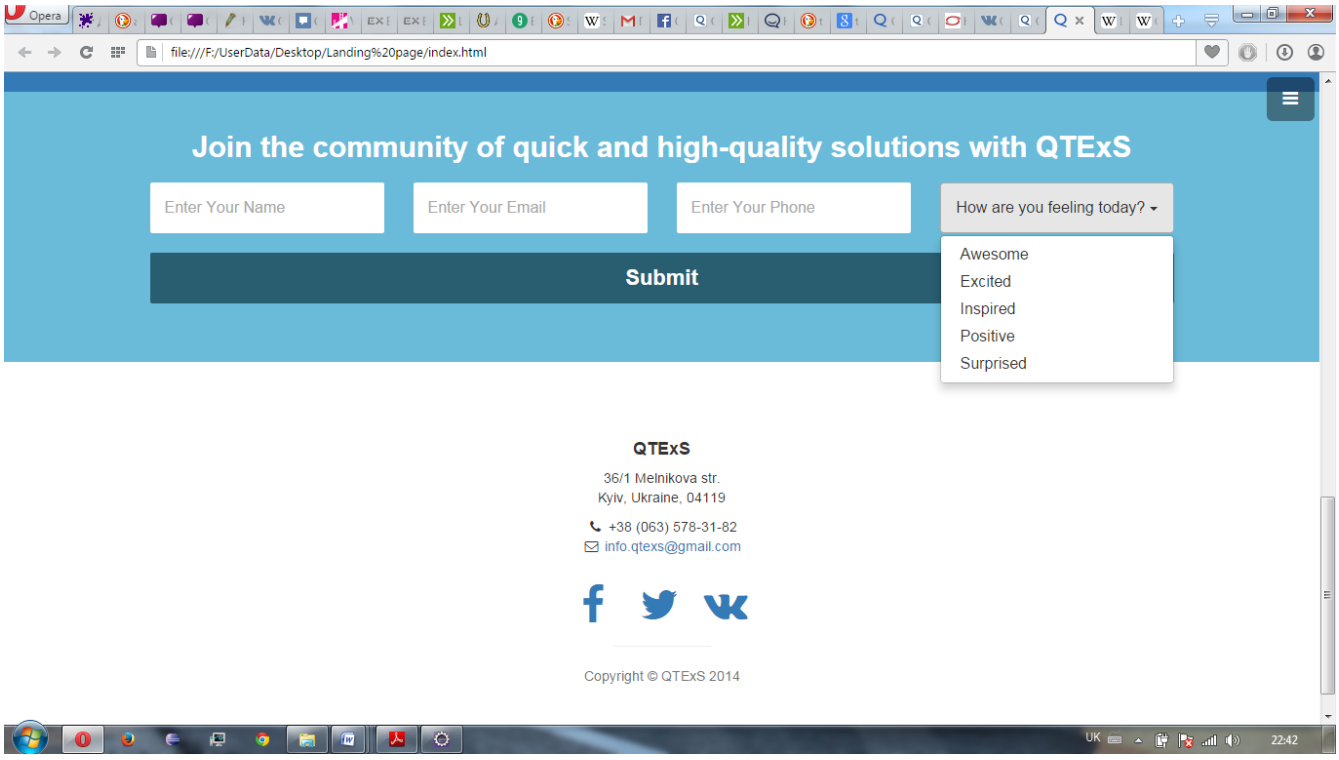

Рисунок 9 – 3-тя інформаційна сторінка, написаний php скріпт

На третьому листі був написаний php скріпт, що відправляє на е-мейл info.qtexs@gmail.com контакну інформацію перекладачів, що готові з нами

співпрацювати. Метою створення цієї інформаційної сторінки було набивання бази клієнтів та створення співтовариства QTExS.

Друга версія: Landing page дата доступа: 11.05.2015р.

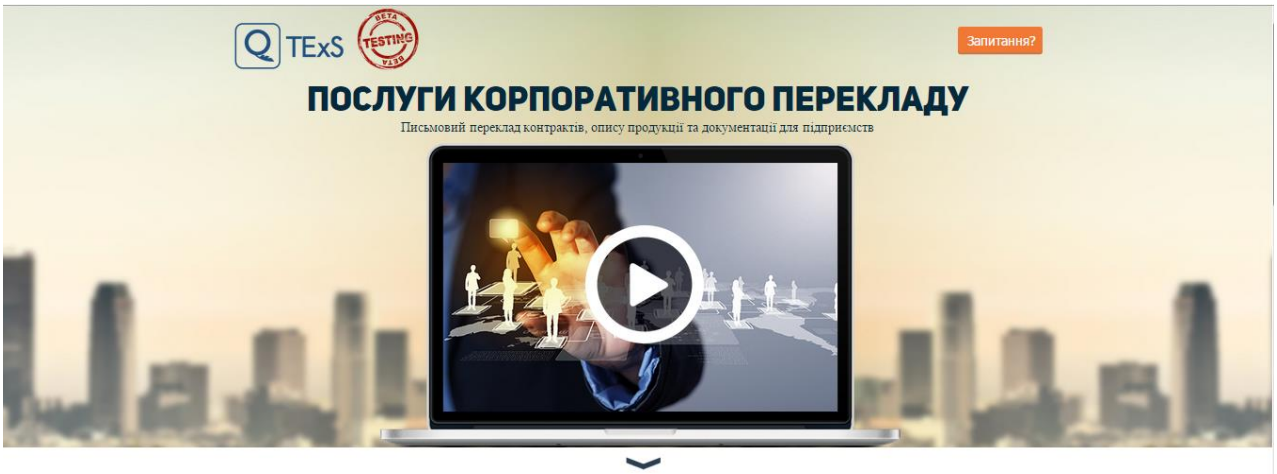

Рисунок 10 - Landing page Version 2.0.

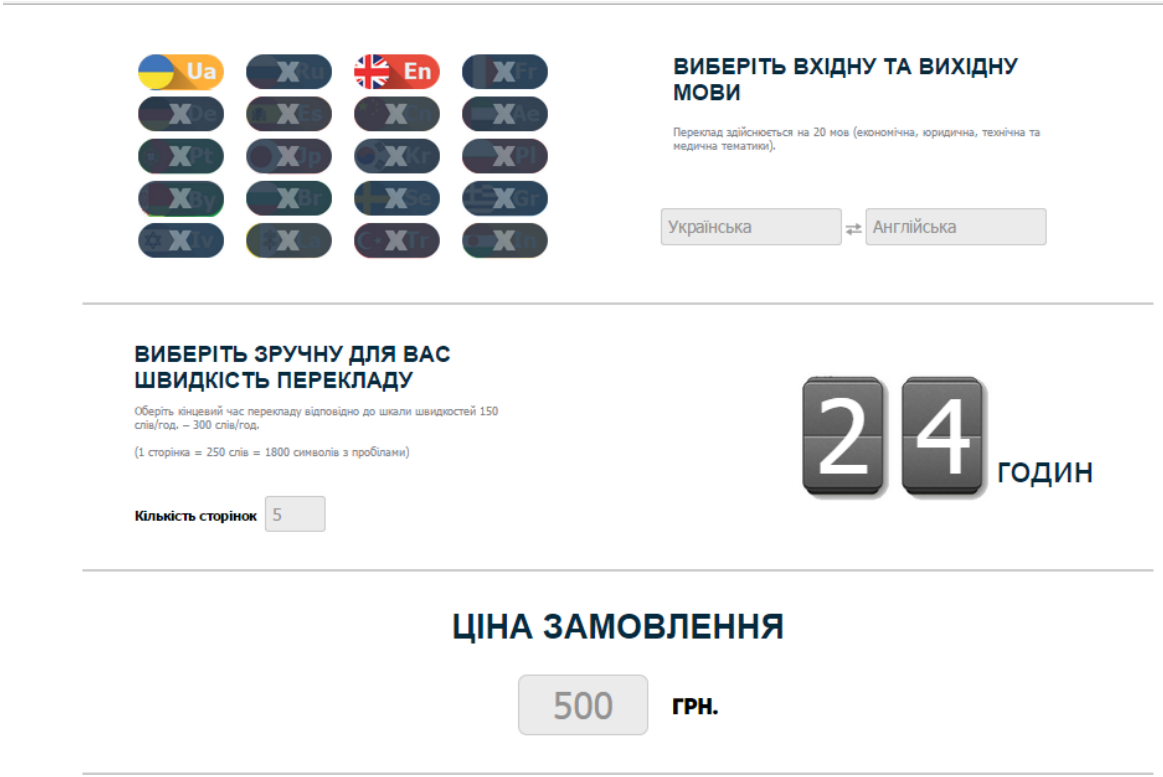

Рисунок 11 - Landing page Version 2.0

Обчислення цін та швидкісних характеристик перекладачів. Для написання використовувались такі технології, як scss (смас), що дозволяє легко масштабувати продукт на різні технології

2.2. Медіа маркетинг порталу

За допомогою медіа порталів таких як vk.com, facebook.com, twiter.com було створено групи, для того, щоб користувачі, майбутні співробітники могли слідкувати за нашими новинами. Щоб перейти на ці сторінки достатньо натиснути на ярлики внизу Landing page.

VK[21], Facebook[22],Twiter[23]

Завдяки цьому медіа маркетингу ми залучаємо потенційних перекладачів та клієнтів у майбутньому. Також збільшуємо інтерес до нашого творіння.

2.3. Розвиток QTExS

Під час входу на ринок тестується бізнес – модель при 0% комісії. Після залучення до платформи 20 компаній, вводиться комісія у розмірі 5% від ціни замовлення, що буде стягуватися із команд перекладачів. Перелік методів маркетингу: SEO, прямі продажі, залучення посольств, контекстна реклама

Ядро перекладацького порталу версії 1.0 написане на мові програмування Java. В якості тестування взятий сервер Apache TomCat 7.0. Структура проекту виглядає приблизно так:

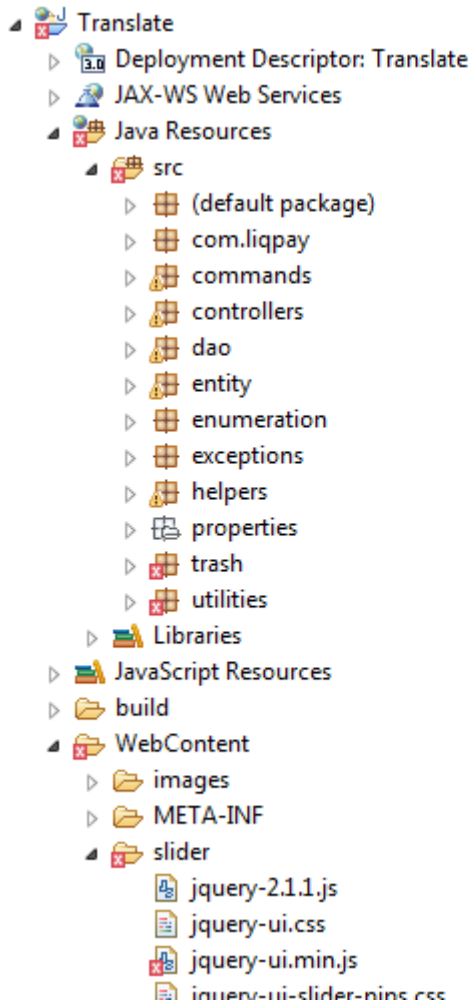

Рисунок 12 – Внутрішня структура проекту

Структурні елементи:

- Java Resources
- Deployment Descriptor
- Web Content

2.4. Java Resources

2.4.1. LiqPay

В цій теці знаходиться джерела Java коду, та пакети даних. Кожний пакет виконує ряд функцій. За Code Convention Java, пакети називаються маленькими літерами, та з кінця на початок DNSім’ям (адресу сайту в браузері). Уніфікація правил для написання програм через patterns (патерни), спрощує процес

об’єднання версій продукту написаний декількома людьми, та впровадження його на ринок інтелектуальної праці (world wide web).

Першим пакетом, є com.liqpay. Його можна скачати в офіціальному дестрибутиві GitHub[24]. SDKдля розробки написаний декількома мовами, а саме[25]:

- Ruby
- Python
- Java
- Erlang
- Php
- Haskell
- Nodejs
- Perl
- IOS
- AOS

Таку систему оплати використовує Укрзалізниця для оплати білетів в режимі онлайн[26]. Вона надає безпечний доступ оплати білетів. Супроводжується такими захисними елементами як SMSаутенфікація, SSLугода з підтримкою SSH та RSA шифруванням.

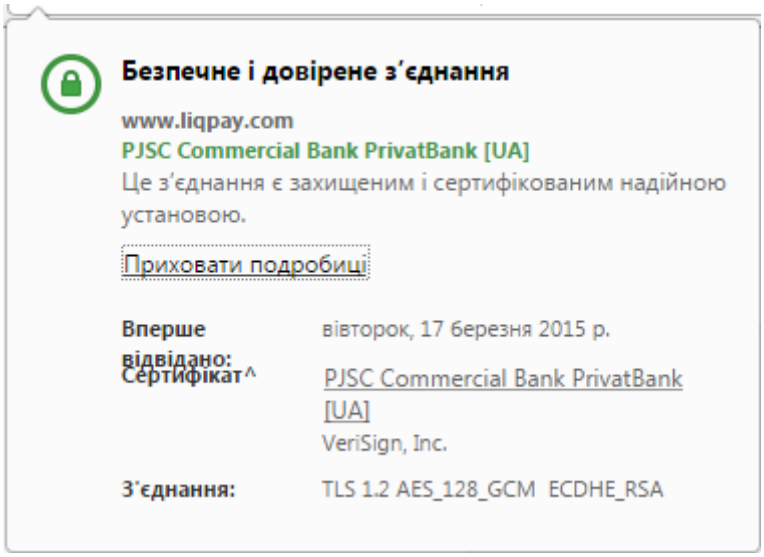

Рисунок 13 – Тип з’єднання

В цій програмі використано відкритий API цієї компанії. Що забезпечить надійну оплату, та відповідний захист системи.

Liq&Buy 3.0

Прийом платежів на сайті ***client***->***server***

Для початку прийому платежів на Вашому сайті необхідно:

- Зареєструватися на www.liqpay.com
- Створити магазин у Вашому акаунті використовуючи майстер-установки
- Отримати готову HTML-кнопку для оплати або створити просту HTML форму

HTML форму необхідно відправити методом *POST* на URL *https://www.liqpay.com/api/checkout* з двома параметрами *data* і *signature*, де: *data* –результат функції *base64_encode ($json_string)signature* - результат функції *base64_encode(sha1($private_key . $data . $private_key))*

Таблиця 2.1 - JSON параметрів

JSON параметри	Значення
Version	**Required** Версія API. значення – *3*
public_key	**Required** Публічний ключ - ідентифікатор магазина. Отримати ключ можна в налаштуваннях магазина
Amount	**Required** сума платежу. *Наприклад: 5, 7.34*
Currency	**Required** Валюта платежу. *Можливі значення: USD, EUR, RUB, UAH*

Продовження таблиці 2.1.

Descr iption	**Required** Опис платежу.
order_id	**Required** Унікальний ID покупки у Вашому магазині. Максимальна довжина **255** символів.
Recurringbytoken	**default *0*** Этот параметр позволяет генерировать *card_token*плательщика, который вы получите в callback запросе на*server_url*. *card_token* позволяет проводить платежи в offline используя метод *payment/paytoken*. Услуга активируется через менеджера LiqPay. Можливі значення: *1*
Type	**default – buy** Тип платежу. Можливі значення: *buy* - покупка *donate* - благодійність Якщо тип платежу *donate* платник зможе змінити суму платежу.
Subscribe	**Optional** Регулярний платіж. Можливі значення: *1*
subscribe_date_start	**required если subscribe=1** якщо. Час необхідно вказувати в наступному форматі*2015-03-31 00:00:00* по *UTC*
subscribe_periodicity	**required если subscribe=1** Періодичність списання коштів. Можливі значення: *month* - раз на місяць

Продовження таблиці 2.1.

	year - раз на рік
server_url	**Optional** URL API у Вашому магазині для повідомлень про зміну статусу платежу (*сервер* -> *сервер*). Максимальна довжина **510** символів. Докладніше
result_url	**Optional** URL у Вашому магазині на який покупець буде переадресовано після завершення покупки. Максимальна довжина **510** символів.
pay_way	**default *card,liqpay,delayed,invoice,privat24*** Спосіб яким клієнт зможе здійснити оплату. Можливі значення: *card* - Карта *liqpay* - Кошелек LiqPay *delayed* - ТСО (термінал) *invoice* - Инвойс *privat24* - Приват24
Language	**Optional** Мова платіжної сторінки. Можливі значення: *ru*, *en*
Sandbox	**Optional** Включає тестовий режим для розробників. Гроші на карту не зараховуються. Щоб включити тестовий режим, необхідно передати значення *1* . Всі тестові платежі будуть мати статус *sandbox* - успішний тестовий платіж.

2.4.2. Концепція MVC

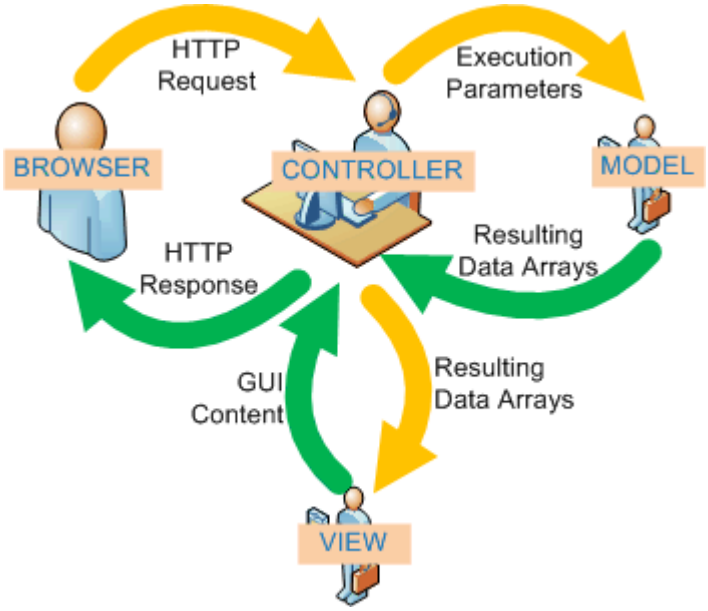

Рисунок 14 - MVC – Model View Controller модель вигляд контролер

Дана концепція JavaEE демонструє взаємозв’язок бази даних з виглядом webсторінки та способами імплементації даних в веб площину. В цій концепції продемонстровано використання декількох патернів програмування:

- Command
- DAO
- Factory

Вони покращують читабельність, масштабованість, пере використання та швидкість розробки якісного програмного забезпечення та інтеграцію його в виробництво чи малий бізнес.

2.4.2.1. Model

Йдемо знизу вверх. База даних складається з релятивійских таблиць. Ці таблиці відображуються в javakоді сутностями (enteties), які є моделлю даних, що ми будемо зберігати чи використовувати. Щоб створити сутність потрібно використати мета данні мови джава (аннотації), приклад наведено нижче

```
@Entity
@Table(name = "CUSTOMERS")
public class Customer {

    public interface WithPasswordView {};
    public interface WithoutPasswordView extends WithPasswordView {};

    @Id
    @Column(name = "ID")
    @GeneratedValue
    @JsonView(WithPasswordView.class)
    private Integer id;

    @Column(name = "FIRSTNAME")
    @JsonView(WithPasswordView.class)
    private String firstname;

    @Column(name = "LASTNAME")
    @JsonView(WithPasswordView.class)
```

Рисунок 15 – Приклад сутностей

В крайньому випадку, коли проект буде запускатись на сервері він має унаслідуватись від Serializable, щоб в разі чого записатись на HDD.

DAO- data access object

Це об'єкт що надає абстрактний інтерфейс до деяких видів баз даних чи механізмів персистентності реалізуючи певні операції без розкриття деталей бази даних. Він надає відображення від програмних викликів до рівня персистентності. Така ізоляція розділює запити до даних в термінах предметної області та їх реалізацію засобами СКБД.

Цей паттерн проектування можна застосовувати до більшості мов програмування, видів програмного забезпечення з потребою персистентності та більшості типів баз даних, але він традиційно асоціюється з застосунками Java EE та реляційними БД доступ до яких здійснюють через JDBC API що пов'язано з походженням паттерна із збірки найкращих практик Sun Microsystems.[27] ("Core J2EE Patterns") для цієї платформи.

DAOFactory - це об’єкт, що призначений тільки для створення екземплярів DAO.

2.4.2.2. Controller

ClassController–по суті це є JavaServlet [28], на який приходять запити. Він є основним і ключовим елементом нашої системи в залежності від того, що прийде на вхід. В особливості параметр commandв GETчи POSTзапиті, така команда і буде виконуватись нашою серверною частиною.

В той час команда повертає адрес сторінки на яку буде переадресовано клієнта, або частину сторінки, якщо ми використовуємо технологію AJAXв конкретній частині портала.

Команда дозволяє інкапсулювати всю інформацію, необхідну для виконання певних операцій, які можуть бути виконані пізніше, використавши об’єкт команди

Приклад[29]:

Ваш бос дуже вимогливий — він ніколи не переймається тим, як буде робитися робота і не особливо переймається тим, хто її буде робити – йому головне щоб вона була зроблена, як тільки замовник дасть добро. Проте вашому босу ніхто не заважає назначати людей, які будуть працювати над виконанням завдання. Він вирішив, що ви, оскільки ви на високих позиціях у компанії, ідеально підходите для того, щоб зібрати бригаду, отримати список вимог від замовника і бути готовими запустити роботу, як тільки замовник підпише контракт.

Замовник має зв’язок із вами і може попросити вас виконати всю необхідну роботу як тільки запуститься процес після підписання контракту. Замовник — це ваш запускач (invoker), він знає як попросити вас виконати роботу тоді, коли це йому зручно.

2.4.2.3. View

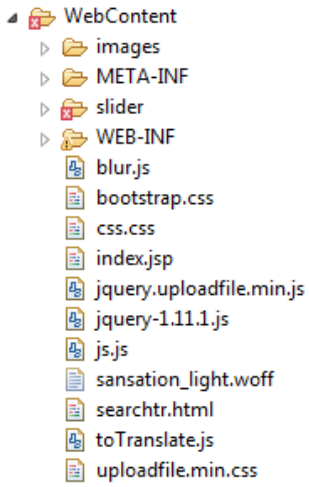

Рисунок 16 – Організація теки клієнтської частини

За вигляд відповідає тека WebContent. Вона має такі структурні частини:

- Основна частина відкрита для доступу ззовні
- META-INF
- WEB-INF

В основній частині зберігаються картинки, індексовані сторінки, джава скріпти та інший виконуваний код, що може бути загальнодоступним для усіх клієнтів. Першою завантажуванню частиною зазвичай називають індекс сторінки (index.html,.jsp…), хоча це може бути змінено в налаштування самого сервера.

WEB-INF– ця тека є закритою і доступ до неї можливий тільки з самого сервера. Тут можна зберігати бібліотеки, що використовуються порталом. Також сторінки, які повинні відображатись тільки після здійснення відповідних дій, або важливість їх є значною і доступ повинен надаватись тільки після аутенфікації користувача.

Такий підхід забезпечить продуманий порядок дій, та збільшить надійність системи в цілому. Він не дозволить користувачу руйнувати логіку виконання програми.

В даній реалізації в цій теці зберігаються тимчасові файли при загрузці на сервер.

Одним з найважливіших файлів при запуску TomCat це web.xml, в якому зберігаються важливі налаштування такі як перша сторінка, що буде представлена користувачу при першому запуску, час відклику і багато інших…

META-INF – Я не знаю для чого він, може ви знаєте ☺

2.5. Deployment Descriptor

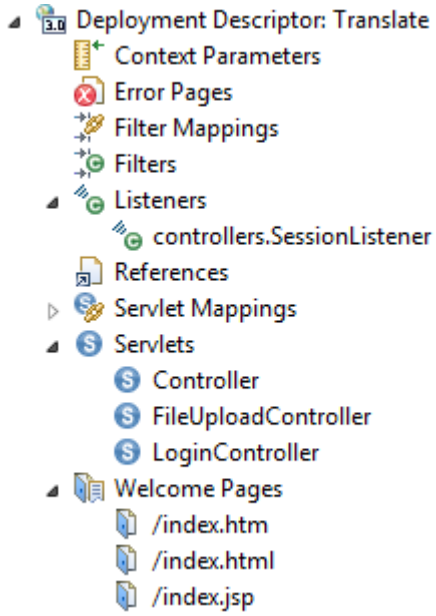

Рисунок 17 – Дескриптор розсортування

В цій структурній частині можна глянути як зв’язується наше застосування. Основні його частини. Візьмемо до прикладу Listeners:

```
@WebListener
publicclass SessionListener implements  HttpSessionListener  {

	privatestaticfinal Logger console =Logger.getLogger(Controller.class.getName());


	@Override
	publicvoid sessionCreated(HttpSessionEvent sessionEvent) {
		System.out.println("Session Created::
ID="+sessionEvent.getSession().getId());

	}

	@Override
	publicvoid sessionDestroyed(HttpSessionEvent sessionEvent) {
		System.out.println("Session Destroyed::
ID="+sessionEvent.getSession().getId());
		Order order=(Order)sessionEvent.getSession().getAttribute("order");
		if(order!=null){
			order.deleteTempFiles();
```

```
        }
    }
```

Тут при знищенні сесії видаляться саме замовлення з підтримуючими до нього файлами. При великому проекті розібратись у ньому буває вкрай важко, тому це досить необхідна річ. Вона спрощує розуміння логіки загального виконання програми.

2.6. Інтерфейс користувача

Значні ресурси було вкладено в роботу над інтерфейсом користувача, зокрема в розробку дизайну, адже привабливий дизайн одразу налаштовує на позитивний лад роботи з порталом. Нижче наведено основні сторінки, з якими працює користувач.

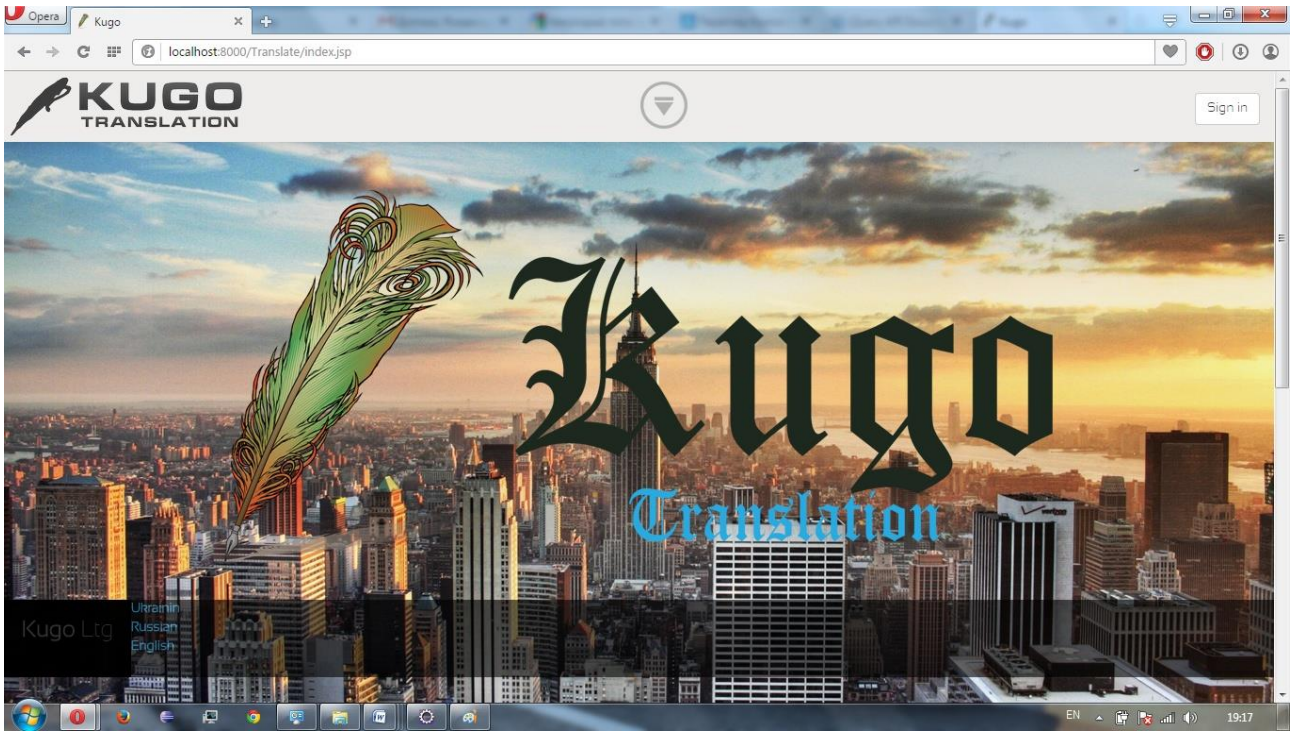

Рисунок 18

Перша версія продукту. Центральна кнопка enject обертається, при натисканні кнопки. В цей момент нижній футер відкривається. Він складається з двох частин «Kugo Ltg» та мов для відображення сторінки.

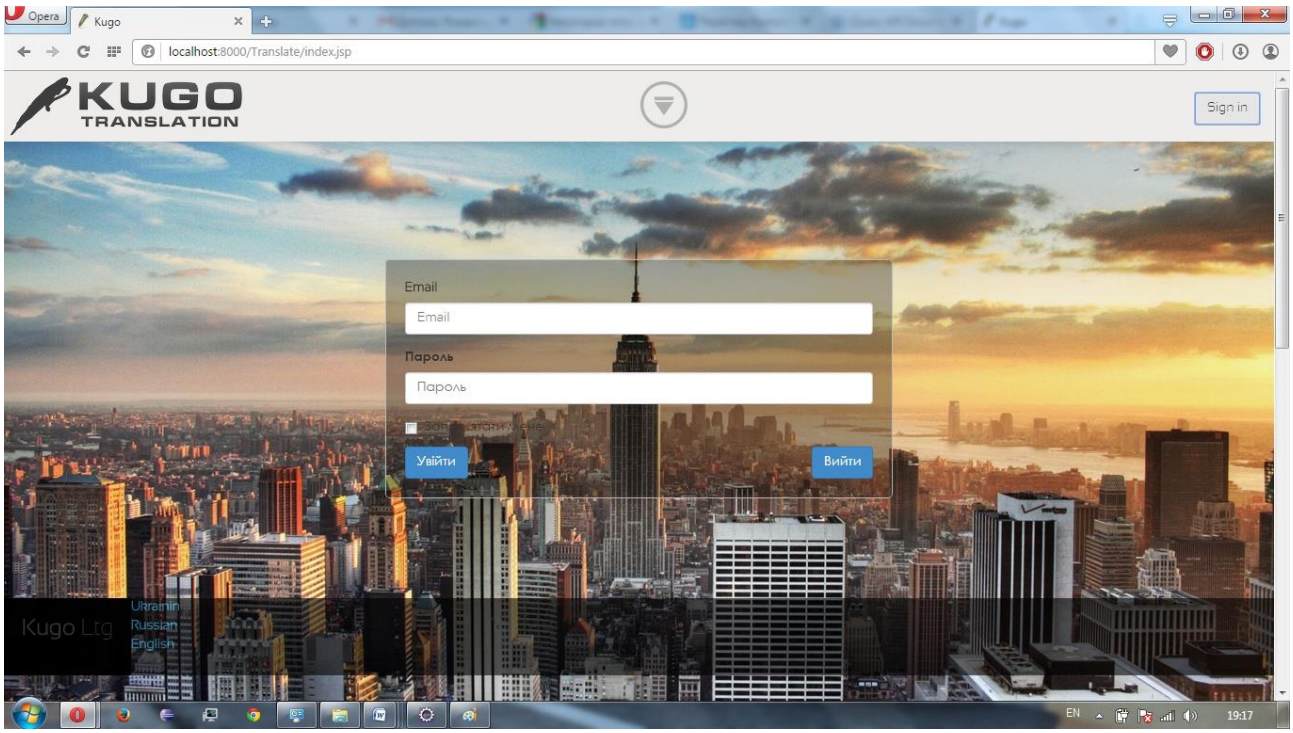

Рисунок 19

Елемент переходу на вхід в систему[30].

Натискаючи поза формою повертаємось в першопочатковий стан

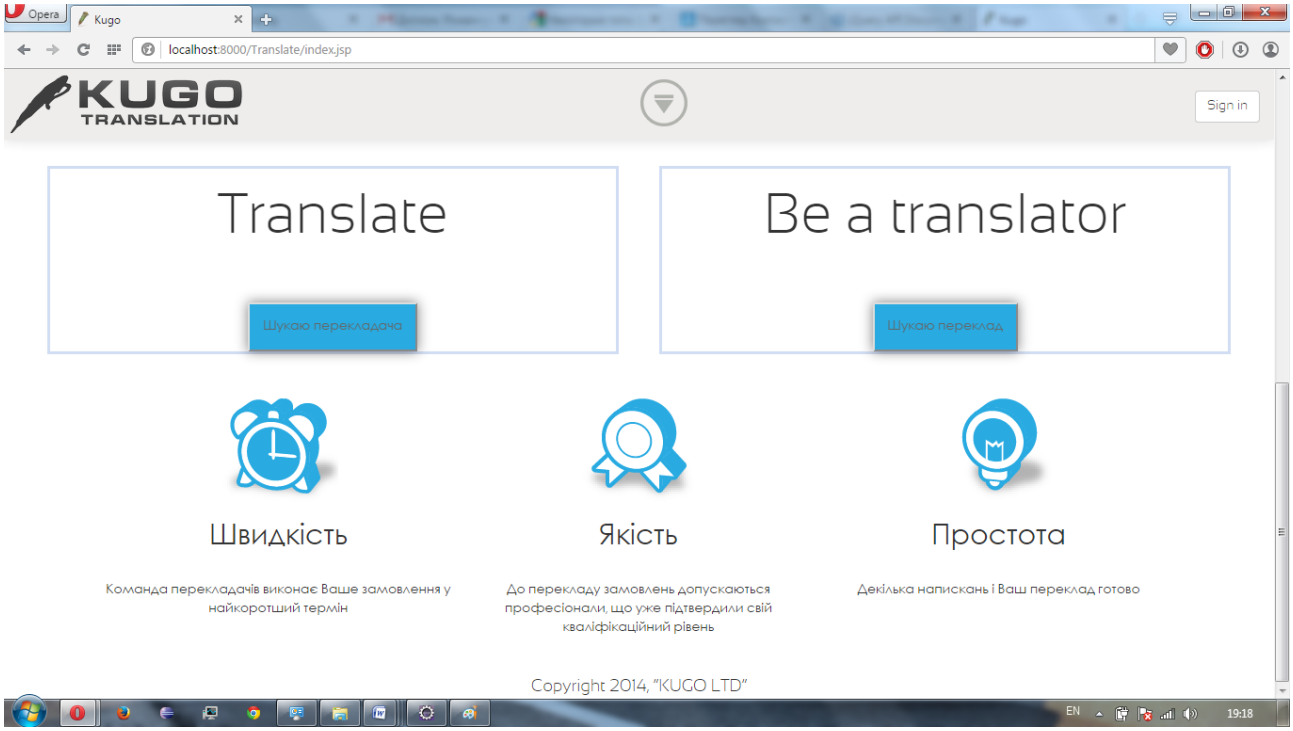

Рисунок 20

Нижня інформаційна сторінка, для пошуку та перекладачів, та самого перекладу. Також переваги у порівнянні з іншими продуктами.

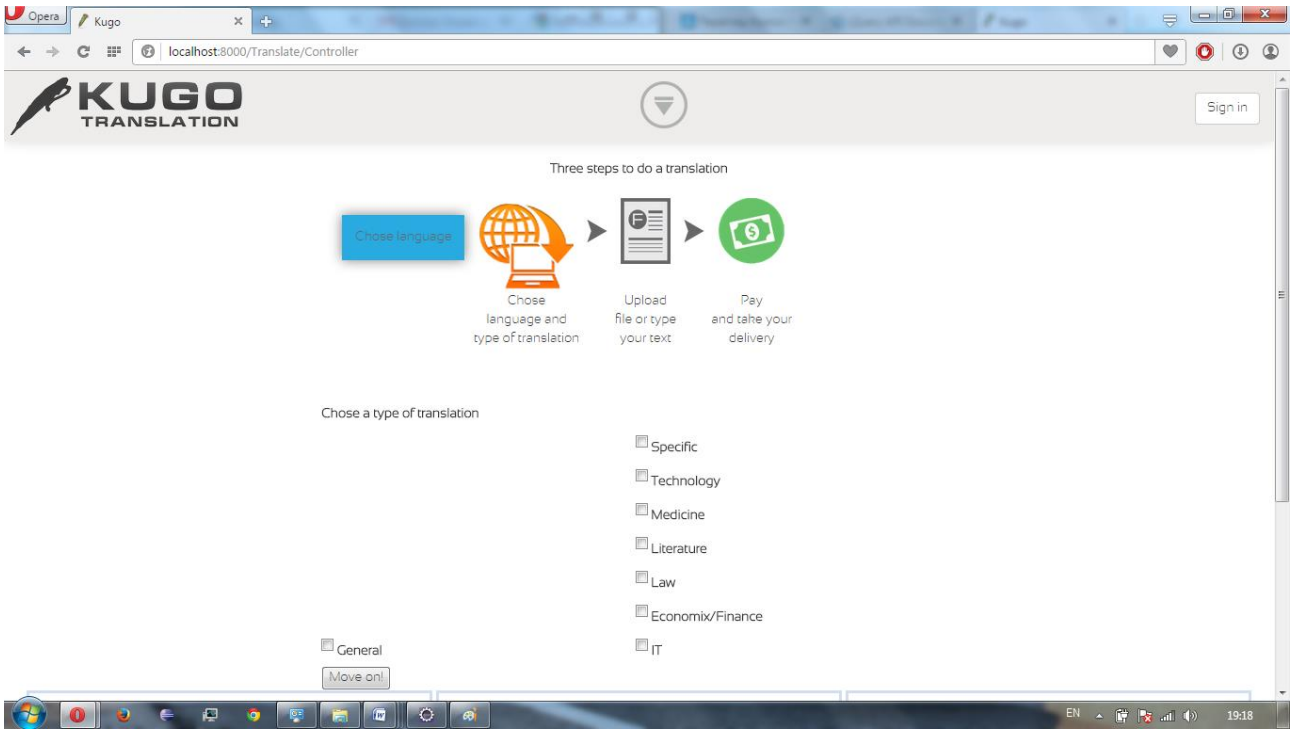

Рисунок 21

Перші моделі, та зосередження на цільовій аудиторії та типу надавання перекладу.

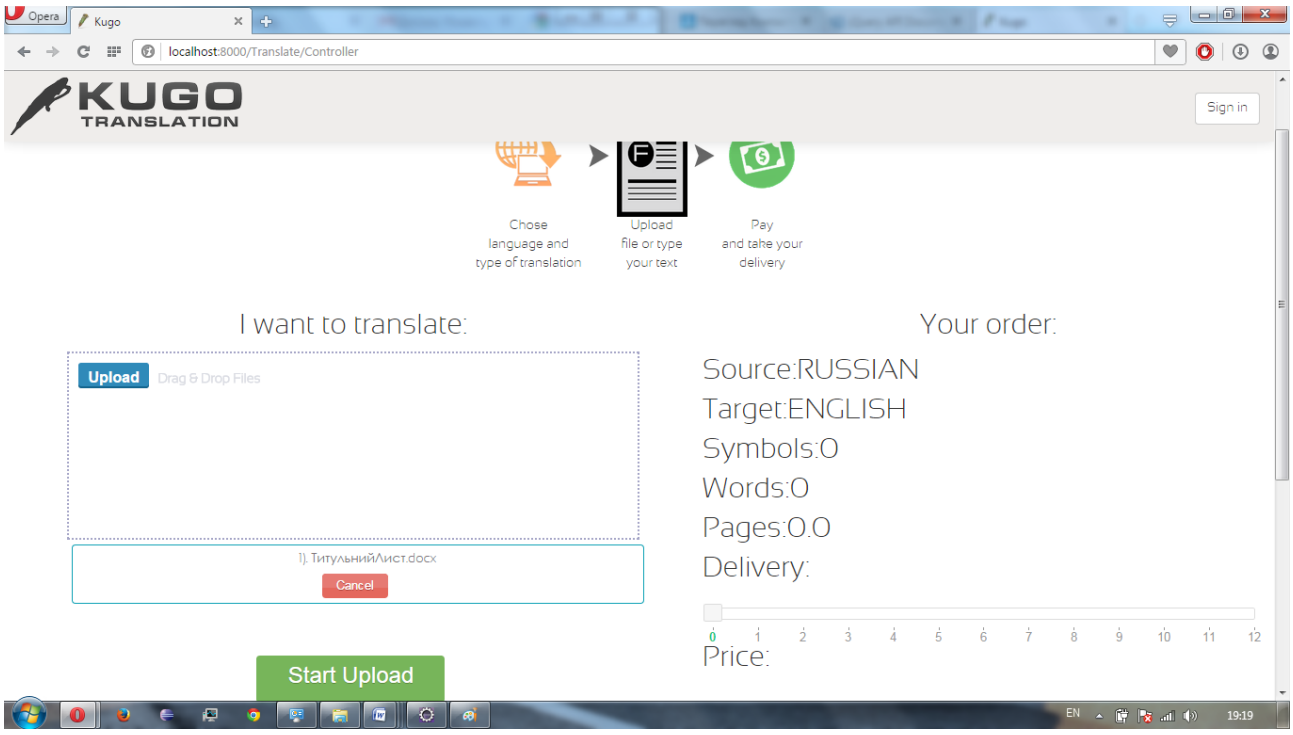

Рисунок 22

Заватаження файла за допомогою jQuery модуля[31]

Обчислення символів та слів.

Для txt файлів потрібно ще визначити кодування, тому що воно може бути utf-8, ANCII … та багатьох інших типів в залежності від локалізації. Тому потрібно було застосувати програмну бібліотеку com.ibm.icu.text.CharsetDetector;[32]

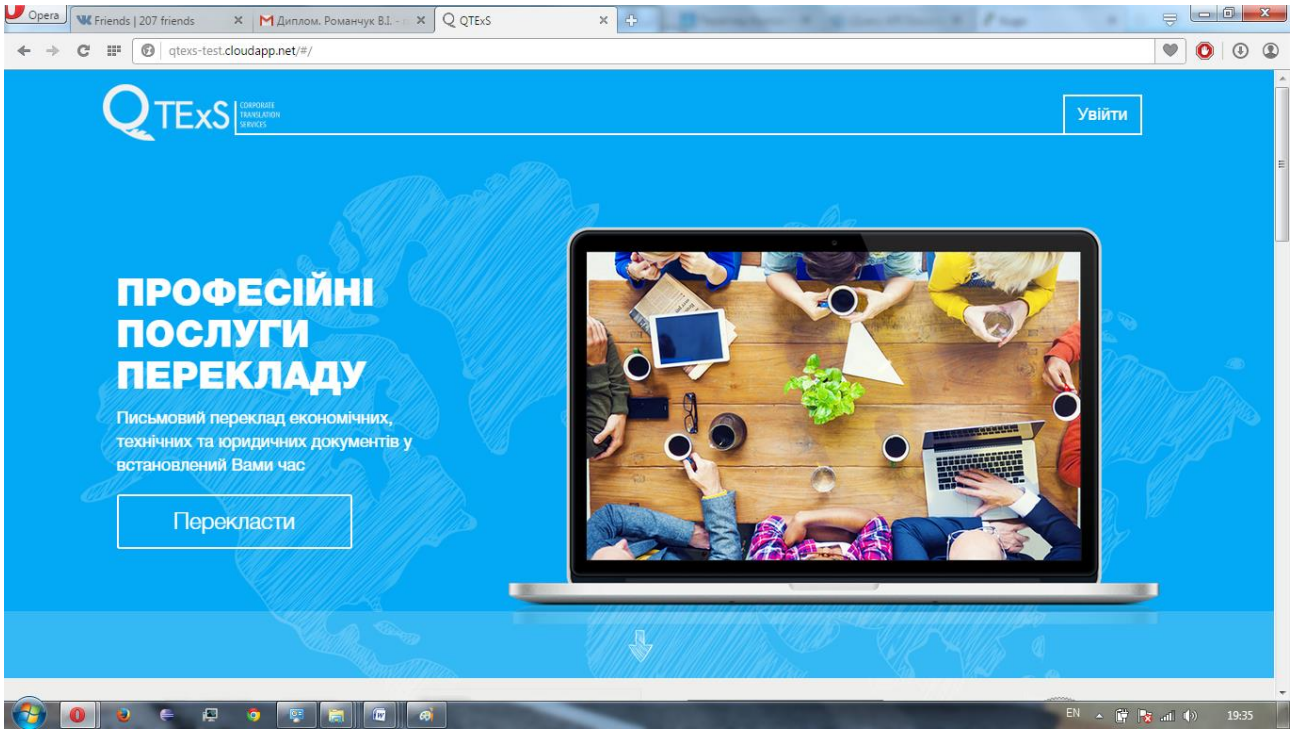

Рисунок 23

Версія 2.0 більш дружня та в правильних тонах, спрощений інтерфейс, прийнято рішення не акцентувати увагу на прийомі нових перекладачів, що може відлякати потенційних клієнтів.

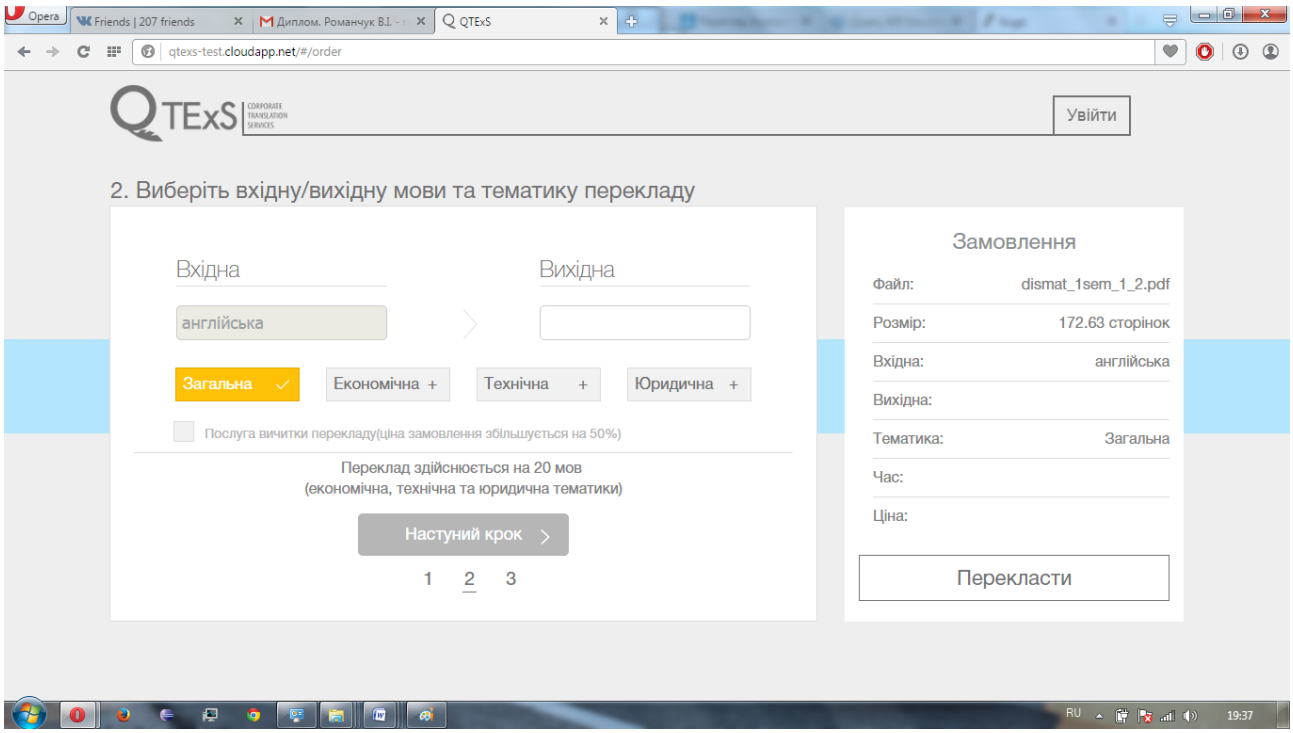

Рисунок 24

4 типи перекладу, та кроки оформлення замовлення

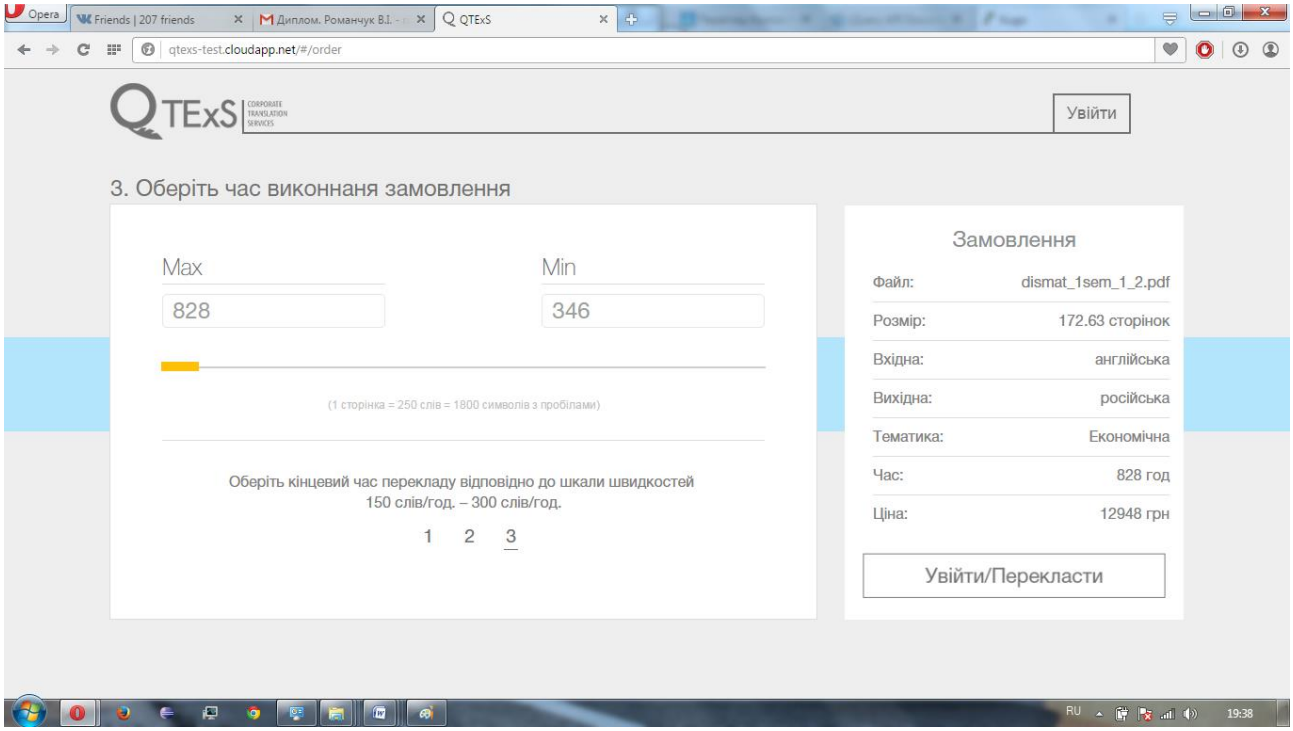

Рисунок 25

Мінімальний та максимальний часовий діапазон перекладацької спроможності

Анімаційні моменти:

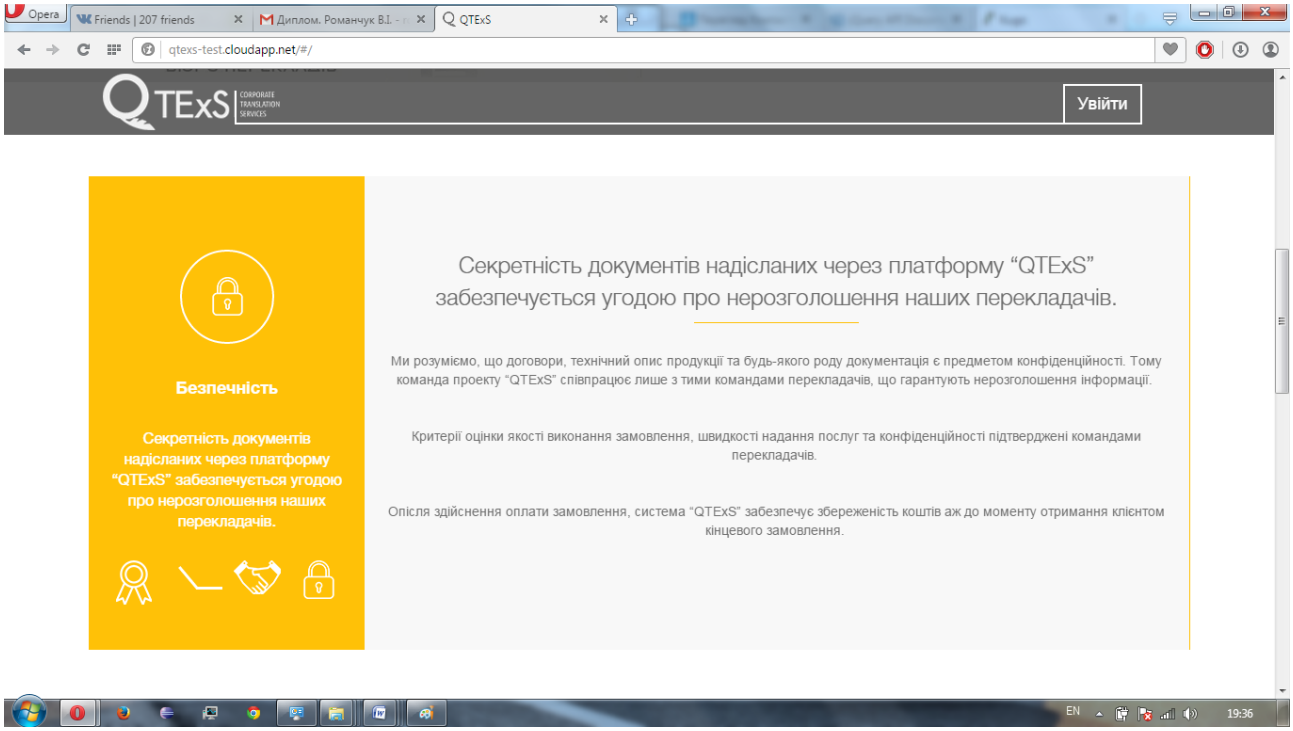

Рисунок 26

Переваги даної платформи:

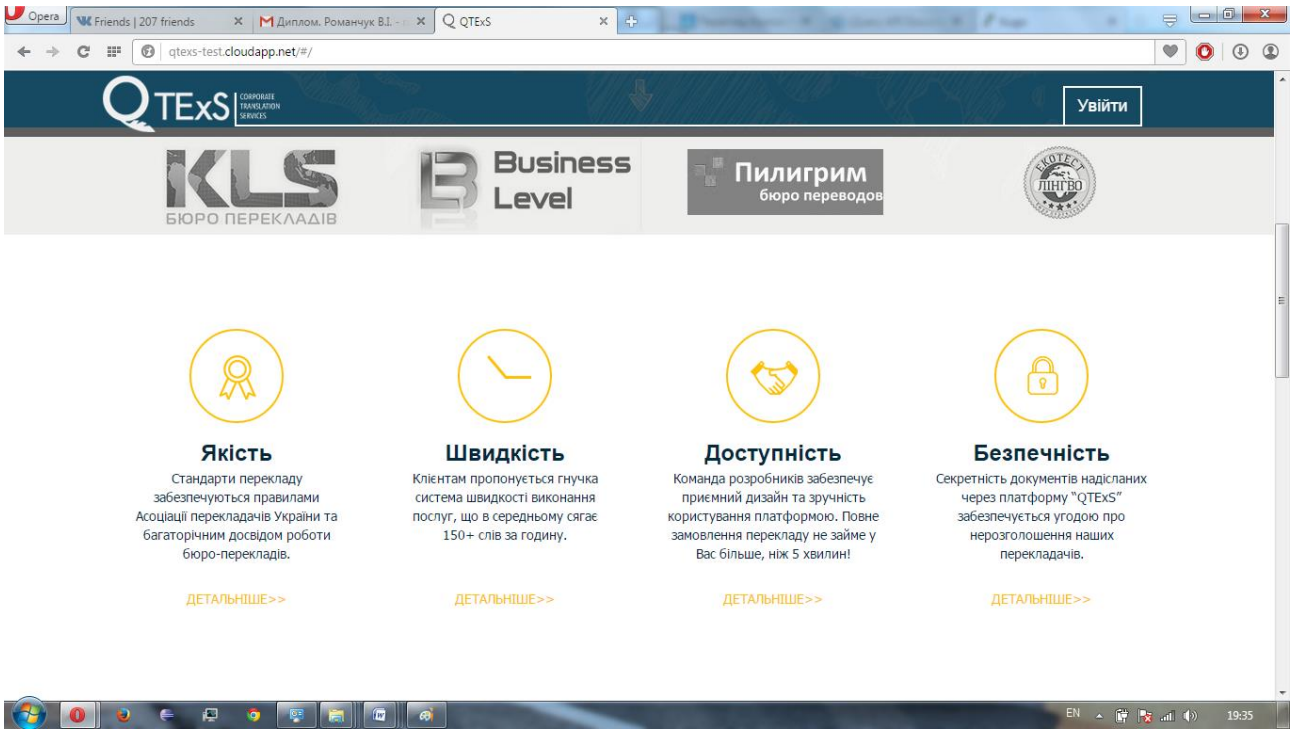

Рисунок 27

Калькулятор ціни та наданої нами швидкості перекладу:

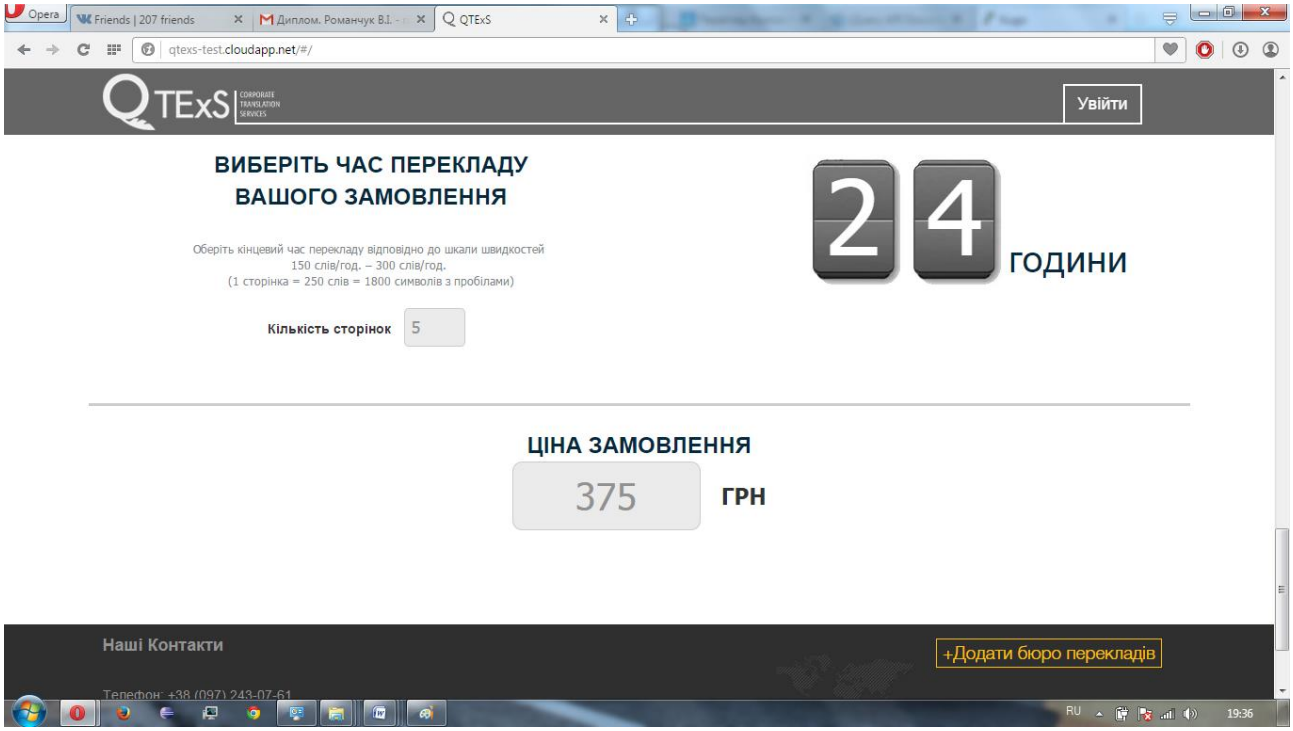

Рисунок 28

3. ОХОРОНА ПРАЦІ

3.1. Вступ

Згідно з Законом України «Про охорону праці», охорона праці - це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності. Цей Закон визначає основні положення щодо реалізації конституційного права працівників на охорону їх життя і здоров'я у процесі трудової діяльності, на належні, безпечні і здорові умови праці, регулює за участю відповідних органів державної влади відносини між роботодавцем і працівником з питань безпеки, гігієни праці та виробничого середовища і встановлює єдиний порядок організації охорони праці в Україні.

Далі наведено параметри приміщення, яке відповідає чинним правилам безпеки, а також рекомендовану систему вправ для зменшення втомлюваності під час роботи.

3.2. Характеристика приміщення

Набір програмних продуктів, який використовується у дипломній роботі, використовуватиметься в приміщенні, план якого приведено на рис. 2

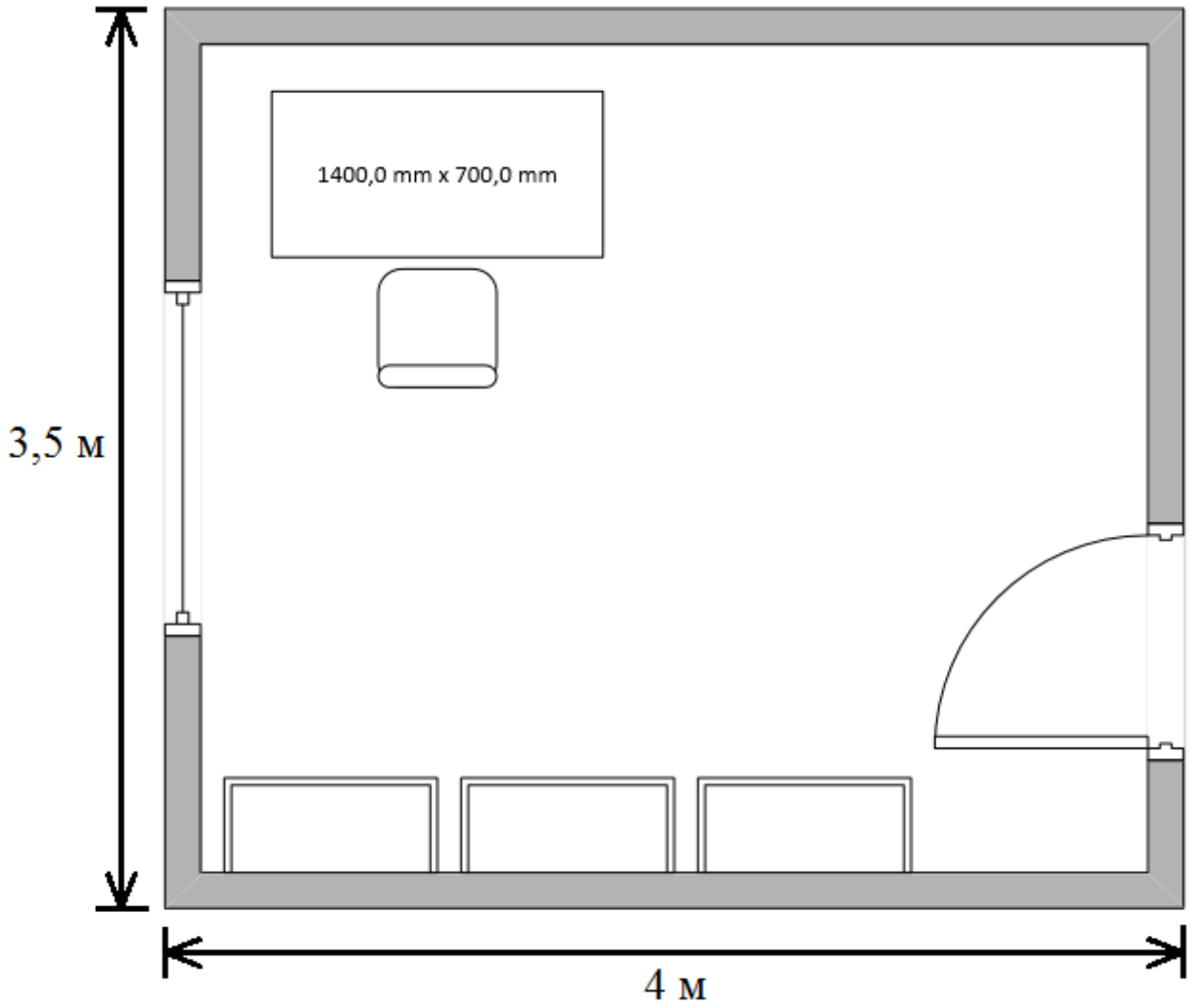

Рисунок 29 – План приміщення

Приміщення має одностороннє природне освітлення і загальне штучне освітлення. Стіни і стеля обклеєні світлими шпалерами, підлога вкрита темним ламінатом. У приміщенні відсутні сильні вібрації та шкідливі речовини. Склад повітря в нормі.

У кімнаті знаходиться ПК з 4-ядерним процесором і 23-дюймовим IPS монітором, а також меблі.

Приміщення має довжину 4 м, ширину 3.5 м, висоту стелі 2,7 м. Кількість робочих місць – одне. Приміщення знаходиться на шістнадцятому поверсі 20-поверхової цегляної будівлі. Площа – 14 м^2, об'єм – 37,8 м^3. Виходячи з цього, отримано дані, наведені в табл. 3.1. Нормативні значення згідно з [32].

Таблиця 3.1 – Фактичні та нормативні значення параметрів приміщення

Параметр	Норма	Реальні параметри
Площа, S	не менше 6 $м^2$	14 $м^2$
Об’єм, V	не менше 15 $м^3$	37,8 $м^3$

Отримані показники відповідають чинним нормам і вимогам.

3.3. Мікрокліматичні умови

Згідно з [33] цю роботу можна віднести до категорії легка 1а. Джерелами тепла в цьому приміщенні є люди, електроустаткування, освітлювальні прилади в темний час доби і система опалювання взимку. Оператором виділяється до 120 ккал теплової енергії за годину. Оптимальні та фактичні значення параметрів мікроклімату приведені в табл 3.2.

Таблиця 3.2 – Значення мікроклімату

Період року	Параметр	Оптимальний	Фактичний
Теплий	Температура	23 – 25 °С	24 °С
	Вологість	40 – 60 %	50 %
	Швидкість повітря	≤ 0.1 м/с	
Холодний	Температура	22 – 24 °С	23 °С
	Вологість	40 – 60 %	55 %
	Швидкість повітря	≤ 0.1 м/с	

Усі показники задовольняють вимогам зазначеним в [10] для робіт категорії легка 1а і є задовільними для здоров'я людини.

3.4. Освітлення

Згідно з [34] ця робота відноситься до V_a розряду зорових робіт. Передбачається використання природного, штучного і змішаного освітлення.

Природне освітлення здійснюється за допомогою вікна, площа якого складає $S' = 1,8*1,5 = 2,7$ м^2 і є боковим освітленням.

У світильниках місцевого і загального освітлення використовуються лампи розжарювання потужністю 75 Вт із світловим потоком лампи 940 лм.

Освітлення задовольняє нормам.

3.5. Шум та вібрація

Джерелом шуму в приміщенні є комп'ютер.

Вентилятори (кулери) системного блоку, процесора, відеокарти і блоку живлення є сучасними і мають низький рівень шуму. Згідно з технічною документацією шум, зумовлений кулером в блоці живлення складає 25 дБ, кулером процесора - 30 дБ, загальний, - 34 дБ. Враховуючи незначний рівень шуму від персонального комп'ютера і незначний рівень фонового шуму від іншого устаткування, можна стверджувати, що сумарний рівень шумового забруднення приміщення не перевищує максимально допустимий рівень коригованої звукової потужності і складає не більше 50 дБ.

При роботі з персональним комп'ютером в робочому приміщенні значення характеристик вібрації на робочих місцях не повинна перевищувати допустимих значень.

3.6. Випромінювання

У приміщенні відсутні джерела інфрачервоного, ультрафіолетового і електромагнітного випромінювання, бо монітор ПК вироблений на основі

рідкокристалічної матриці, підсвітка якої здійснюється неоновою лампою, що не має сильного електромагнітного випромінювання і сертифіковані в Україні.

Блок живлення є екранованим і не випускає вищезазначених видів випромінювання.

3.7. Ергономіка робочого місця

Обладнання і організація робочого місця працівників з ПЕОМ мусять відповідати конструкцій всіх елементів робочого місця.

Висота робочої поверхні складає 700 мм, має простір для ніг 800 мм, довжина столу 1400 мм, ширина – 700 мм. Робочий стілець – обертальне крісло з регулятором висоти.

Робоче місце відповідає вимогам [32].

Для захисту очей забезпечення комфортного куту огляду 23-дюймового монітора відстань від екрану монітора до користувача складає 600-700 мм, робота за комп'ютером виконується не більше 4 годин в день і проводяться періодичні перерви (через кожних 2 години на 10-15 хвилин). Також періодично виконується комплекс вправ для очей.

Рекомендований комплекс вправ для очей:

1. Міцно примружитися на 3-5 секунд. Відкрити очі і не блимати 3-5 секунд. Повторити 7-8 разів.
2. Закрити очі і масажувати їх круговими рухами пальців, не натискаючи на очні яблука, протягом 1-2 хвилин.
3. Поставити вказівні пальці під кожною бровою і злегка потягнути ними шкіру вгору, щоб піднялися очі. Потримати протягом 1-2 секунд, потім відпустити. Повторити 5 разів.

3.8. Висновки

Аналіз умов праці в розглянутому робочому приміщенні показав, що умови праці з ПЕОМ відповідають вимогам [32,33,34,35] оскільки площа та

об’єм не є меншими за нормативні значення, а рівні шуму, вібрації і загазованості не перевищують нормативних обмежень.

Для підтримання параметрів мікроклімату в приміщенні встановлено радіатор центральної водяної системи опалення, що складається з 11 секцій. Ергономіка робочого місця і режим зорової роботи задовольняють вимогам і сприяють зниженню втоми.

4. ВИСНОВКИ

З розвитком галузей промисловості та ввезенням імпортних товарів виникла значна необхідність у швидкому та якісному перекладі, швидкість перекладу забезпечується кваліфікованими спеціалістами, що контролюються асоціацією перекладачів.

В результаті виконання бакалаврської роботи:

1. Проведено аналіз існуючих засобів побудови web-порталів
2. Вивчено досвід побудови web-порталів з точки зору пошуку комерційної ефективності.
3. Реалізовано ядро системи, що забезпечило старт для подальшого розвитку програмного продукту. Складене технічне , завдання. В ядрі системи була використана система LiqPay, з відкритою API.
4. Для підвищення ефективності розробки застосовано SCRUM як процес створення завдавань для проектування.
5. Створена документація у вигляді презентацій та діаграм використання.
6. Розроблено зручний інтерфейс користувача, що підтверджено консультаціями з досвідченими галузевими дизайнерами.

ПЕРЕЛІК ПОСИЛАНЬ

1. Офіційний сайт компанії Pangeanic - Режим доступу : http://www.pangeanic.com/knowledge_center/size-transl. - Дата доступу: 13.06.2015
2. Winter Green. Language Translation Software Market Shares and Forecasts, Worldwide / Market Research, 2011-2017 – January 2011
3. Офіційний сайт компанії Translation, Terminology and Interpretation - Режим доступу : http://www.bt-tb.tpsgc-pwgsc.gc.ca/publications/docum. - Дата доступу: 13.06.2015
4. Офіційний сайт компанії Translationrating - Режим доступу : http://www.translationrating.ru/results2013ukraine/# - Дата доступу: 13.06.2015
5. Офіційний сайт компанії Lingoware - Режим доступу : http://lingoware.ru/analitika/desjat-mifov-o-perevodc.. - Дата доступу: 13.06.2015
6. Том Армстронг ActiveX-Создание Web-приложений, глава 6/ Том Армстронг : БХВ-Петербург, 1998
7. Офіційний сайт компанії Gengo - Режим доступу : Gengo.com - Дата доступу: 13.06.2015
8. Викрам Васвани. Zend Framework. Разработка веб-приложений на PHP/ Викрам Васвани : Питер, 2012
9. Том Кристиансен, Брайан Д Фой, Ларри Уолл, Джон Орвант. Programming Perl / Том Кристиансен, Брайан Д Фой, Ларри Уолл, Джон Орвант : Символ-Плюс, 2006
10. Том Армстронг ActiveX-Создание Web-приложений, глава 6/ Том Армстронг : БХВ-Петербург, 1998.
11. Брюс Эккель Философия Java / Брюс Эккель : Питер, 2015
12. Офіційний сайт компанії ARM - Режим доступу : http://www.arm.com/products/processors/technologies/jazelle.php - Дата доступу : 13.06.2015

13. Герберт Шилдт. Java 8. Полное руководство / Герберт Шилдт : Вильямс, 2015
14. Зуенко А.А. Семантический интерфейс реляционных баз данных в системах моделирования для слабо формализованных предметных областей / Зуенко А.А. : Санк-Петербург, 2009
15. Офіційний сайт компанії Progopedia - Режим доступу : http://progopedia.ru/typing/implicit/ - Дата доступу : 13.06.2015
16. Офіційний сайт компанії UA5.ORG - Режим доступу : http://www.ua5.org/database/189-reljacjjna-baza-danikh.html - Дата доступу : 13.06.2015
17. Офіційний сайт компанії Oracle - Режим доступу : https://wikis.oracle.com/pages/viewpage.action?pageId=27394602 - Дата доступу : 13.06.2015
18. Офіційний сайт компанії Yahoo - Режим доступу : https://answers.yahoo.com/question/index?qid=20091113125347AAmUwL6 - Дата доступу : 13.06.2015
19. Сайт Навчальні матеріали онлайн - Режим доступу : http://pidruchniki.com/19350610/informatika/osnovni_etapi_rozvitku_informatsiynih_sistem - Дата доступу : 13.06.2015
20. Недашківський О.Л. Планування та проектування інформаційних систем / Недашківський О.Л. : Київ, 2014
21. Офіційний сайт компанії Vkontakte - Режим доступу : https://vk.com/qtexs - Дата доступу : 13.06.2015
22. Офіційний сайт компанії Facebook - Режим доступу : https://www.facebook.com/qtexs?ref=profile - Дата доступу : 13.06.2015
23. Офіційний сайт компанії Twitter - Режим доступу : https://twitter.com/InfoQtexs дата доступу – 13.06.2015
24. Офіційний сайт компанії Github - Режим доступу : https://github.com/liqpay?query=sdk - Дата доступу : 3.06.2015

25. Офіційний сайт компанії Liqpay - Режим доступу : https://www.liqpay.com/ua/doc#liq_buy - Дата доступу : 19.04.2015
26. Офіційний сайт компанії Booking - Режим доступу : http://booking.uz.gov.ua - Дата доступу : 13.06.2015
27. Офіційний сайт компанії Oracle - Режим доступу : http://www.oracle.com/technetwork/java/dataaccessobject-138824.html - Дата доступу : 13.06.2015
28. Сайт habrahabr - Режим доступу : http://habrahabr.ru/post/26316/ - Дата доступу : 13.06.2015
29. Сайт Портал знань - Режим доступу : http://www.znannya.org/?view=patterns-command - Дата доступу : 13.06.2015
30. Сайт http://getbootstrap.com - Режим доступу : http://getbootstrap.com/css/#forms - Дата доступу : 30.05.2015
31. Сайт http://getbootstrap.com - Режим доступу : http://hayageek.com/docs/jquery-upload-file.php - Дата доступу : 30.05.2015р
32. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин /ДСанПіН 3.3.2.007-98. – К.: Постанова Головного державного санітарного лікаря України, 1998. - № 7.
33. Санітарні норми мікроклімату виробничих приміщень : ДСН 3.3.6.042-99. – К., 2000.- С.16
34. Природне і штучне освітлення : ДБН В.2.5-28:2006. – К. : Міністерство будівництва, архітектури та житлово-комунального господарства України, 2006. – 68 с. – (Національні стандарти України).
35. Норми визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпеко : НАПБ Б.03.002-2007. – К.: Наказ МНС від 03.12.2007 № 833.