**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Тернопільський національний економічний університет**

**Факультет комп'ютерних інформаційних технологій**

Кафедра комп'ютерних наук

**Тихий Роман Романович**

**Мобільний додаток для вивчення англійської мови/ Mobile application for learning English**

напрям підготовки: 6.050103 - Програмна інженерія фахове спрямування - Програмне забезпечення систем

Бакалаврська дипломна робота

Виконав студент групи ПЗС-42 Р. Р. Тихий

Науковий керівник: викладач Олійник І.С.

Бакалаврську дипломну роботу допущено до захисту:

" " 20 р. Завідувач кафедри

**А. В. Пукас**

**ТЕРНОПІЛЬ - 2019**

РЕЗЮМЕ

Дипломна робота містить Х сторінок, У таблиць, Z рисунки, список використаних джерел із B найменування та C додатки.

Метою дипломної роботи є розробка мобільного додатку для вивчення англійської мови.

Об’єкт дослідження – процеси покращення знань англійської мови.

Предмет дослідження – методи та засоби для розробки мобільного додатку для вивчення англійської мови.

Методи розробки базуються на мові програмування C#, платформі Xamarin, Asp net core 2.2 та СУБД MySQL/SQL server.

Результатом є готовий додаток для вивчення англійської мови.

Ключові слова: C#, Xamarin, Asp net core 2.2, MYSQL, SQL server

SUMMARY

Thesis contains 95 pages, 12 tables, 43 figures, list of sources with 21 titles and 4 applications.

The aim of the thesis is the development a mobile application for learning English.

The object of research is a means the process of increasing the knowledge of English.

The subject of research is development of mobile application to the system of learning English.

Technologies used to develop software are programming language C#, платформі Xamarin, Asp net core 2.2 та СУБД MySQL/SQL server.

The result is a completed mobile application for learning English.

Keywords: C#, Xamarin, Asp net core 2.2, MYSQL, SQL

ЗМІСТ

ВСТУП

РОЗДІЛ І АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

* 1. Коротка характеристика об’єкту управління
  2. Опис предметної області
  3. Огляд і аналіз існуючих аналогів
  4. Специфікація вимог до модуля (системи)

Висновки до розділу I

РОЗДІЛ ІІ ПРОЕКТУВАННЯ

* 1. Розробка архітектури програмної системи
  2. Проектування структури бази даних

Висновки до розділу II

РОЗДІЛ ІІІ ПРОГРАМНА РЕАЛІЗАЦІЯ

* 1. Програмна реалізація проекту
  2. Програмна реалізація бази даних

Висновки до розділу III

РОЗДІЛ ІV ТЕСТУВАННЯ ТА ДОСЛІДНА ЕКСПЛУАТАЦІЯ

|  |  |
| --- | --- |
| 4.1 Тестування |  |
| * 1. Розгортання програмного продукту   2. Інструкція користувача |  |
| Висновки до розділу IV |  |
| ВИСНОВКИ  СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ |  |
| ДОДАТОК А ВИХІДНИЙ КОД РОЗРОБКИ МОБІЛЬНОГО ДОДАТКУ ДЛЯ ВИВЧЕННЯ АНГЛІЙСЬКОЇ МОВИ |  |
| ДОДАТОК Б РЕЗУЛЬТАТ ФІЗИЧНОГО ПРОЕКТУВАННЯ БД  ДОДАТОК В РЕЗУЛЬТАТИ ТЕСТУВАННЯ  ДОДАТОК Г ФУНКЦІОНАЛЬНІ ТЕСТОВІ ВИПАДКИ |  |

ВСТУП

Зараз важко уявити наше повсякденне життя без використання електронно обчислювальних носіїв. Вони мають місце уже практично у всіх сферах людського життя: наука, медицина, спорт, машинобудування, інфраструктура та багато інших. В наш час дуже важливо «бути в темі» та встигати за цим стрімким розвитком. Думаю всі розуміють наскільки велике значення технології мають у нашому житті.

На мій погляд, сучасні людині просто необхідно володіти знанням англійської мови, яка є найпоширенішою та основою для всієї індустрії інформаційних технологій. Подібні системи є дуже зручними, адже вони дозволяють вчитися у будь-якій точці світу. Інтерактивне навчання стає все дедалі популярнішим, чого варті лише різні платформи з онлайн курсами. Як приклад це: Coursera, Udemy, Microsoft online academy та український Prometheus. Зараз у одному смартфоні може вміщуватись сила-силенна цікавих функцій, які допомагатимуть нам у найбуденніших справах, економлячи наш час та енергію. Тепер не потрібно надокучати постійно зайнятим місцевим мешканцям, випитуючи дорогу, пропускати попутні маршрутки, чекаючи на «свою» і сподіваючись, що от-от приїде, довго ламати мозок над тим, що сьогодні одягнути. Й у продуктовому магазині можна зорієнтуватися на відстані. Враховуючи актуальність даного напрямку, в якості теми для роботи була прийнята тема «Мобільний додаток» та мова програмування C#, в якості базової для реалізації.

Життя сучасної людини важко уявити без смартфона. Ми використовуємо ці багатофункціональні пристрої для різних задач. Тому чому б не використовувати їх з метою навчання англійської. Зараз найбільш поширеними операційними системами для мобільний пристроїв є Adnroid та IOS. А мова програмування C# та новітній фреймворк для розробки додатків Xamarin надають всі необхідні засоби та інструменти. На мій погляд зараз саме не хватає подібних додатків та систем. Люди використовують смартфони для розваг або для того щоб «вбити час», тож я вважаю що їх можна використовувати з користю та для розвитку.

Метою роботи є аналіз та детальне дослідження уже існуючих програмних продуктів для вивчення англійської мови і розробка власного мобільного додатку, на основі проаналізованих програмних рішень, який б дозволив використовувати ряд функцій та інструментів для вивчення англійської мови.

Отож, спираючись на мету, можна виділи такі основні задачі, які ставляться до розроблюваного продукту:

1. Проаналізувати мобільні додатки для вивчення англійської мови які існують зараз.
2. На основі опрацьованого матеріалу розробити функціональні та не функціональні вимоги до розроблюваної програмної системи.
3. Спроектувати базу даних програмної системи.
4. Обрати методи та засоби реалізації для втілення в життя поставленого завдання.
5. Розробити мобільний додаток для вивчення англійської мови згідно з виконаними задачами, які описані в пунктах 1-4.

Об’єкт дослідження – процеси покращення рівня володіння англійською мовою.

Предмет досліджень – застосування сучасних методів та технологій проектування для створення мобільного додатку для вивчення англійської мови.

Під час створення системи буде використано наступні технології, необхідні для розробки системи:

* C# – одна з найкращих мов програмування на даний час, підтримувана компанією Microsoft;
* ASP NET core – платформа для створення веб-орієнтованих додатків створена компанією Microsoft;
* Xamarin – платформа для розробки мобільний додатків, які запускають як на Android так і на IOS;
* SQL – мова, яка дає змогу взаємодіяти із даними, які зберігаються у таблицях БД та виконувати різного роду маніпуляції із ними.

Практична цінність розроблюваного програмного продукту полягає в наданні користувачу зручних можливостей, функцій та інструментів для вивчення англійської мови, і наданні можливості доступу до системи із будь-якого місця з покриттям мережі Інтернет.

РОЗДІЛ І

АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

* 1. Коротка характеристика об’єкту управління

В світі існує безліч країн, а мов, якими розмовляють їх жителі, ще більше. Як же населенню Землі зрозуміти один одного? Для цього існують міжнародні мови, які і дають змогу всім нам спілкуватися, незалежно від національності та місця проживання. Однією з міжнародних мов є **англійська**. Значення англійської мови в сучасному світі важко переоцінити. Адже не можна не зважати на вибір більше 1 мільярда людей, які її використовують. І якщо для половини з них вона є рідною, то близько 600 мільйонів вибрали саме її як іноземну.

Яку ж роль в нашому житті зараз відіграє англійська мова? В цілому, політичне, економічне, наукове, спортивне життя всього світу «протікає» англійською мовою. Всілякі саміти і зустрічі глав держав, підписання указів, дебати – все це проводиться англійською мовою. Міжнародна торгівля, робота банківської системи, діяльність транспортної системи здійснюється англійською мовою. Міжнародні конференції, вивчення світового досвіду та обмін інформацією відбувається лише з використанням англійської мови. Та що там говорити – олімпійські ігри і всілякі змагання між країнами вибрали офіційною мовою саме англійську. Ми ще не враховували кіноіндустрію та музичний олімп. Фільми американського виробництва міцно увійшли в наше життя, а будь-який поп-виконавець вважає престижним заспівати не менше однієї пісні англійською мовою.

Значення англійської мови в сучасному світі настільки велике, що її знання не є розкішшю. Її вчать скрізь: у школах, університетах, на курсах. Будь-яка освічена людина просто зобов'язана володіти англійською мовою, тому що саме вона є ключем до подальшої самоосвіти та самовдосконалення. Хочете подорожувати і не відчувати себе білою вороною, а вільно спілкуватися з людьми різних національностей? Хочете стати студентом закордонного університету? Хочете престижну роботу з просуванням по кар'єрних сходах? А, може, ви бажаєте працювати закордоном? Порада одна – вивчайте англійську мову!!!

Знання англійської мови в сучасному світі є своєрідним вікном у світ. Володіючи цією мовою міжнародного спілкування, ви зможете досягти поставлених цілей за допомогою нових можливостей. І ви обов'язково зрозумієте, що значення англійської мови не перебільшено!

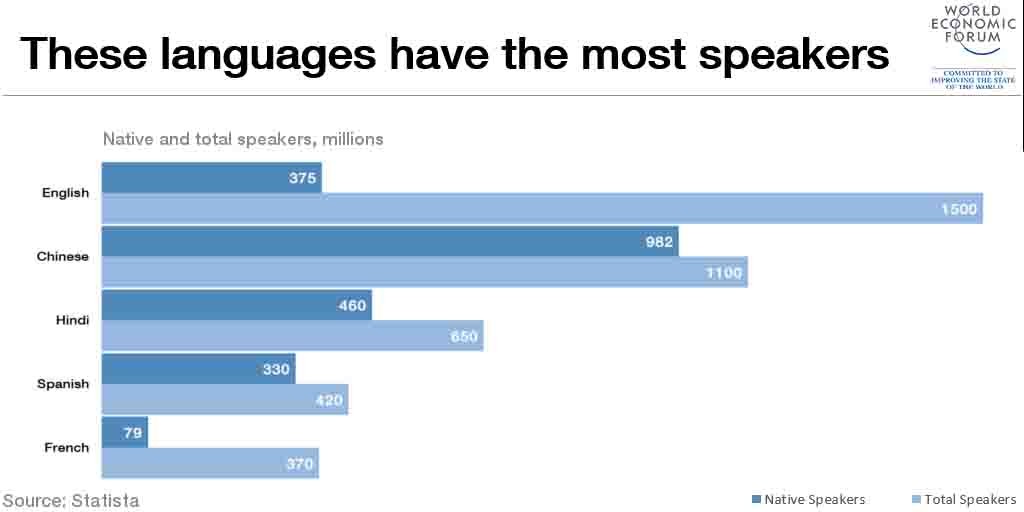


Рисунок 1.1 – Кількість носіїв англійської мови

Англійська є рідною мовою для більш ніж 370 млн. і другою – для 611 млн. людей, проте ще сотні мільйонів знають її, бо це **міжнародна мова бізнесу і науки**.

Точно визначити кількість людей, що говорять певною мовою, та рівень володіння нею важко, проте, за деякими оцінками, до 2 млрд. людей (тобто близько чверті населення Землі) розуміють англійську хоча б на базовому рівні чи користуються нею у спілкуванні.

За даними Вікіпедії, англійською говорять щонайменше в 115 країнах (із приблизно 200 країн світу), а офіційною мовою вона є в майже 60 країнах, в тому числі. таких світових лідерах, як Великобританія, США, Індія, Канада і Австралія. Також вона є однією з шести робочих мов ООН. Дві третини наукових праць і половина газет у світі публікуються англійською. 51% інтернет-сайтів у світі є англомовними. Згідно звіту Internet World Stats, станом на червень 2017 року 985 млн. (25.3%) користувачів Інтернету спілкувалися англійською.

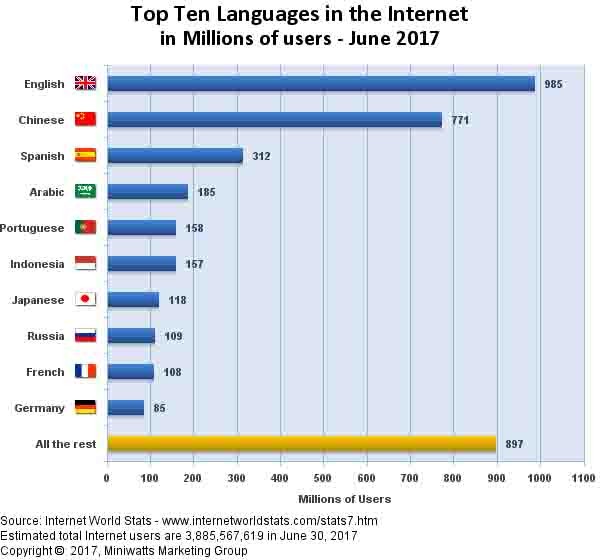


Рис 1.2 - Найбільш популярні мови серед користувачів інтерн

* 1. Опис предметної області

Моделювання бізнес процесів на початку створення системи є дуже важливим етапом у створенні програмного забезпечення, адже він надає можливість побачити загальну картину та зрозуміти основні функції.

Для належної організації роботи мобільного додатку для вивчення англійської мови необхідне виконання таких основних бізнес-процесів (див. рис. 1.2):

* процес створення на наповнення словників;
* процес проходження тестування;
* процес перекладу тексту зчитаного з камери.

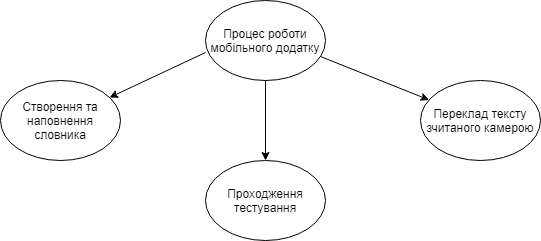


Рисунок 1.3 – Діаграма бізнес-процесів розроблюваного програмного продукту

Розглянемо детальніше кожен з вище представлених бізнес-процесів. На рисунку 1.3 бачимо діаграму функцій процесу створення та наповнення словника.

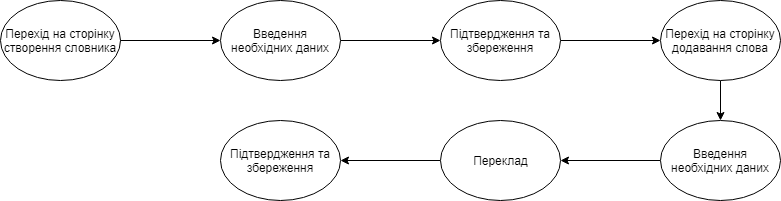


Рис 1.3 – Діаграма функцій процесу «Створення та наповнення словника»

Одним із етапів роботи з системою є створення та наповнення словників. Цей етап складається з:

* введення даних для створення словника (користувач вводить назву створюваного словника)
* підтвердження збереження (дані зберігаються у БД)
* додавання слова у словник. Введення даних для додавання слова (текст, мова оригіналу, мова для перекладу)
* переклад тексту на вибрану мову
* підтвердження збереження (дані зберігаються у БД)

Характеристику бізнес-процесу «Створення та наповнення словника» наведено в таблиці 1.1.

Таблиця 1.1

Характеристика процесу створення словника

|  |  |
| --- | --- |
| Назва характеристики | Значення характеристики |
| Ім'я бізнес-процесу | Створення та наповнення словника |
| Основні учасники | Користувач |
| Вхідна подія | Введення даних для створення словника |
| Вхідні документи | Назва створюваного словника |
| Вихідна подія | Збережений у БД кортеж із даними студента |
| Вихідні документи | - |
| Клієнт бізнес-процесу | Створений словник |

На рисунку 1.4 бачимо діаграму функцій процесу «Проходження тестування»

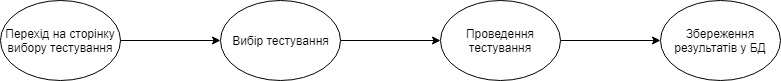


Рис 1.4 – Діаграма функцій процесу «проходження тестування»

Цей процес складається з таких основних пунктів, а саме:

* вибір тестування (лексика, граматика та інші теми)
* проведення тестування (безпосередньо проходження тестів, відповіді на запитання)
* збереження результатів у БД

Характеристику бізнес-процесу «Проходження тестування» наведено в таблиці 1.2.

Таблиця 1.2

Характеристика бізнес-процесу «Проходження тестування»

|  |  |
| --- | --- |
| Назва характеристики | Значення характеристики |
| Ім'я бізнес-процесу | Проходження тестування |
| Основні учасники | Користувач |
| Вхідна подія | Вибір тестування (лексика, граматика та інші теми) |
| Вхідні документи | - |
| Вихідна подія | Збереження результатів у БД |
| Вихідні документи | - |
| Клієнт бізнес-процесу | Формування результату проходження тестування |

На рисунку 1.5 бачимо діаграму функцій процесу перекладу тексту зчитаного камерою.

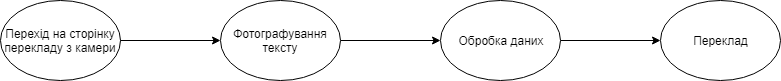


Рис 1.5 – Діаграма функцій процесу «Переклад тексту зчитаного камерою»

Цей процес складається таких пунктів:

* фотографування тексту
* обробка даних та зчитування тексту за допомогою AI алгоритмів
* переклад зчитаного тексту на обрану мову

Характеристика процесу «Формування звітності»

Таблиця 1.3

|  |  |
| --- | --- |
| Назва характеристики | Значення характеристики |
| Ім'я бізнес-процесу | Переклад тексту зчитаного камерою |
| Основні учасники | Користувач |
| Вхідна подія | Фотографування тексту |
| Вхідні документи | Фото тексту |
| Вихідна подія | Переклад тексту |
| Вихідні документи | Перекладений текст |
| Клієнт бізнес-процесу | Перекладений текст |

* 1. Огляд і аналіз існуючих аналогів

На етапі проектування програмного продукту доцільно розглянути існуючі на даний час програми, які безпосередньо або опосередковано виконують функції програмного забезпечення для вивчення англійської мови, зокрема мобільні додатки.

При виконанні аналізу конкурентних сервісів для вивчення англійської мови, я виділив найбільш популярні додатки: «Sentense master pro», «Duolingo», «Memrise». Мій додаток буде надавати функціонал для зручного і легкого вивчення англійської мови, створювати власні словники та групувати слова, продходити інтерактивні тести з граматики, лексики та ін, здійснювати переклад тексту знятого на камеру пристрою та вивчати правильну вимову. Користувачем цього додатку може бути будь-хто, хто бажає покращити свої знання з англійської та отримати зручні інструменти для цього.

“**Sentense master pro”**

Це фантастична мовна програма, спрямована на те, щоб допомогти молодим людям поліпшити свої мовні здібності. Орієнтуючись на побудову речень, програма заохочує користувачів кинути виклик у ігровому середовищі, будувати речення розставляючи слова у правильному порядку. Суть гри полягає в тому, щоб дати користувачеві кілька слів, які він складе у певному порядку а потім розшифрувати і сформувати повне речення в межах певного проміжку часу. І при цьому, якщо користувач вибирає неправильні відповіді він втрачає час.

Одним з найбільш привабливих аспектів програми є той факт, що він може використовуватися різними людьми з різним рівнем володіння англійської мови. Наприклад, він буде корисним, як додаток для дітей молодшого віку, які вже мають певний рівень володіння читанням і письмом, щоб покращити своє розуміння так і для користувачів старшого віку, особливо якщо вони вивчають англійську мову як додаткову і хочуть покращити свою кваліфікацію. Він також може бути використаний для більш досвідчених користувачів, які хочуть продовжувати збагачувати свої власні знання. Даний мобільний надає можливість вивчити велику кількість сталих фраз, висловів, приказок та прислів’їв, але це завдання не з легких, тому як бичимо майже кожен може знайти собі щось корисне.

Веб-сайт розробника встановлює, що програма була повністю розроблена педагогами, і чим більше ви використовуєте програму, тим більше розумієте, що це так, від дизайну і до самого вмісту завдань.



Рис 1.6 – Процес гри в додатку «Sentense master pro»

Зі слів розробників даний продукт може бути використаний як автономний додаток і підключення до інтернет мережі не є обов’язковим, але для повної синхронізації і зручності воно необхідне. Вони мають цілий набір додатків, спрямованих на вдосконалення англійської та інших мов.

Основні функції системи:

* вивчення англійської мови за рахунок гри, де користувач складає речення розставивши слова у правильному порядку
* вивчення лексики
* вивчення сталих фраз, висловів, прислів’їв та приказок
* відстежування результатів та прогресу
* велика база завдань та слів

Основні недоліки системи

* платний контент
* версія для IOS платна
* погана робота на старих версіях системи Android

компанії «Первый БИТ», модуль «Призначення стипендії» для

«1С:Підприємства».

Модуль «Обрахування стипендії» компанії Іnfosuite по своїй суті являє собою прикладне рішення, призначене для обліку студентів і обрахунку стипендії, яке забезпечує:

* облік студентів по факультетах і групам;
* нарахування стипендій різного типу;
* нарахування утримань;
* розрахунок і виплату стипендії з врахуванням утримань;
* формування звітності по стипендії. Функціональні можливості:
* зберігання інформації про кожного студента, в розрізі груп, підрозділів (факультетів) і курсів з використанням типових довідників «Працівники» та

«Фізичні особи»;

* зберігання списків груп і курсів в спеціальному довіднику «Факультети і групи»;
* можливість внесення ставок стипендії за різні періоди;
* реєстрація кадрових документів (прийом студентів в групи, переміщення студентів із однієї групи в іншу, відрахування (випуск) студентів);
* підбір студентів по параметрах у всіх документах;
* реєстрація даних про розмір і бухгалтерський облік різного типу стипендій
* реєстрація (додавання, зміна, видалення) утримань;
* використання типових механізмів обрахунку заробітної плати для обрахування стипендії;
* використання механізму виправлень для розрахункових документів;
* виплата стипендій, як через касу, так і на банківські картки (рахунки);
* виплата матеріальної допомоги, як оподатковуваної, так і неоподатковуваної;
* депонування невиплачених сум стипендій;
* реєстрація і проведення утримань за виконавчими листами;
* формування різного типу звітів по стипендіях.

Інтерфейс програми зображено на рисунку 1.6

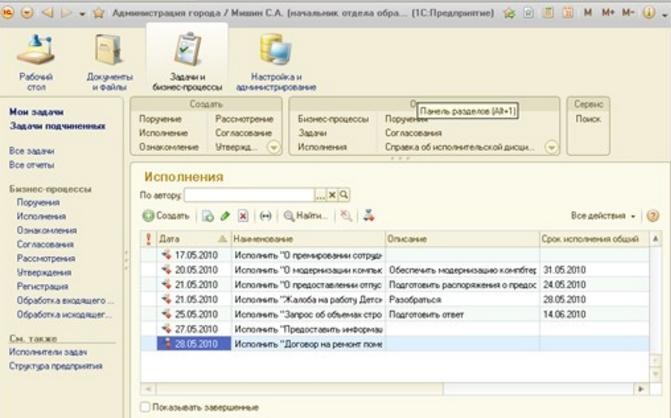


Рисунок 1.6 – Інтерфейс програмного модуля компанії Infosuite

«БИТ. Расчет стипендий 8» - програмна система для автоматизації розрахунку стипендій, кадрового обліку студентів, аналізу планових і фактичних нарахувань з можливістю деталізації в розрізі джерел фінансування. Програма вирішує завдання:

* кадрового обліку студентів;
* розрахунку стипендій та інших надбавок;
* виплата і депонування стипендій;
* розрахунок ПДФО і надсилання даних до податкової інспекції;
* надання всієї необхідної звітності по стипендії;
* формування бухгалтерської звітності. Функціональні можливості:
* автоматизація зарахування, переведення з курсу на курс і відрахування студентів;
* реєстрація даних про розмір і бухгалтерський облік стипендій, реєстрація довільної кількості надбавок і доплат, реєстрація постійних і разових утримань;
* розрахунок ПДФО і надсилання даних до податкової;
* формування платіжних відомостей в банк і касу, депонування;
* відображення в бухгалтерському обліку операцій по нарахуванню і утриманню;
* формування звітності по нарахуванню, розрахункових лисках, звітах по виплаті стипендій та нарахованих податках, книг депонентів, розрахунково

– платіжних відомостях та ін.;

* аналіз нарахувань і виплат за джерелами фінансування;
* деталізований звіт по студенту: кадрова інформація, нарахування та утримання за певний період.

Вигляд головного вікна програми зображено на рисунку 1.7

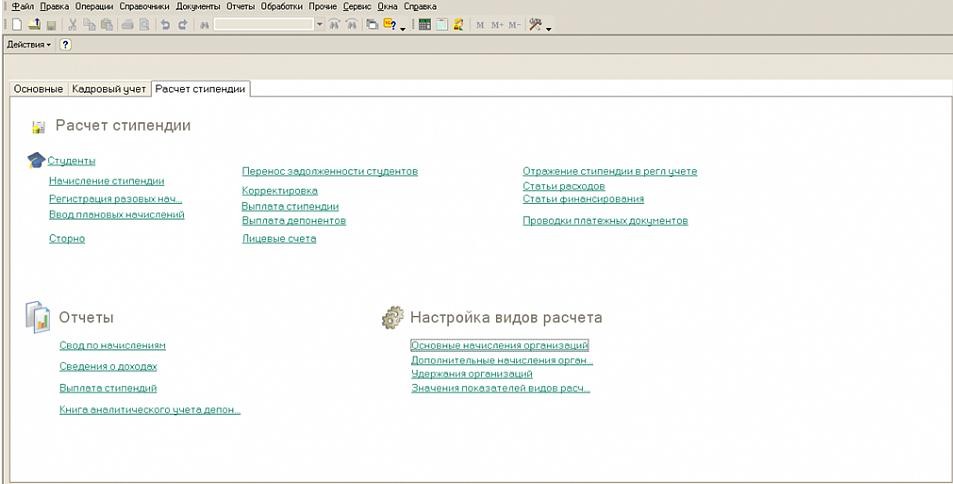


Рисунок 1.7 – Головне вікно програмного модуля «БИТ.Расчет стипендий 8»

Як видно з рисунку 1.7 за допомогою програмного модуля «БИТ.Расчет стипендий 8» можна виконувати такі операції:

1. Розрахунок стипендій:

* нарахування стипендії;
* реєстрація разових нарахувань;
* вивід планових нарахувань;
* перенесення заборгованості студентів;
* корегування;
* виплата стипендії;
* виплата депонентів;
* операції з особовими рахунками;
* відображення стипендії в регулярних звітах;
* інформація про витрати;
* інформація про нарахування;
* проведення платіжних документів.

1. Формування звітів:

* звіт про нарахування;
* звіт про дохід;
* звіт про виплату стипендій;
* Книга аналітичного обліку депонентів.

1. Налаштування виду розрахунку:

* основні нарахування організацій;
* додаткові нарахування організацій;
* утримання організацій;
* значення показників видів розрахунку.

Інформація про кожного студента, список студентів в розрізі груп, підрозділів (факультетів) і курсів зображено на рисунку 1.8 та рисунку 1.9

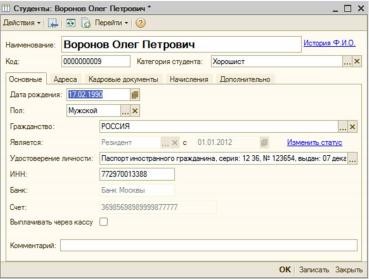


Рисунок 1.8 – Відомості про конкретного студента

Як видно з рисунку 1.8 в зображеному вікні програми можна переглядати та редагувати основні відомості про студента, а також відомості про нарахування стипендії. Також доступним є перегляд кадрової документації.

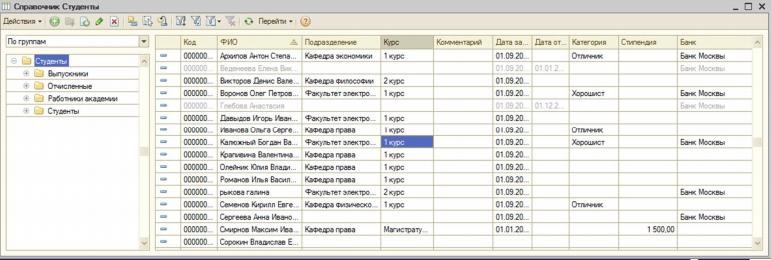


Рисунок 1.9 – Списки студентів

Як видно з рисунку 1.9 в зображеному вікні програми відкривається можливість перегляду списку студентів та основної інформації відповідно до сформованих груп (випускники, відраховані, працівники академії, студенти).

На рисунку 1.10 зображено вікно, яке демонструє автоматизацію щомісячного нарахування стипендії, обчислення нарахувань, утримань, ПДФО. Документ можна заповнити як вручну, так і автоматично.

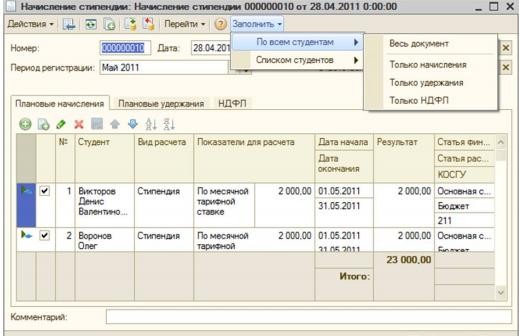


Рисунок 1.10 – Вікно нарахування стипендії

Як видно з рисунку 1.10 в зображеному вікні програми відкривається можливість автоматичного заповнення нарахувань по стипендії, за певний період.

Результатом проведеної операції, представленої на рисунку 1.10, є сформована платіжна відомість, яку показано на рисунку 1.12

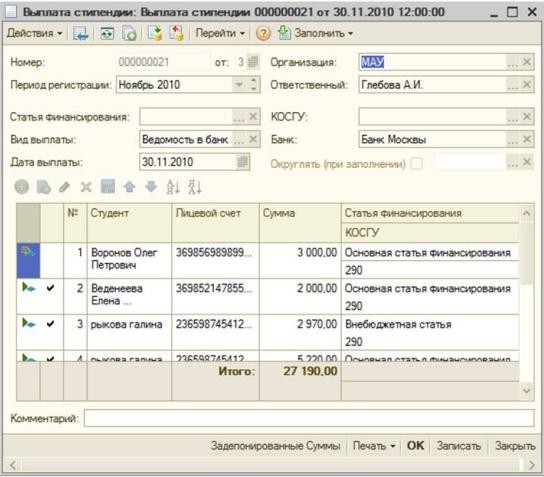


Рисунок 1.11 – Вікно виплати стипендії

Як видно з рисунку 1.11 в зображеному вікні програми відкривається можливість заповнення платіжної відомості та супровідних атрибутів. Після обробки відомість можна відправити на друк. Вигляд сформованої платіжної відомості зображено на рисунку 1.12

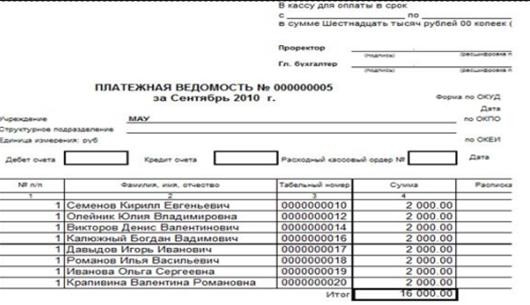


Рисунок 1.12 – Платіжна відомість

Модуль «Призначення стипендії ТНЕУ» для підприємства 1С застосовується для автоматизації процесу нарахування стипендії. Включає в себе можливість ведення журналу успішності, за підсумками сесій та списків пільговиків, відповідно до факультету.

Функціональні можливості:

* зберігання основної інформації про студента, його успішності;
* ведення реєстру пільговиків;
* формування наказів призначення стипендії;
* вибір типу стипендії, відповідно досягнень, освітньо-кваліфікаційного рівня та пільгової категорії (за наявності).

Наказ на призначення стипендії зображено на Рисунку 1.13

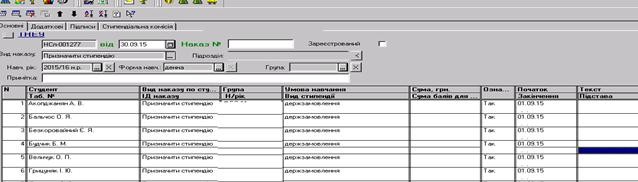


Рисунок 1.13 – Вікно наказу на призначення стипендії

Як видно з рисунку 1.13, наказ на призначення стипендії формується кожним факультетом університету окремо, при чому формування списку студентів-стипендіатів відбувається вручну шляхом почергового додавання груп. При чому вид стипендії, яку призначатимуть студенту потрібно обирати для кожного окремо, відповідно до середнього балу за сесію та пільгової категорії, якщо вона є.

Сформована відомість успішності групи за семестр зображена на рисунку

1.14

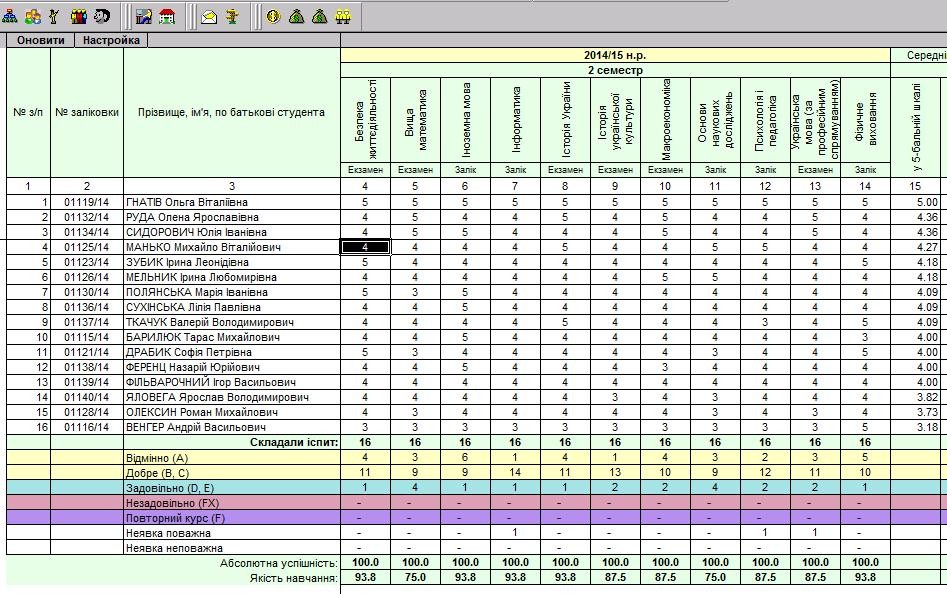


Рисунок 1.14 – Вікно успішності

Отже, якщо студент набрав нижче «4» балів він не претендує на стипендію, виключенням є лише студенти-пільговики, яким достатньо набрати

«3» бали за сесію. Формування успішності студентів за сесію відбувається вручну для кожної групи. Однак у вікні успішності не виокремлюються студенти-пільговики, тому необхідно додатково переглядати реєстр студентів за пільгами. Вибір студентів пільговиків і сформовані списки пільговиків зображено на рисунках 1.15 та 1.16.

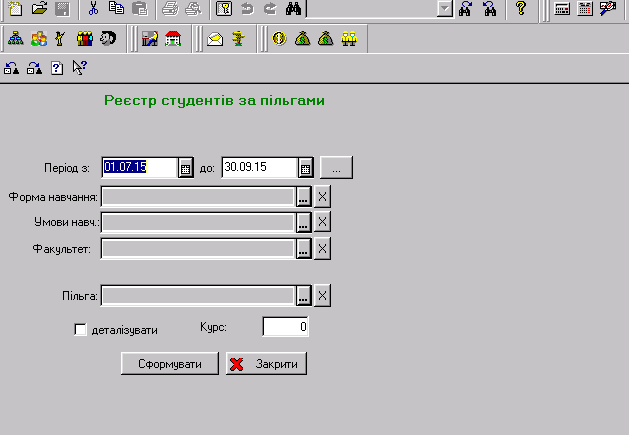


Рисунок 1.15 – Вибір студентів пільговиків

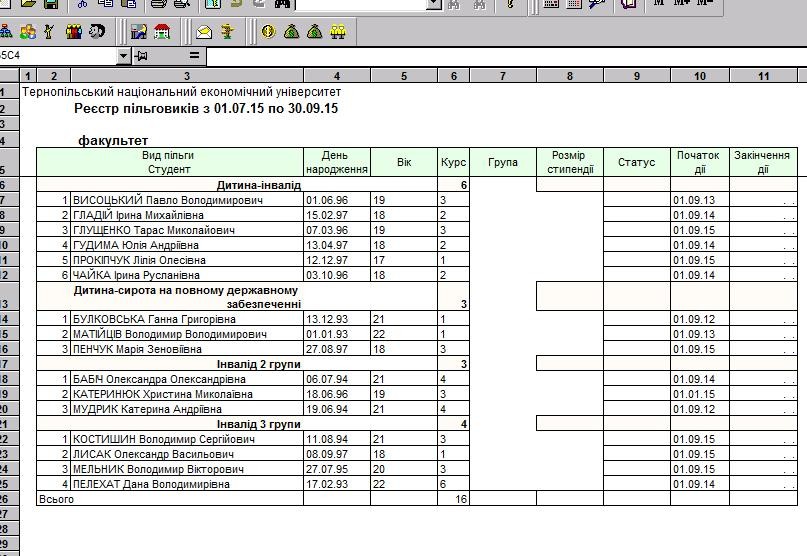


Рисунок 1.16 – Сформований список пільговиків по факультету

Тип стипендії, яку призначатимуть студенту необхідно також обирати вручну, відповідно до середнього балу за сесію та пільгової категорії. Форму вибору типу стипендії зображено на рисунку 1.17

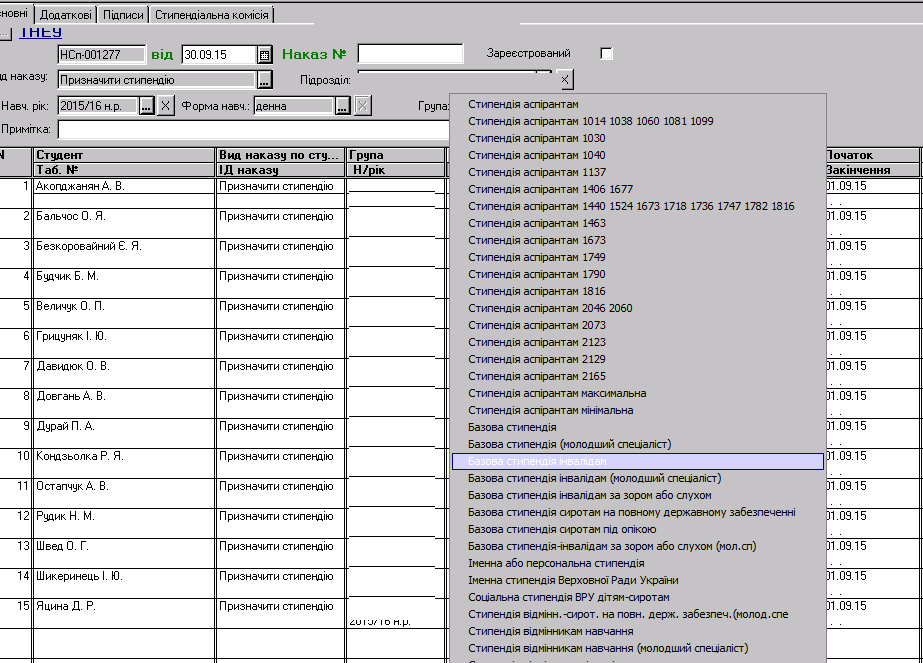


Рисунок 1.17 – Вибір типу стипендії

Проте варто відмітити, що процес визначення можливості отримання стипендії студентом, та процес ідентифікації пільгової категорії(при її наявності) здійснюється вручну. Для кожного освітнього рівня прохідним балом, для отримання стипендії є «4», для отримання підвищеної стипендії «5». Але існують й виключення, при наявності пільгової категорії. Розмір отримуваної стипендії, відповідно до раніше згаданих критеріїв визначається працівником.

Весь цей процес можна спростити, шляхом впровадження певних програмних рішень, за допомогою яких аналіз даних працівником “вручну” - відійшов би на задній план. Цим самим зменшився б вплив людського фактору, мінімізуючи можливість помилки в процесі призначення стипендії.

У таблиці 1.4 представлено порівняльну характеристику програмних продуктів.

Таблиця 1.4

Порівняльна характеристика програмних продуктів

|  |  |  |  |
| --- | --- | --- | --- |
| Фірма-розробник | Infosuite | «Первый БИТ» | 1С:Підприємство |
| Тип продукту | Програмний модуль для  підприємства 1С | Програмний модуль для підприємства 1С | Програмний модуль для підприємства 1С |
| Дата заснування | 18-03-2013 | Лютий, 2010р | - |
| Місце знаходження | Росія, Москва | Росія, Москва | - |
| Версії продукту | Клієнтська ліцензія на 1  робоче місце | БИТ. Расчёт стипендий. Клієнтська ліцензія на 1  робоче місце | 1.0 |
| Функціональність | достатня | середня | середня |
| Інтерфейс користувача | Інтуїтивно- зрозумілий | Простий, одні і ті ж дії можуть бути викликані як за допомогою панелі,  так і через робочий стіл | Максимально простий та доступний користувачу |
| Допомога  користувачу | Відсутня | Присутня | Відсутня |

* 1. Специфікація вимог до модуля (системи)

Кожна предметна область має свої термінологію. Для забезпечення правильного розуміння користувачем цих понять формується глосарій. Глосарій для обраної предметної області наведено у таблиці 1.5

Таблиця 1.5

Глосарій

|  |  |
| --- | --- |
| Термін | Опис Терміну |
| 1. Основні поняття та критерії предметної області та проекту | |
| Депонування | передача коштів банківським установам задля їх зберігання на певний термін. Також сюди відноситься передача певної грошової суми (цінних паперів), яку повинен передати боржник у судові органи для подальшої передачі кредитору.  [1] |
| Стипендія | грошова виплата, яку нараховують регулярно (як правило щомісяця) учням і студентам середніх спеціальних та вищих навчальних закладів, а також аспірантам та докторантам, за  умови успішного навчання.[2] |
| Індексація | механізм який спрямований на підвищення доходу громадян, що надає можливість повністю або частково відшкодувати затрачені кошти, на споживчі товари та послуги які  підпадають подорожчанню. [3] |

Продовження таблиці 1.5

|  |  |
| --- | --- |
| Податкові пільги | бонус, що надається певним верствам населення, який  включає в себе можливість сплачувати податок в меншому розмірі або ж не сплачувати його взагалі. [4] |
| ПДФО (Податок на доходи фіз. осіб) | загальнодержавний податок, який стягується з доходів фізичних осіб (громадян — резидентів) і не резидентів, які  отримують доходи з джерел їх походження в Україні. [5] |
| 2. Користувачі системи | |
| Адміністратор | людина, функціональними зобов’язаннями якої являється  виконання усіх видів операцій, доступних у системі. |
| Головний бухгалтер | керівник відділу бухгалтерії, який забезпечує ведення  бухгалтерського обліку в установі. [6] |
| Бухгалтер | фахівець, функціональними зобов'язаностями якого є  відстеження руху коштів, нарахування зарплат та стипендій, ведення бухгалтерської звітності. [7] |
| 3. Вхідні та вихідні документи | |
| Відомість успішності | містить відмітки про успішність студента, за підсумком сесії. |
| Наказ на виплату  стипендії | містить вказівки на виплату стипендії та всі супроводжуючі  атрибути. [8] |

Опис внутрішньої роботи системи та її поведінка наведені у специфікації функціональних вимог (таблиця 1.6).

Таблиця 1.6

Специфікація функціональних вимог

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ід. вимоги | Назва  (варіанту вимоги ) | Атрибути вимог | | |
| Пріоритет | Складність | Контакт |
| 1 | Авторизація  користувача | Обов’язкове | середня | Адміністратор, головний  бухгалтер, бухгалтер |
| 2 | Додавання  даних | Обов’язкове | середня | Адміністратор, головний  бухгалтер, бухгалтер |
| 3 | Редагування  даних | Обов’язкове | низька | Адміністратор, головний  бухгалтер |
| 4 | Опрацювання  даних | Обов’язкове | низька | Адміністратор, головний  бухгалтер, бухгалтер |
| 5 | Формування  звітів | Обов’язкове | висока | Адміністратор, головний  бухгалтер, бухгалтер |

Вимоги до характеру поведінки системи визначаються нефункціональними вимогами. Нефункціональні вимоги до системи наведено у таблиці 1.7.

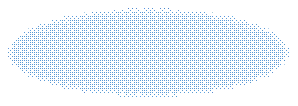
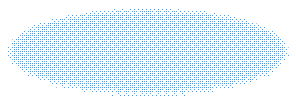
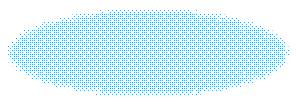
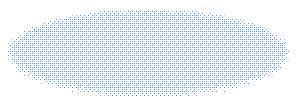
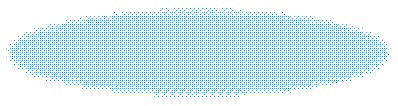
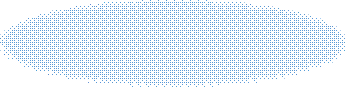
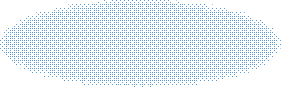
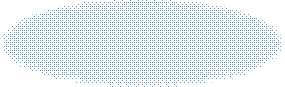
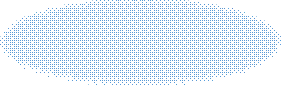
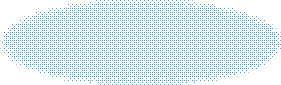
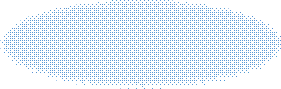
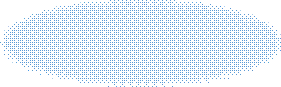
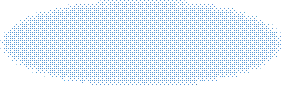
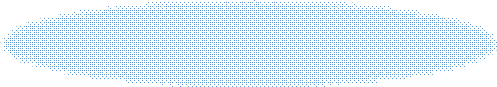
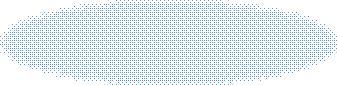
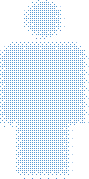
Таблиця 1.7

Специфікація нефункціональних вимог

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ід.  вимоги | Назва вимоги | Атрибути вимог | | |
| Пріоритет | Складність | Контакт |
| 1. | Застосованість | Рекомендовано | Висока | Адміністратор, головний  бухгалтер, бухгалтер |
| 2. | Надійність | Обов’ язково | Висока | Адміністратор, головний  бухгалтер, бухгалтер |
| 3. | Безпека та  захист даних | Обов’ язково | Висока | Адміністратор, головний  бухгалтер, бухгалтер |
| 4. | Відмовостійкість | Обов’ язково | Висока | Адміністратор, головний  бухгалтер, бухгалтер |

В системі є три типи користувачів: адміністратор, головний бухгалтер та бухгалтер. Для кожного із них система надаватиме різний функціонал. Спільними для усіх видів користувачів будуть функції авторизації, роботи зі студентами, формування звітності та деякі можливості роботи зі стипендією. Відношення між користувачами та варіантами використання зображено на рисунку 1.18

Перегляд



Додавання

Редагування

Видалення

<<include>> <<include>><<include>><<include>> <<include>>

Авторизація

<<extend>>

Головний Бухгалтер

Робота зі студентами

<<include>>

Перегляд

Реєстрація

<<include>>

Робота з користувачами

<<include>>

Додавання

Користувач

Адміністратор

<<include>>

Видалення

Редагування

Бухгалтер

Робота з даними про стипендію

<<include>> <<include>> <<include>>

Формування звітності

<<include>>

Нарахування Редагування

Рисунок 1.18 Діаграма варіантів використання

Розглянемо детальніше кожен із варіантів використання. Для початку роботи з системою потрібно ідентифікуватися. Це можна зробити за допомогою

пункту «Авторизація». Після успішного логування на сайті, система, відповідно до типу облікового запису, відкриє основний функціонал.

У таблиці 1.8 представлено опис варіанту використання «Авторизація».

Таблиця 1.8

Варіант використання «Авторизація»

|  |  |
| --- | --- |
| Контекст  використання | Метою функції є ідентифікація користувача та надання йому  відповідних можливостей |
| Дійові особи | Адміністратор, головний бухгалтер, бухгалтер |
| Передумова | – |
| Тригер | Перший запуск додатку |
| Сценарій | 1. – Ввід авторизаційних даних 2. – Натискання кнопки підтвердження авторизації |
| Пост умова | Відкривається основний функціонал системи / повідомлення про  помилку |

Логічний і концептуальний опис функціональних можливостей системи для сценарію «Авторизація», відображено на ескізі екранної форми, яка представлена на рисунку 1.19



Рисунок 1.19 – Кадр для сценарію «Авторизація»

Далі розглянемо варіант використання пункту «Робота зі студентами -> Перегляд». За допомогою цієї функції перед користувачем системи відкриється можливість перегляду детальної інформації про студентів.

У таблиці 1.9 представлено опис варіанту використання «Робота зі студентами -> Перегляд».

Таблиця 1.9

Варіант використання «Робота зі студентами -> Перегляд»

|  |  |
| --- | --- |
| Контекст використання | Метою функції є формування та відображення даних студента |
| Дійові особи | Адміністратор, головний бухгалтер, бухгалтер |
| Передумова | Успішна авторизація |
| Тригер | Вибір пункту в меню |
| Сценарій | 1. – Вибір пункту «Робота зі студентами» 2. – Вибір пункту «Перегляд» |
| Пост умова | Відображені дані студентів |

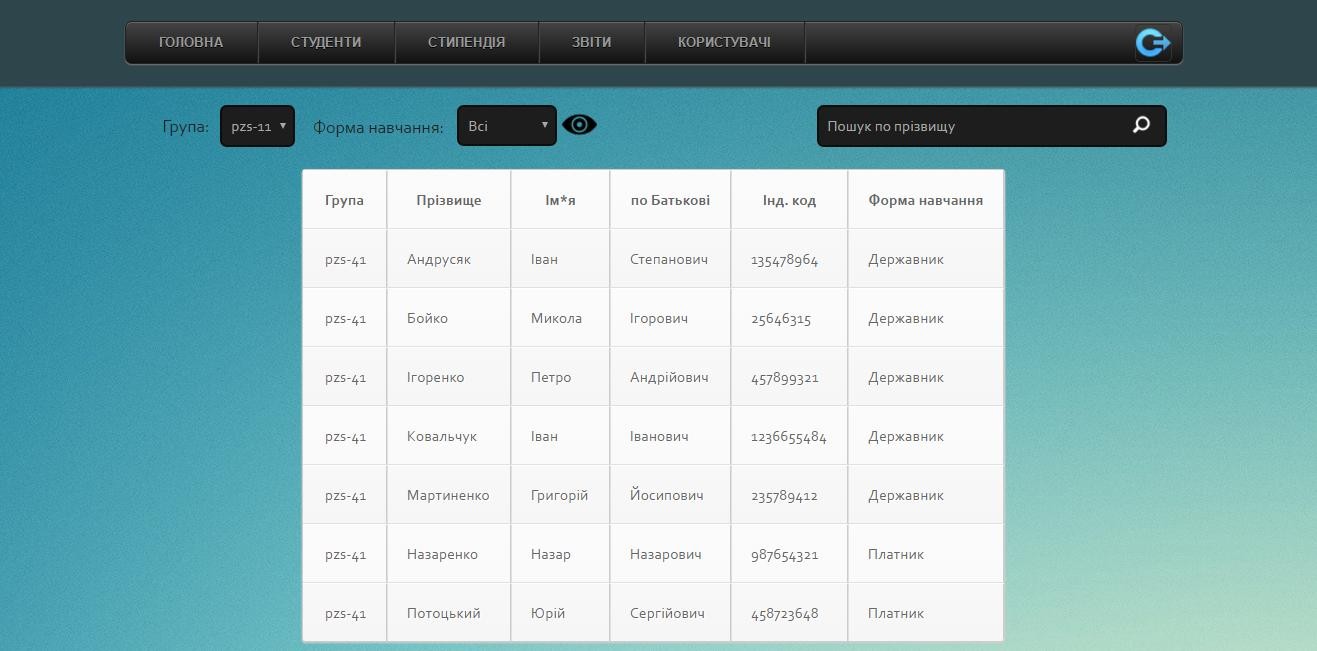
Логічний і концептуальний опис функціональних можливостей системи для сценарію «Робота зі студентами -> Перегляд», відображено на ескізі екранної форми, яка представлена на рисунку 1.20

Рисунок 1.20 – Кадр для сценарію «Робота зі студентами -> Перегляд» Далі розглянемо варіант використання пункту «Робота зі студентами ->

Редагування». За допомогою цієї функції перед користувачем системи відкриється можливість редагування даних, шляхом їх перезапису.

У таблиці 1.10 представлено опис варіанту використання «Робота зі студентами -> Редагування».

Таблиця 1.10

Варіант використання «Робота зі студентами -> Редагування»

|  |  |
| --- | --- |
| Контекст  використання | Метою функції є надання можливості редагування уже збережених  даних |
| Дійові особи | Адміністратор, головний бухгалтер, бухгалтер |
| Передумова | Успішна авторизація |
| Тригер | Вибір пункту в меню |
| Сценарій | 1. – Вибір потрібного, для внесення змін запису 2. – Введення потрібних даних 3. – Натиснення кнопки підтвердження |
| Пост умова | Повідомлення про успішне / неуспішне внесення змін |

Логічний і концептуальний опис функціональних можливостей системи для сценарію «Робота зі студентами -> Редагування», відображено на ескізі екранної форми, яка представлена на рисунку 1.21

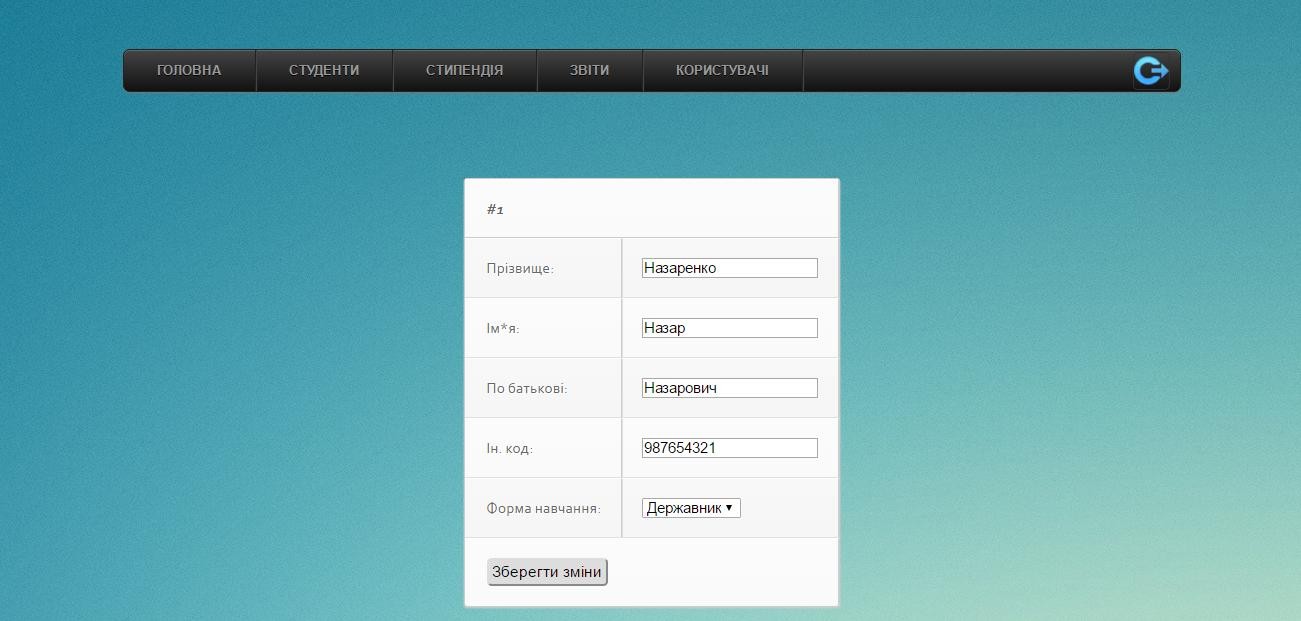


Рисунок 1.21 – Кадр для сценарію «Робота зі студентами -> Редагування»

Висновки до розділу І

В процесі виконання розділу було складено та описано організаційну структуру управління ВНЗ, визначено основні процеси, які реалізують бізнес- логіку обраної предметної області. Також визначено основні функціональні та не функціональні вимоги до веб-орієнтованої системи нарахування стипендії. Розглянуто рівні доступу користувачів до функціоналу, який буде надаватися системою.

Провівши аналіз програмних продуктів ведучих фірм-розробників ПЗ

(Infosuite, Первый БИТ та модуль «Призначення стипендії» для

«1С:Підприємства» ), можна використати деякі рішення при розробці. А саме: ведення кадрового обліку студентів, розрахунок суми стипендії, можливість ведення звітності. Проте функціонал потребує доопрацювання, оскільки немає єдиної системи, доступ до якої можливий засобами мережі Інтернет, що дозволило б працівнику не прив’язуватись до робочого місця з попередньо- встановленим ПЗ. Також ні однин із аналогів немає функції обрахунку суми стипендії відповідно до рейтингу та пільгової категорії без втручання працівника, який виконує це за допомогою ряду рутинних дій, яких можна було б уникнути. Враховуючи ці чинники буде розроблено веб-орієнтований програмний продукт, який вирішить вище-згадані проблеми.

РОЗДІЛ ІІ ПРОЕКТУВАННЯ

* 1. Розробка архітектури програмної системи

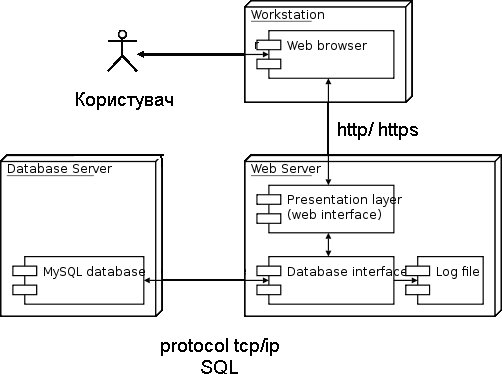
Веб додаток буде використовувати 3 основні компоненти: веб браузер, веб сервер та базу даних. На рисунку 2.1 зображено діаграму компонентів, яка відображає залежності між компонентами програмного забезпечення.

Рисунок 2.1 – Діаграма компонентів розроблюваної програмної системи

Статичне представлення структурної моделі зображають у вигляді діаграми класів. Структурна модель додатку є досить складною. Система матиме велику кількість класів, що забезпечуватимуть можливість її функціонування. Більшість модулів системи відповідатимуть за інтерфейс або стосуватимуться особливостей налагодження роботи. Тому їх відображення на даному етапі є не потрібним оскільки воно зробить діаграму класів занадто громіздкою та важкою для читання і сприйняття. Отже на рисунку 2.2 діаграма відображена лише тими класами, які забезпечуватимуть виконання логіки програми.

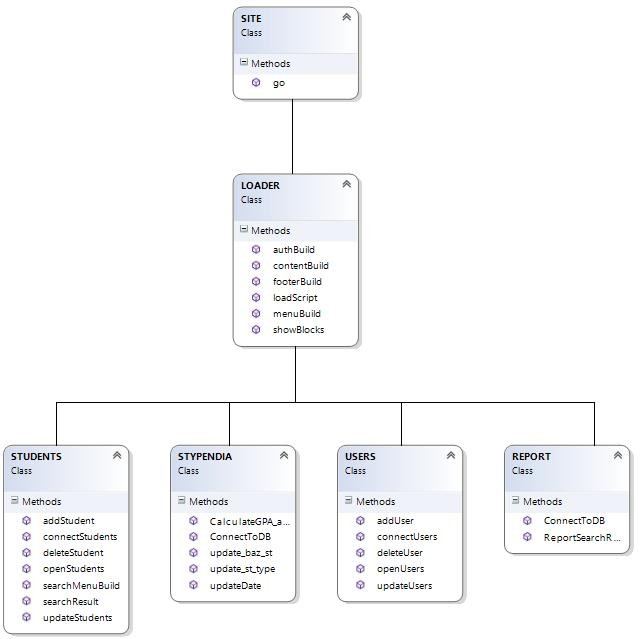


Рисунок 2.22 - Діаграма класів

На рисунку 2.3 зображена діаграма послідовності для варіанту використання «Редагування користувачів», яка відображає взаємодію об'єктів впорядкованих за часом. Можливість маніпулювання із користувачами буде доступна лише адміністратору.

Вихідний код розробки класів наведено у додатку А.

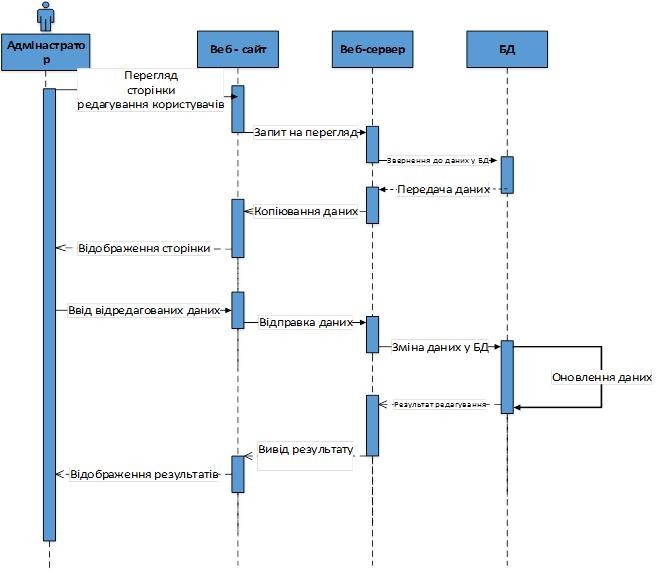


Рисунок 2.3 - Діаграма послідовності (процесу редагування користувачів)

На рисунку 2.4 зображена діаграма послідовності для варіанту використання «Додавання студента», яка відображає взаємодію об'єктів впорядкованих за часом.

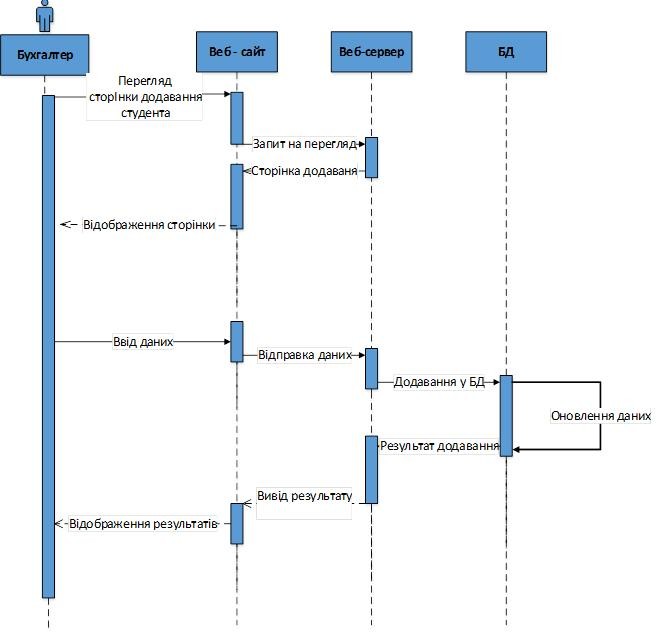
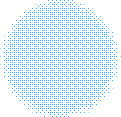


Рисунок 2.4 - Діаграма послідовності процесу додавання студента

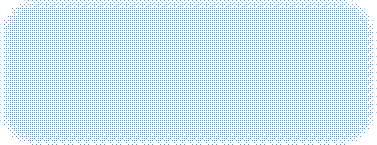
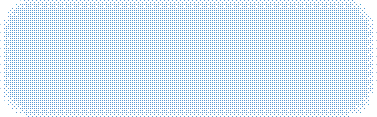
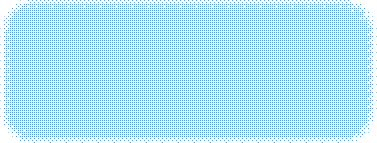
Як видно з рисунку, найдовшим процесом буде відображення результатів після внесення змін, оскільки він містить найбільше етапів.

Для моделювання поведінки об'єктів системи при переході з одного стану в інший використовують діаграми станів. Діаграма станів для функції

«Редагування» зображено на рисунку 2.5



Перехід на сторінку редагування



Очікування вводу даних

Введені дані Обробка запиту Збережені дані

Відображення

результатів

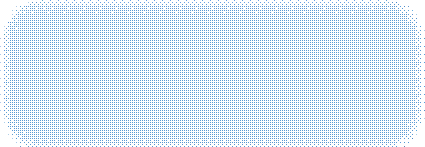
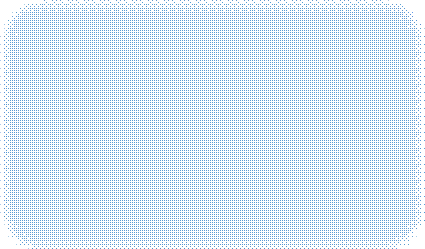
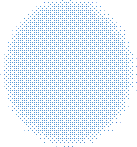
Список результатів



Рисунок 2.5 - Діаграма станів (редагування)

Як видно з рисунку, після переходу на сторінку редагування, система очікує вводу даних. Коли користувач здійснить дану операцію, введені дані обробляються, після чого зберігаються у БД. В подальшому дані оновлюються і відображаються користувачу.

На рисунку 2.6 зображено діаграму станів (видалення)



Перехід на сторінку видалення Очікування

вибору даних для видалення

Вибрані дані Обробка запиту

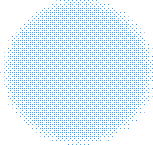
Видалені дані



Рисунок 2.6 - Діаграма станів (видалення)

Як видно з рисунку, після переходу на сторінку видалення, обраний кортеж обробляється системою. В ході виконання запиту запис видаляється.

На рисунку 2.7 зображено діаграму станів (додавання студента)

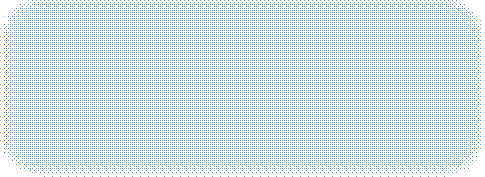
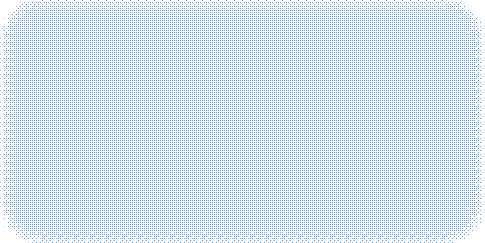


Перехід на сторінку додавання



Очікування вводу даних

Введені дані



Перевірка інформації про студетна

Збереження даних

Відображення резудьтатів

Оброблені дані

Список результатів

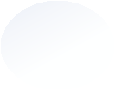


Рисунок 2.7 - Діаграма станів (Додавання студента)

Як видно з рисунку, після переходу на сторінку додавання, система очікує вводу даних. Якщо введено ідентифікаційний код студента, ідентичний тому, який вже є у БД, система повідомить про те, що запис про студента уже присутній і запропонує перевірити правильність вводу. Якщо ж всі поля заповнені коректно дані успішно будуть внесені у БД та відобразиться повідомлення, про успішно доданий запис.

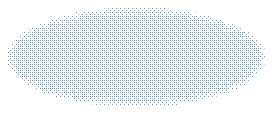
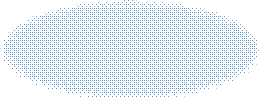
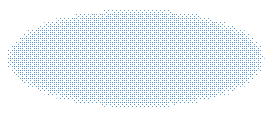
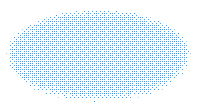
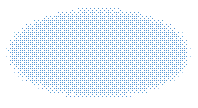
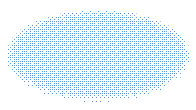
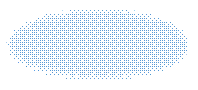
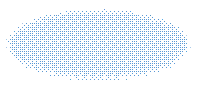
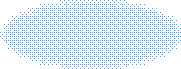
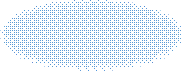
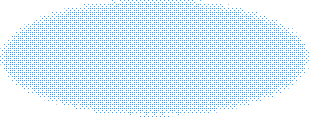
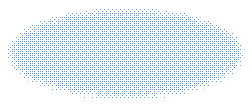
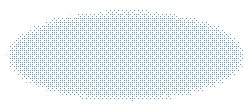
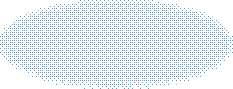
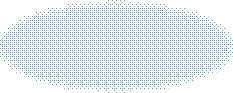
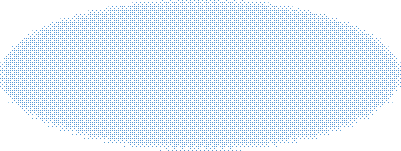
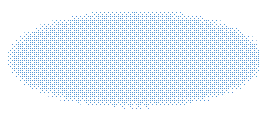
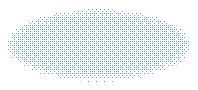
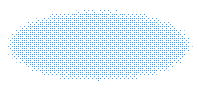
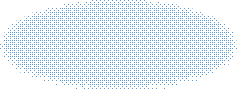
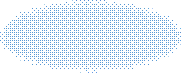
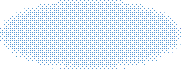
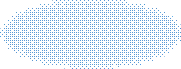
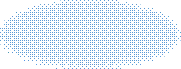
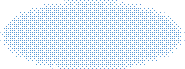
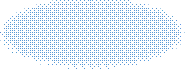
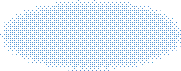
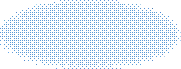
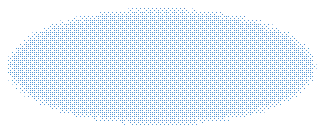
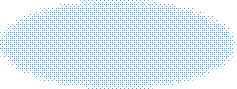
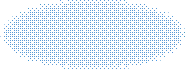
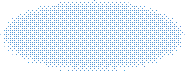
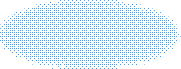
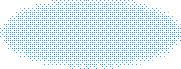
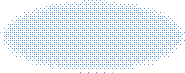
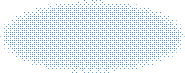
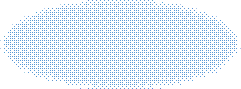
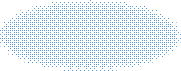
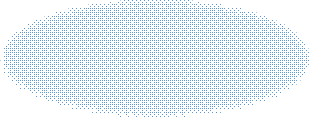
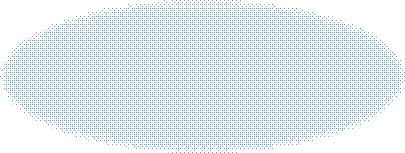
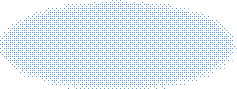
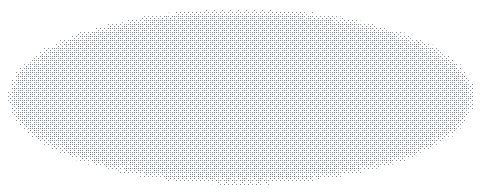
* 1. Проектування структури бази даних

Для нормальної роботи системи, потрібно забезпечити збереження великого обсягу даних, доступ до яких буде можливий лише користувачам

цього програмного продукту. Це стало можливим за допомогою баз даних та засобами їх управління, зокрема системою управління базами даних (СУБД) MySQL, яка використовується в системі.

Для реалізації програмного продукту було обрано клієнт-серверну архітектуру. У цій архітектурі запити розподілені між постачальниками послуг (серверами) та споживачами (користувачами). Фізично клієнт і сервер – це програмне забезпечення. Зазвичай вони між собою взаємодіють, за допомогою мережі, через мережеві протоколи і знаходяться на різних обчислювальних машинах. Програми – сервера, очікують від клієнтських програм запитів і надають їм свої ресурси, у вигляді даних або ж сервісних функцій.

На рисунку 2.8 наведено об’єкти та зв’язки між ними, які будуть використовуватись на етапі експлуатації системи.



користувач

група

посада

назва

тип стипендії

логін

пароль

Допуск

коефіцієнт

назва

Пільгова категорія

Рейтинг

Сума

Cтудент

назва типу стип.

персональні дані

сума

інд. код

Стипендія

сесія

Тип

Базова стипендія

прізвище

по батькові

звичайна

підвищена

предмет

ім' я

викладач

Освітній рівень

сума

оцінка

термін дії

ПІБ

максимальний поріг

середній бал

допуск

назва

початок

кінець

назва

мінімальний поріг

5тибальне знач.

Рисунок 2.8 – Діаграма елементів і зв'язків

При високорівневому (концептуальному) проектуванні баз даних використовують ER-модель[9]. Під час проектування БД відбувається перетворення ER-моделі в конкретну схему бази даних на основі обраної моделі даних. На рисунку 2.9 зображено модель ERD для розроблюваної системи.

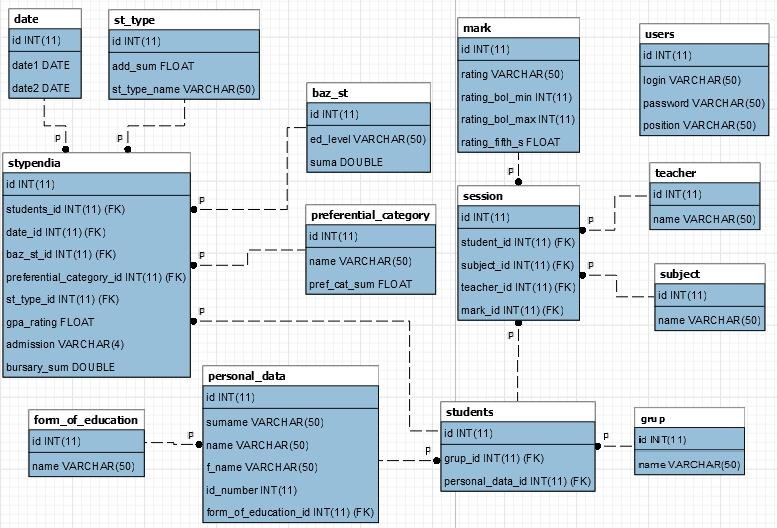


Рисунок 2.9 – Модель сутність-зв’язок (ERD)

Усі відношення автоматично індексуються, по первинному ключі. Завдяки чому значно пришвидшується процес пошуку даних у БД. У розроблюваній системі створено foreign keys для наступних атрибутів:

1. students\_id;
2. date\_id;
3. baz\_st\_id;
4. preferential\_category\_id;
5. st\_type\_id;
6. form\_of\_education\_id;
7. grup\_id;
8. personal\_data\_id;
9. stuent\_id;
10. subject\_id;
11. teacher\_id;
12. mark\_id.

Результатом фізичного проектування є DDL код, який надано в додатку Б.

Висновки до розділу ІІ

В ході виконання розділу було розглянуто детальну модель веб- орієнтованої системи нарахування стипендії та основні функції, що забезпечуватимуть чітке виконання вимог до системи. Визначено зв’язки між компонентами системи та їх взаємодію. Обрано архітектуру для реалізації веб- орієнтованої системи нарахування стипендії. Розглянуто основні класи, за допомогою яких буде можливим функціонування системи та засоби які будуть реалізовувати бізнес-логіку предметної області. Окрім цього спроектовано схему бази даних та обрано поля для пошуку.

РОЗДІЛ III

ПРОГРАМНА РЕАЛІЗАЦІЯ

* 1. Програмна реалізація проекту

*Обґрунтування вибору мови програмування*

Для реалізації веб-орієнтованої програмної системи нарахування стипендії було обрано мову програмування PHP[10].

PHP (Hypertext PreProcessor) – широко-застосовувана мова опису сценаріїв загального призначення із відкритим вихідним кодом. Інакше кажучи, PHP – це мова програмування яка призначена для написання веб-додатків, які будуть виконуватись на веб-сервері. Невід’ємною перевагою PHP є надання можливості веб-розробнику швидкого створення динамічних веб-сторінок. За допомогою цієї мови програмування можна створювати якісні веб-додатки за короткі проміжки часу, які будуть підтримуватись у майбутньому і будуть мати можливість до швидкої модифікації. Окрім цього за допомогою PHP можна досить просто організувати комунікацію із БД. PHP не викликає труднощів у вивченні і водночас здатна задовільнити запити професійних програмістів.

До основних аспектів, які сприяли вибору мови програмування належать:

* легкість у вивченні;
* безкоштовне поширення із відкритим вихідним кодом;
* простота у комунікації із БД;
* добре організована система комунікації із серверами;
* висока швидкість обробки сценаріїв;
* надійно реалізований механізм безпеки;
* гнучкість у використанні;

*Організація інтерфейсу з користувачем*

При проектуванні інтерфейсу системи основна увага приділялася простоті та легкості сприйняття. Завдяки цьому у користувача який буде вперше працювати із системою не виникне ніяких труднощів, оскільки все буде зрозуміло на інтуїтивному рівні.

В залежності від займаної посади користувачу системи буде відкрито доступ до тих функцій, виконання яких його зобов’язують посадові обов’язки. Інтерфейс головної сторінки системи містить у собі такі блоки:

* Menu:
* Header
* Content
* Footer

Для забезпечення зручності і досягнення максимальної швидкодії в роботі було вирішено викристати AJAX[11]. AJAX (Asynchronous JavaScript And XML) - підхід до побудови інтерфейсу, в розгортанні роботи якого веб- сторінка без перезавантаження відсилає певні запити на сервер і повертає потрібні дані. Тобто блоки: menu, header та footer залишаються статичними, а в блок content завантажуються дані, відповідно до обраної користувачем функції. Лістинг коду AJAX наведено нижче.

<script>

function showContent(link) {

var cont = document.getElementById('contentBody'); var loading = document.getElementById('loading'); cont.innerHTML = loading.innerHTML;

var http = createRequestObject(); if( http ) {

http.open('get', link); http.onreadystatechange = function () { if(http.readyState == 4) { cont.innerHTML = http.responseText;

}

}

http.send(null);

} else { document.location = link;

}

}

function createRequestObject() {

try { return new XMLHttpRequest() } catch(e) {

try { return new ActiveXObject('Msxml2.XMLHTTP') } catch(e) {

try { return new ActiveXObject('Microsoft.XMLHTTP') } catch(e) { return null; }

}

}

}

</script>

Коли в блок content не завантажено великої кількості даних, основне меню залишається без змін. В іншій ситуації, коли в цьому блоці знаходиться великий об’єм даних і є потреба в прогортанні сторінки вниз – блок menu залишається у верхній частині сайту. Це забезпечує додаткову користувацьку свободу, оскільки з любого місця можна повернутися на головну сторінку чи викликати будь-яку іншу функцію. Лістинг коду цієї функції наведено нижче.

$(document).ready(function(){

$("#header").removeClass("default");

$(window).scroll(function(){ if ($(this).scrollTop() > 20) {

$("#header").addClass("default").fadeIn('fast');

} else {

$("#header").removeClass("default").fadeIn('fast');

};

});

$('a[href^="#"]').click(function () { elementClick = $(this).attr("href");

destination = $(elementClick).offset().top; if($.browser.safari){

$('body').animate( { scrollTop: destination }, 1000 );

} else {

$('html').animate( { scrollTop: destination }, 1000 );

}

return false;

});

});

*Вимоги до апаратного забезпечення*

* + Персональний комп’ютер;
  + Встановлений інтернет-браузер;
  + Доступ до мережі інтернет;

*Опис класів*

При проектуванні веб-орієнтованої системи нарахування стипендії використовувався об’єктно орієнтований підхід. Суть цього підходу полягає у представленні програми у вигляді об’єктів, які певним чином взаємодіють між собою. Об’єктом являється екземпляр класу, а клас – певний спосіб опису сутностей, який безпосередньо визначає правила для взаємодії, поведінку та стан.

В процесі розробки програмної системи було реалізовано такі класи:

* Site
* Loader
* Students
* Stypendia
* Users
* Report

За допомогою класу Site, методу go() відбувається процес ідентифікації користувача та надання йому відвідного рівня доступу до системи. Якщо користувача із введеними даними не зареєстровано у системі – в доступі до основного функціоналу буде відмовлено.

За допомогою класу Loader відбувається побудова блоків сайту та механізм їх завантаження. Цей клас містить наступні методи:

* loadScript();
* showBlocks();
* authBuild();
* menuBuild();
* contentBuild();
* footerBuild();

За допомогою класу Students можна виконувати перегляд списку студентів, додавання нового студента, редагування даних студента, та видалення запису із БД. Окрім цього є можливість формування списків студентів за певними критеріями, а саме: група та форма навчання. Крім цього реалізований також пошук по прізвищу. Цей клас містить наступні методи:

* openStudents();
* searchMenuBuild();
* searchResult();
* addStudent();
* updateStudents();
* deleteStudent();

За допомогою класу Stypendia можна обрахувати середній бал студентів та призначити стипендію відповідно до певних критеріїв, а саме: можливості отримувати пільгову та підвищену стипендію, редагувати базові відомості про стипендію (дату дії, розмір підвищеної стипендії, розмір звичайної стипендії, відповідно до ОКР). Цей клас містить наступні методи:

* CalculateGPA\_and\_bursarySum();
* updateDate();
* update\_st\_type();
* update\_baz\_st();

Вихідний код, який реалізовується в методі CalculateGPAandBursarySum

зображено нижче.

function CalculateGPA\_and\_bursarySum($load\_data\_query,$max\_students\_id\_query,$update\_GPA\_query)

{

$max\_students\_id\_query =(" SELECT MAX(students\_id) as m\_s\_id FROM stypendia ");

$res = mysql\_query($load\_data\_query) or die(mysql\_error());

$max\_student\_id = mysql\_query($max\_students\_id\_query) or die(mysql\_error()); while ($m\_s\_id\_arr = mysql\_fetch\_array($max\_student\_id)) {

echo "<tr>\n";

$m\_s\_id = $m\_s\_id\_arr['m\_s\_id']; for ($i=1; $i<=$m\_s\_id;$i++){

$update\_GPA\_query =("UPDATE stypendia SET gpa\_rating = (

Select avg(mark.rating\_fifth\_s) FROM mark

INNER JOIN `session` ON `session`.mark\_id = mark.id WHERE `session`.student\_id = '$i'

)

WHERE stypendia.students\_id ='$i' ");

$res3 = mysql\_query($update\_GPA\_query) or die(mysql\_error());

}

}

echo ('

<table class="simple-little-table" border="1" cellspacing="0">

<tr style=\"border: solid 1px #000\">

<td align="center"><b>Група</b></td>

<td align="center"><b>Прізвище</b></td>

<td align="center"><b>Ім\*я</b></td>

<td align="center"><b>По батькові</b></td>

<td align="center"><b>Пільгова категорія</b></td>

<td align="center"><b>Середній бал</b></td>

<td align="center"><b>Допуск</b></td>

<td align="center"><b>Сума, грн</b></td>

<a id ="1">

</tr> ');

while ($row = mysql\_fetch\_array($res)) { echo "<tr>\n";

echo "<td>".$row['grup']."</td>\n"; echo "<td>".$row['surname']."</td>\n";

echo "<td>".$row['student\_name']."</td>\n"; echo "<td>".$row['f\_name']."</td>\n";

echo "<td>".$row['preferential\_category']."</td>\n"; echo "<td>".$row['gpa\_rating']."</td>\n";

echo "<td>".$row['admission']."</td>\n"; echo "<td>".$row['bursary\_sum']."</td>\n";

}

echo ("</table>\n </a>");

$res = mysql\_query('CALL `calculate\_stypendia`()') or die(mysql\_error());

$max\_student\_id2 = mysql\_query($max\_students\_id\_query) or die(mysql\_error()); while ($m\_s\_id\_arr2 = mysql\_fetch\_array($max\_student\_id2)) {

$m\_s\_id2 = $m\_s\_id\_arr2['m\_s\_id']; for ($j=1; $j<=$m\_s\_id2;$j++){

$update\_bursary\_sum\_query =("UPDATE stypendia SET bursary\_sum = (

SELECT `sum` FROM (

SELECT suma + pref\_cat\_sum + add\_sum as `sum` FROM stypendia

INNER JOIN baz\_st on baz\_st.id = stypendia.baz\_st\_id

INNER JOIN preferential\_category on preferential\_category.id = stypendia.preferential\_category\_id

INNER JOIN st\_type on st\_type.id = stypendia.st\_type\_id WHERE stypendia.students\_id = '$j'

)

AS ResultSet

)

WHERE stypendia.students\_id ='$j' AND stypendia.admission = 'yes' ");

$resultat = mysql\_query($update\_bursary\_sum\_query) or die(mysql\_error());

}

}

mysql\_close();

}

За допомогою класу Users можна переглянути список зареєстрованих користувачів системи, редагувати дані користувачів, додати нового та видалити уже існуючого користувача системи. Цей клас містить наступні методи:

* openUsers();
* addUser();
* updateUsers();
* deleteUser();

Клас Report дозволяє сформувати наказ на стипендію та переглянути звітність по стипендії. Є можливість формування списку по певним критеріям (група, наявність пільгової категорії, рейтинг). Окрім цього вкінці списку буде вказано кількість результатів, які задовольняють вище-згадані критерії, загальний середній рейтинг по цих результатах та загальна потрачена сума. Цей клас реалізує метод ReportSearchResult().

* 1. Програмна реалізація бази даних

Для комунікації із системою було обрано СУБД MySQL[12]. На сьогоднішній день MySQL набула широкого застосування і підтримується великою кількістю мов програмування. Основним застосунком якої є керування реляційними БД (тих баз даних, у яких дані зберігаються в певних таблицях завдяки чому забезпечується максимальна швидкість та гнучкість). Середовищем розробки і адміністрування БД MySQL було обрано MySQL Workbench, оскільки він є безкоштовним та зручним у використанні.

До основних переваг MySQL Workbench можна віднести:

* можливість редагування даних в таблиці, а також можливість легко представити графічну модель БД;
* наявність налагодженого і простого в застосуванні механізму створення зв’язків між таблицями;
* наявність простого редактора SQL-запитів, в результаті виконання яких отримаємо таблицю із відповідними даними;

Основним інструментом для операцій з таблицями використано SQL-

запити[13]. Існує 4 типи SQL-запитів, а саме:

* вибірка (SELECT *назва\_колонки\_1*, *назва\_колонки\_2* FROM *назва\_таблиці* );
* вставка (INSERT INTO *назва\_таблиці* (*назва\_колонки\_1*, *назва\_колонки\_2*, *назва\_колонки\_3*,...) VALUES (*значення\_1*, *значення\_2*, *значення\_3*,...));
* редагування (UPDATE *назва\_таблиці* SET *назва\_колонки\_* = *значення\_1*,

*назва\_колонки\_2* = *значення\_2*,... WHERE *деяка\_колонка* =

*деяке\_значення*);

* видалення (DELETE FROM *назва\_таблиці* WHERE *деяка\_колонка* = *деяке\_значення*); Для реалізації бізнес-логіки процесу нарахування стипендії в методі CalculateGPAandBursarySum() класу Stypendia відбувається виклик збереженої процедури під назвою calculate\_stypendia. Вихідний код stored-процедури[14]

наведено нижче.

CREATE DEFINER=`root`@`%` PROCEDURE `calculate\_stypendia`() LANGUAGE SQL

NOT DETERMINISTIC MODIFIES SQL DATA SQL SECURITY DEFINER COMMENT ''

BEGIN

UPDATE stypendia

SET stypendia.st\_type\_id = (SELECT id from st\_type where st\_type.st\_type\_name ='Звичайна')

WHERE stypendia.gpa\_rating >=4 and stypendia.gpa\_rating<5; UPDATE stypendia

SET stypendia.st\_type\_id = (SELECT id from st\_type where st\_type.st\_type\_name ='Підвищена')

WHERE stypendia.gpa\_rating =5; UPDATE stypendia

SET stypendia.st\_type\_id = (SELECT id from st\_type where st\_type.st\_type\_name ='Звичайна')

WHERE stypendia.gpa\_rating >=3 and stypendia.gpa\_rating<4 and stypendia.preferential\_category\_id != (SELECT id from preferential\_category WHERE preferential\_category.name='Немає'); UPDATE stypendia

SET stypendia.admission = 'no' WHERE stypendia.gpa\_rating <4; UPDATE stypendia

SET stypendia.admission = 'yes' WHERE stypendia.gpa\_rating >=4; UPDATE stypendia

SET stypendia.admission = 'yes'

WHERE stypendia.preferential\_category\_id != (SELECT id from preferential\_category WHERE preferential\_category.name='Немає'); UPDATE stypendia

SET stypendia.admission = 'no' WHERE stypendia.gpa\_rating <3; UPDATE stypendia

SET stypendia.bursary\_sum = 0 WHERE stypendia.admission ='no'; END

Висновки до розділу ІІІ

В процесі виконання розділу було обґрунтовано вибір мови програмування яка використовується, розглянуто основні аспекти проектування інтерфейсу з користувачем. Обрано підхід, за допомогою якого користувацький інтерфейс досягатиме максимальної швидкодії в роботі. Визначено вимоги до апаратного забезпечення. Проаналізовано основні класи, які використані при проектуванні веб-орієнтованої системи нарахування стипендії та наведено приклад їх програмної реалізації. Також обрано основні інструменти, необхідні для комунікації із системою та наведено їхні основні переваги.

РОЗДІЛ IV

ТЕСТУВАННЯ ТА ДОСЛІДНА ЕКСПЛУАТАЦІЯ

* 1. Тестування

Тестування – завершальний етап у розробці будь-якого програмного продукту, який відіграє дуже важливу роль в створенні якісного ПЗ. З рівнем складності програми зростає і рівень складності її тестування. Основною метою тестування є класифікація рівня якості програмного продукту. Ця задача реалізовується за допомогою двох основних чинників, а саме:

* демонстрація учасникам проекту відповідності програми до поставлених вимог;
* визначення факторів, в ході яких певні функції системи різняться від тих, що прописані в специфікації.

Практично неможливо зразу ж написати ідеальний програмний продукт який б не містив у собі ніяких помилок. І як буває у більшості випадків, винуватцями появи помилок у розроблюваному програмному продукті є самі програмісти. Отож задачею кваліфікованого програміста є не лише написання програми, а й відстеження в ній помилок. Виходячи з цього тестування веб- орієнтованої системи нарахування стипендії є не лише доцільним, а й необхідним етапом розроби.

Під час проведення тестування програмного продукту були використані такі види тестування[15]:

* функціональне тестування;
* тестування сумісності;
* тестування продуктивності;
* тестування GUI;
* тестування безпеки.

*Функціональне тестування*

В розгортанні функціонального тестування перевірятимуться усі функції системи на предмет відмінності між реальною поведінкою і очікуваною, відповідно до специфікації.

Використано як ручне тестування так і автоматизоване. Інструментом для реалізації автоматизованого тестування слугував плагін для веб-браузера Mozilla Firefox під назвою Selenium[16].

На рисунку 4.1 зображено результат тестування y Selenium функції додавання студента.

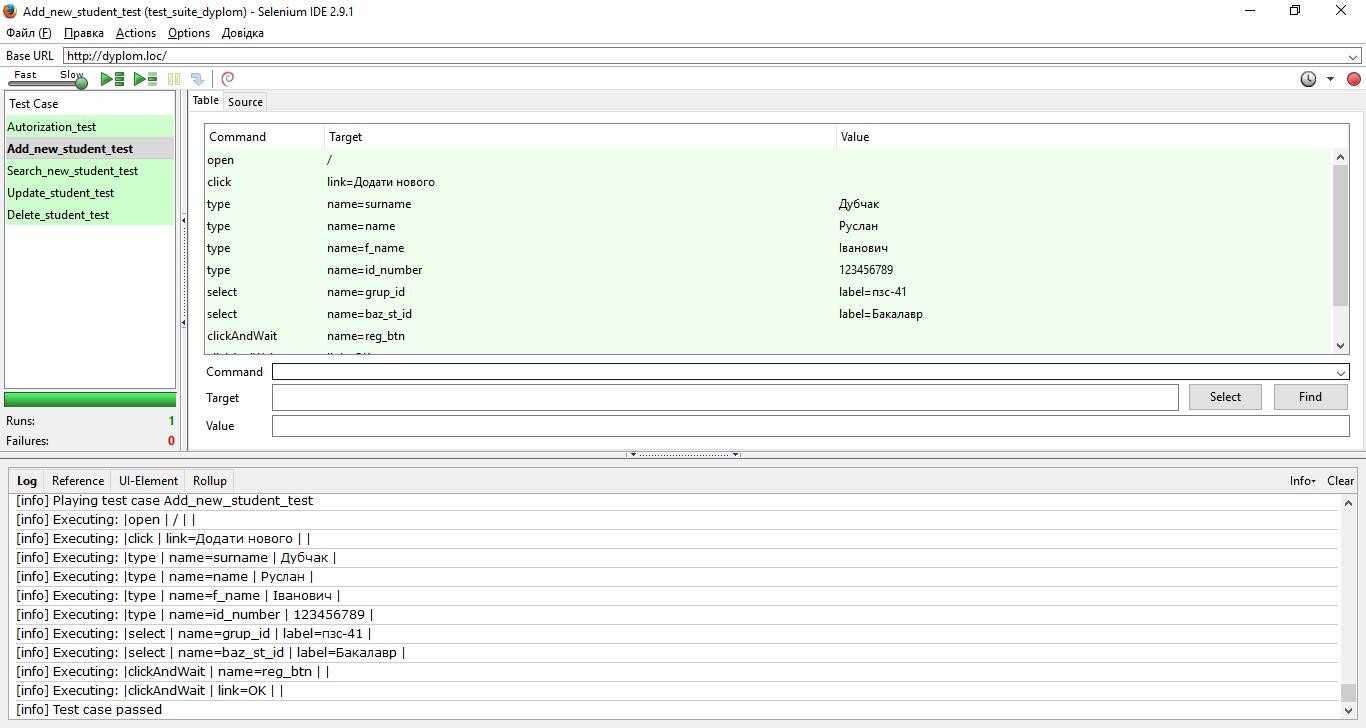


Рисунок 4.1 – Тестування функції додавання студента у Selenium

Перелік, вихідний код та логи тест-кейсів[17] наведено у додатку В.

В ході розробки системи та її тестування було налаштовано та наповнено тестову бузу даних.

Під час розробки тестових випадків було спроектовано тести для функціонального тестування, тестування безпеки. Кожен тестовий випадок містить детальні кроки, тестові дані і очікувані результати. Спроектовано 10 функціональних тестових випадків. Результат наведено у додатку Г.

В таблиці 4.1 наведено результати розподілу функціональних тестових випадків та наборів текстових даних для них за варіантами використання, на прикладі:

* варіанту використання «авторизація»;
* варіанту використання «додавання студента»;
* варіанту використання «перегляд списку студентів»;
* варіанту використання «пошук студента»;
* варіанту використання «редагування даних студента»;
* варіанту використання «видалення даних студента»

Таблиця 4.1

Розподіл функціональних тестових випадків та наборів текстових даних для цих випадків за варіантами використання

|  |  |  |
| --- | --- | --- |
| Варіанти використання | Тестові випадки | Тестові дані |
| Check Authorization | 3 | 3 |
| Check option “Add\_student” | 5 | 5 |
| Check option “view\_students” | 5 | 5 |
| Check option “view students\_  search\_result” | 3 | 3 |
| Check option “update\_students” | 5 | 5 |
| Check option “delete\_student” | 5 | 5 |
| Загалом | 26 | 26 |

*Тестування сумісності*

Тестування сумісності – вид тестування метою якого являється перевірка коректності роботи програмного продукту в різних середовищах, в нашому випадку середовищем виступає веб-браузер. Завдяки цьому тесту отримаємо змогу відслідкувати і виправити недоліки системи задля правильного функціонування у всіх браузерах.

Перевірка сумісності буде проводитись у таких браузерах:

* Google Chrome;
* Opera;
* Mozilla Firefox;
* Internet Explorer;
* Microsoft Edge;

Підсумок тестування: В результаті тестування пройшли успішно 99 зі 100 множин тестових даних, отже тестування на сумісність розглядається як частково успішне адже 99% тестових випадків пройшли.

Відомі дефекти:

Дефект: В браузері Mozilla Firefox при перегляді сторінки зі звітністю, блок вибору критеріїв пошуку змістився вліво.

Опис: при перегляді сторінок в різних браузерах інтерфейс системи повинен залишатись однаковим, допускається лише мінімальне відхилення.

Подолання дефекту: Такого роду дефект можна виправити, змінивши тип позиціонування блоку з “align = center” на певне значення px відступу зліва.

*Тестування продуктивності*

Основними показниками продуктивності програмного продукту є час з’єднання з сервером та час відповіді сервера. Чим менше значення цього показника тим краще, адже запити будуть швидко оброблятись і виконуватись.

Ці показники можна дізнатись за допомогою тестування продуктивності, в процесі виконання якого, шляхом навантаження буде визначено їх швидкодію.

Для перевірки продуктивності системи використано такі види тестування продуктивності:

* тестування стабільності;
* конфігураційне тестування;
* навантажувальне тестування;
* стрес-тестування.

Підсумок тестування: В результаті тестування 10 із 10 тестів виявились успішними. Час з’єднання з сервером – 0,12 с., час відповіді сервера – 0,23 с. Відповідно, виходячи із отриманих результатів, тестування можна вважати успішним.

*Тестування GUI*

Тестування GUI (graphical user interface) [18] – вид тестування, метою якого являється визначення степені зручності певного об’єкта (користувацького інтерфейсу, веб-сторінки) для його безпосереднього застосування. Інакше кажучи - це перевірка на ергономічність. Тобто свого роду оцінка зручності використання програми шляхом залучення звичайних користувачів в якості тестувальників і аналіз їх висновків.

В ході тестування GUI було залучено 5 користувачів, яким запропоновано протестувати інтерфейс системи.

Підсумок тестування: 5 з 5 користувачів, які займались тестуванням GUI одноголосно стверджували про те, що інтерфейс системи є легким для сприйняття та зрозумілим на інтуїтивному рівні. Виходячи з цього можна зробити висновок, що тестування GUI веб-орієнтованої системи нарахування стипендії виявилось вдалим.

*Тестування безпеки*

Тестування безпеки – вид тестування метою якого являється визначення ступеня захисту даних програмного продукту при збереженні функціональності.

Основні аспекти безпеки, з яких складається тестування:

* конфіденційність;
* цілісність;
* аутентифікація;
* доступність даних;
* безвідмовність.

Підсумок тестування:

В результаті тестування 10 із 10 тестів виявились успішними. Доступ до функціоналу системи сайту можливий після проходження процесу авторизації в результаті проходження якого відкриється функціонал, відповідно до типу облікового запису. Паролі для доступу в систему зберігаються під MD5- кодуванням. Завдяки цьому забезпечується додатковий рівень безпеки, оскільки вгадати чи декодувати пароль під цим типом шифрування практично неможливо. Виходячи з отриманих даних можна зробити висновок, що тестування безпеки пройшло успішно.

* 1. Розгортання програмного продукту

*Щоб мати змогу користуватися веб-орієнтованою системою нарахування стипендії її потрібно попередньо розмістити на сервері. Список вимог, дотримання яких необхідно для перетворення звичайного ПК в сервер:*

* Статичний IP[19];
* Мінімальна швидкість доступу до мережі інтернет 10Mbps;
* Рекомендовано встановлення Unix-системи;
* Встановлення і конфігурація Apache-сервера, PHP, MySQL, PHPMyAdmin

та FTP-сервера ;

* Відкриття 80-го порта на вхід в Firewall[20];
* Постійно включений комп’ ютер
  1. Інструкція користувача

Для відкриття основного функціоналу системи потрібно авторизуватися. Цей процес передбачає введення логіна та пароля і натискання на кнопку

«Вхід».

На рисунку 4.2 зображено сторінку авторизації.

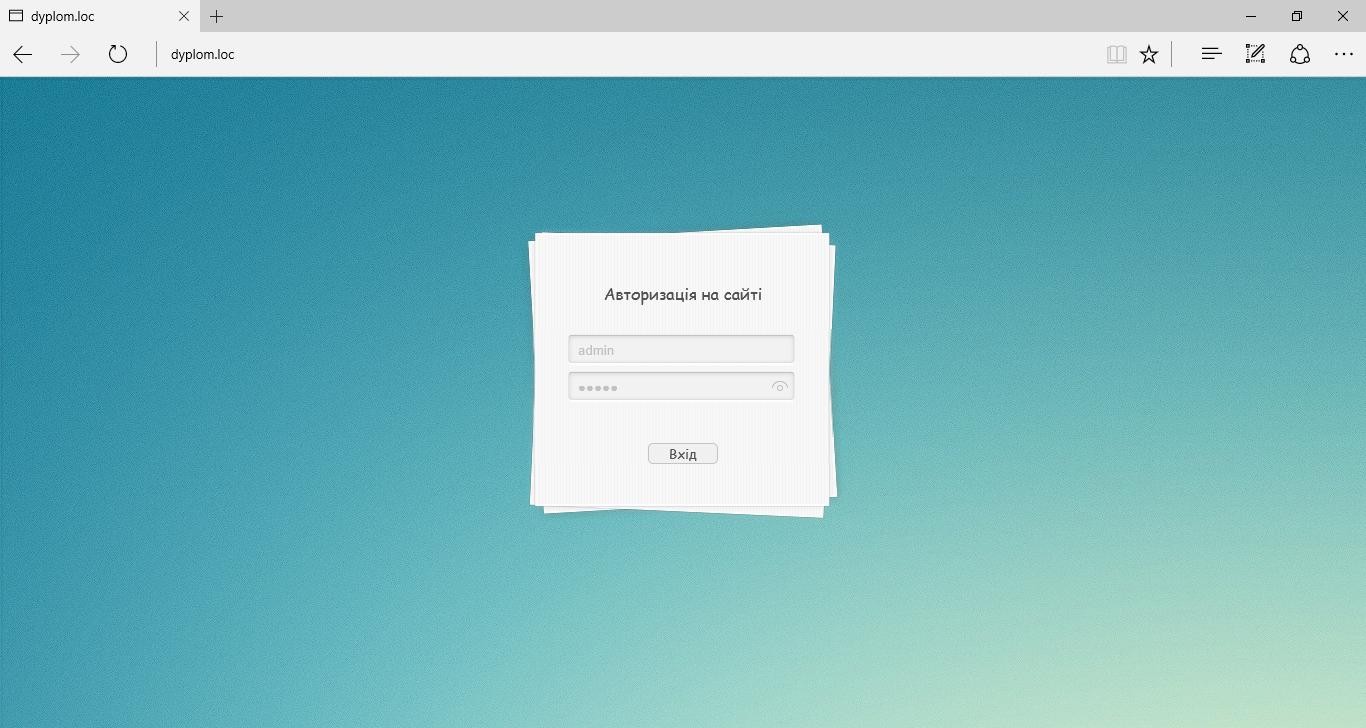


Рисунок 4.2 – Авторизація в системі

Після введення коректних даних, система відповідно до рівня доступу користувача відкриє основний функціонал. Також відповідні дані запишуться у змінні сесії, за допомогою яких при відкритті сторінок системи в інших вікнах браузера користувач буде залишатися залогованим у системі. На рисунку 4.3 зображено головну сторінку сайту.

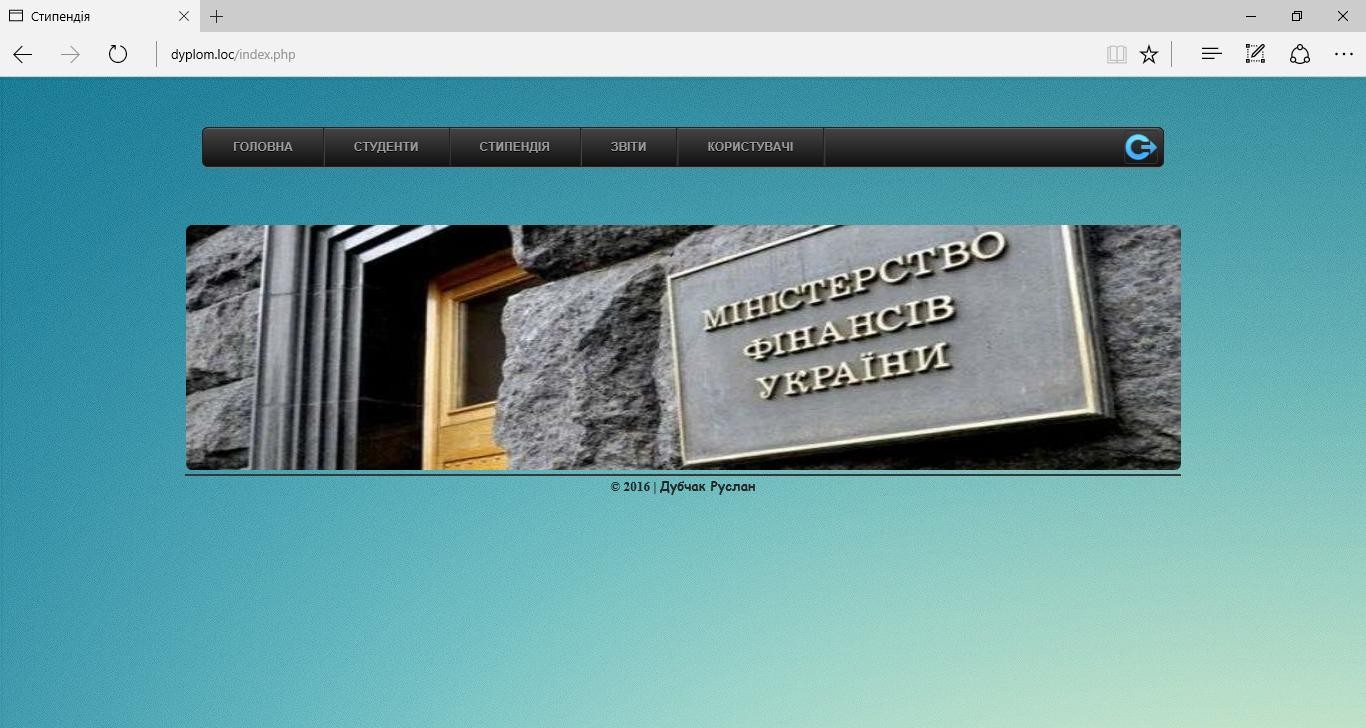


Рисунок 4.3 – Головна сторінка сайту

Для того, щоб обрати певну функцію системи використовується навігаційне меню. Вибір функції на прикладі вкладки «Стипендія -> Редагувати» зображено на рисунку 4.4.

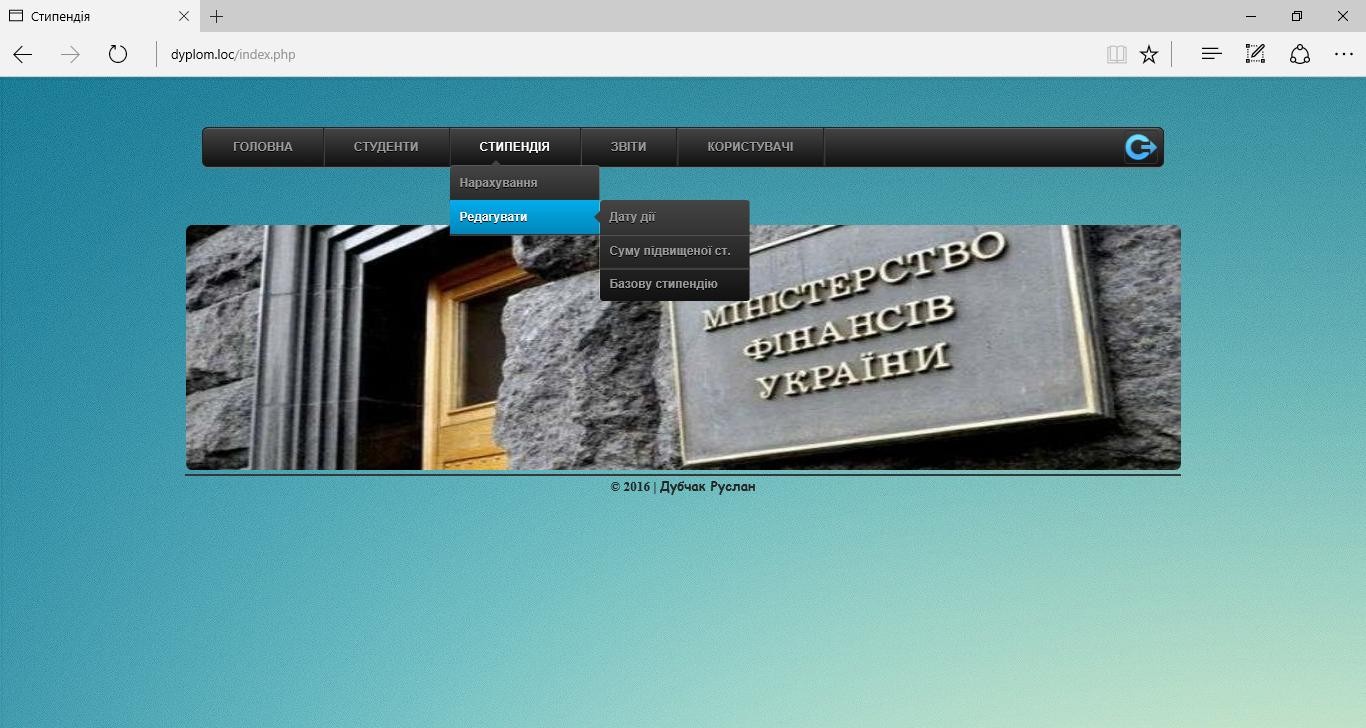


Рисунок 4.4 - Вибір функції «Стипендія -> Редагувати»

Як видно з рисунку 4.4 - наведений вказівник викликає третій рівень меню, за допомогою якого можна редагувати базові стипендійні відомості. Це досить зручно, оскільки редагування базових стипендійних відомостей можна реалізувати працівнику засобами системи, без потреби втручання фахівця і змін вихідного коду. На екранну форму завантажуються відповідні дані із БД, надаючи можливість користувачу системи переглянути теперішні дані та внести зміни (див. рис. 4.5). Після натискання на кнопку «Зберегти зміни» дані успішно перезапишуться у БД. Після цього користувачеві буде повідомлено про успішно здійснену операцію (див.рис. 4.6).

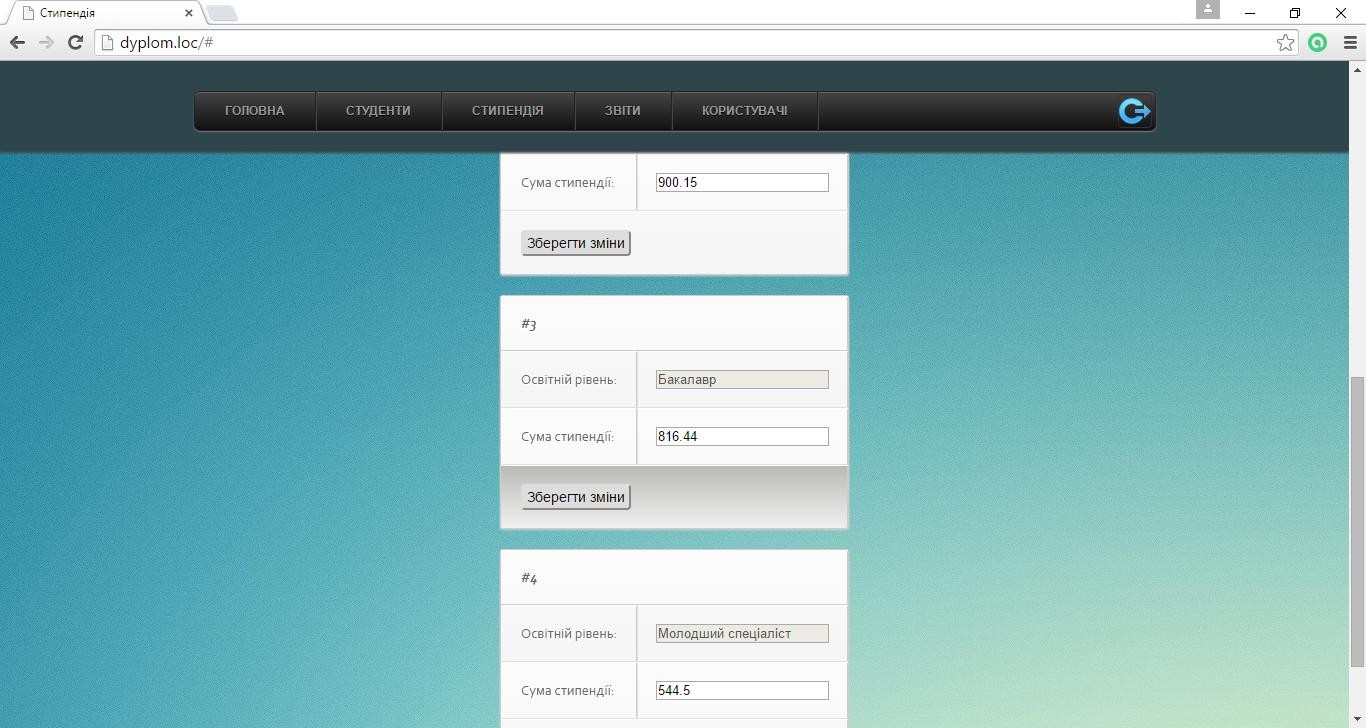


Рисунок 4.5 – Редагування базової стипендії

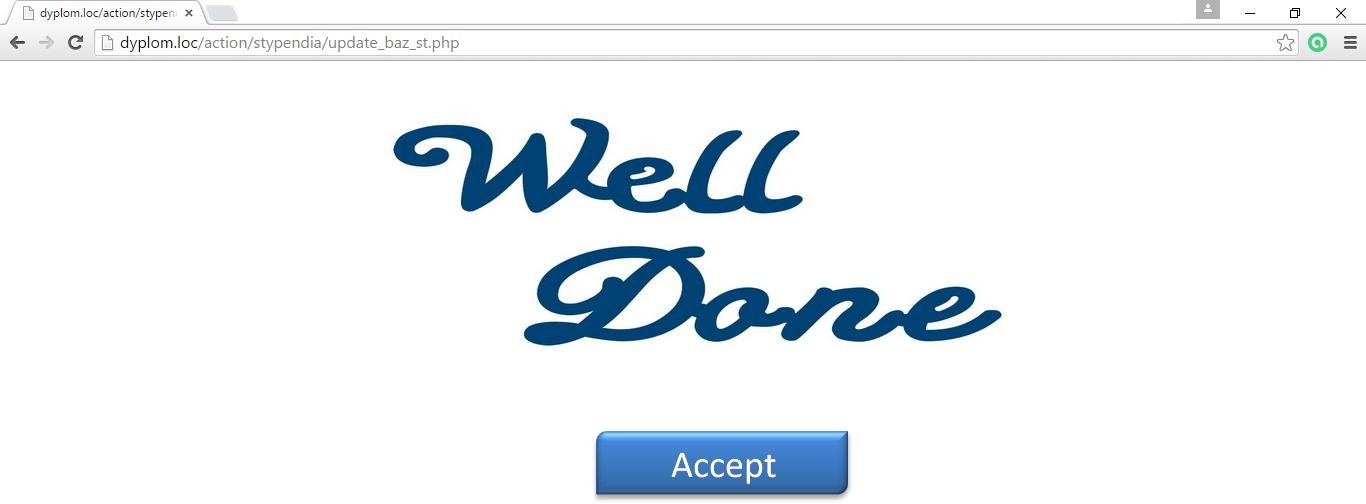


Рисунок 4.6 – Успішно виконана операція

Для обрахунку стипендійних даних необхідно обрати пункт «Стипендія -

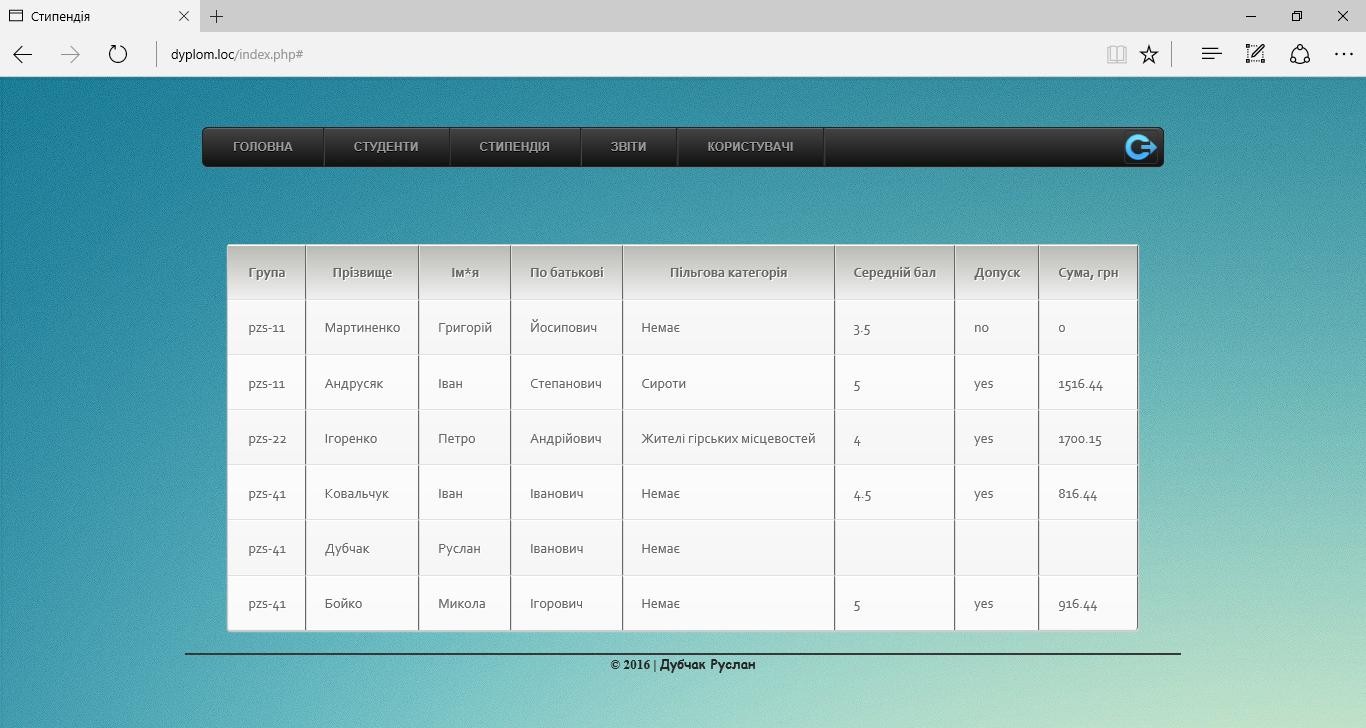
> Нарахування». Відповідно до результатів заліково-екзаменаційної сесії формується рейтинг та надається допуск для отримання стипендії, відповідно до цього рейтингу та наявності пільгової категорії. Результат цієї дії зображено на рисунку 4.7.

Рисунок 4.7 – Результат виконання функції «Стипендія -> Нарахування» Для того, щоб сформувати наказ на стипендію потрібно обрати пункт

«Звіти -> Cформувати наказ». На основі раніше обрахованих даних формується список стипендіатів та відомостей про них. Результатом дії цієї функції буде сформований для друку наказ, який зображено на рисунку 4.8.

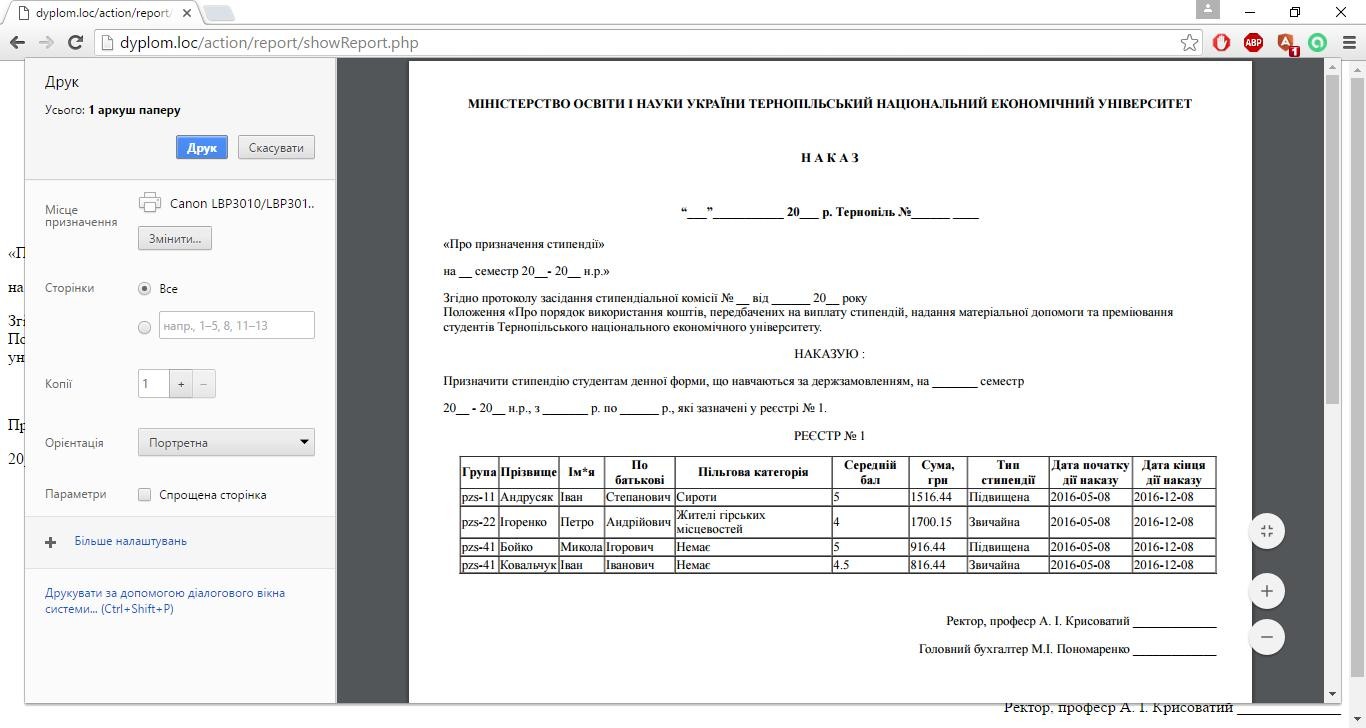


Рисунок 4.8 - Результат дії функції «Звіти -> Cформувати наказ»

Перегляд звітності по призначеній стипендії можна здійснити за допомогою функції «Звіти -> Перегляд», результатом дії якої буде сторінка на якій потрібно обрати критерії формування звіту. Цю сторінку зображено на рисунку 4.9. Після вибору критеріїв формування списків і натискання на кнопку пошуку сформується таблиця із відповідним звітом (див. рис. 4.10).

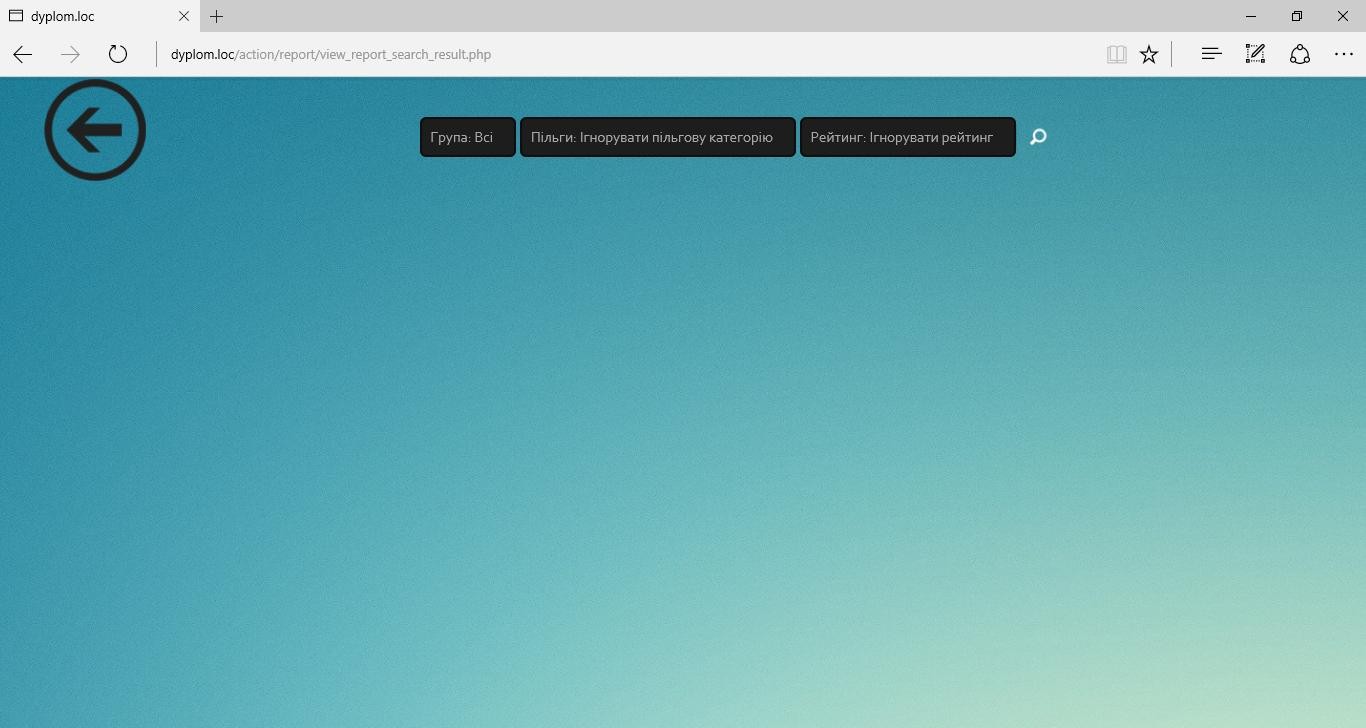


Рисунок 4.9 - Результат дії функції «Звіти -> Перегляд»

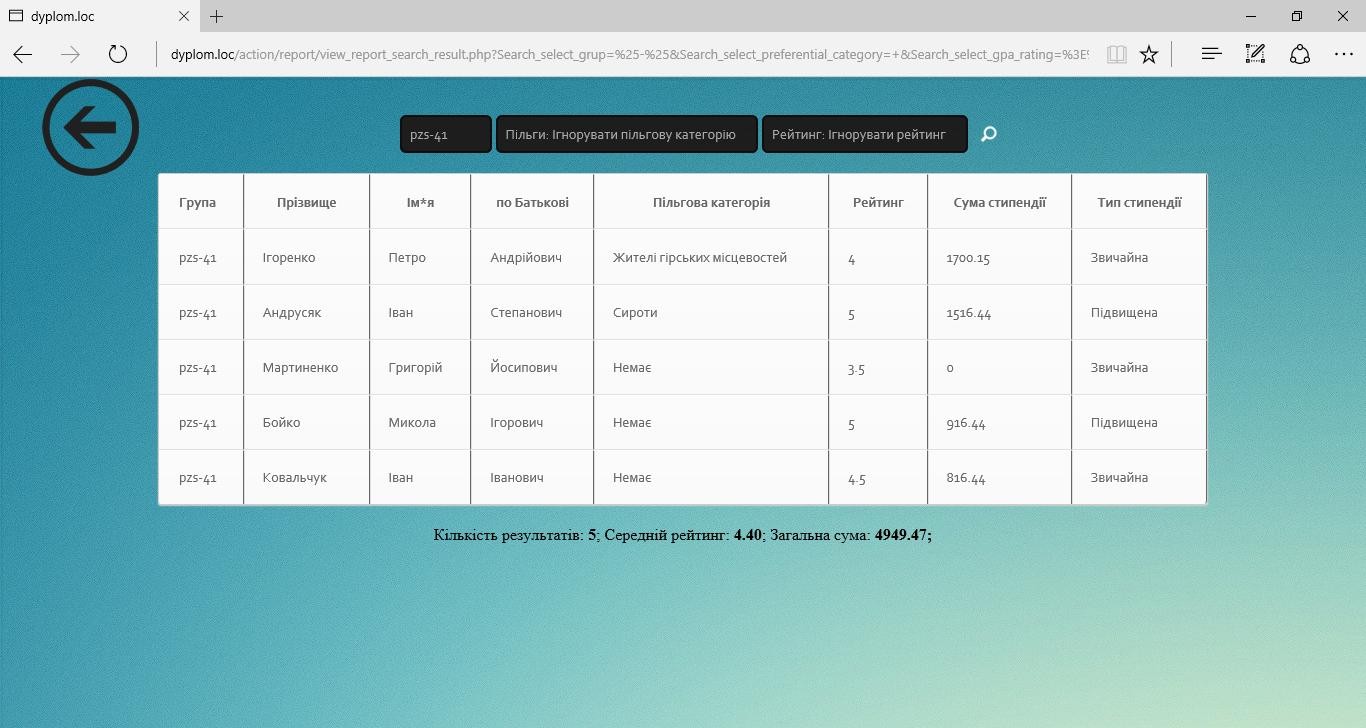


Рисунок 4.10 – Сторінка зі сформованим результатом стипендійних даних по обраним критеріям

Для перегляду списку студентів використовується пункт «Студенти ->

Відкрити список». Результатом виконання цієї функції буде сторінка із

сформованим списком всіх студентів, меню вибору сортування та пошуку (див.

рис. 4.11).

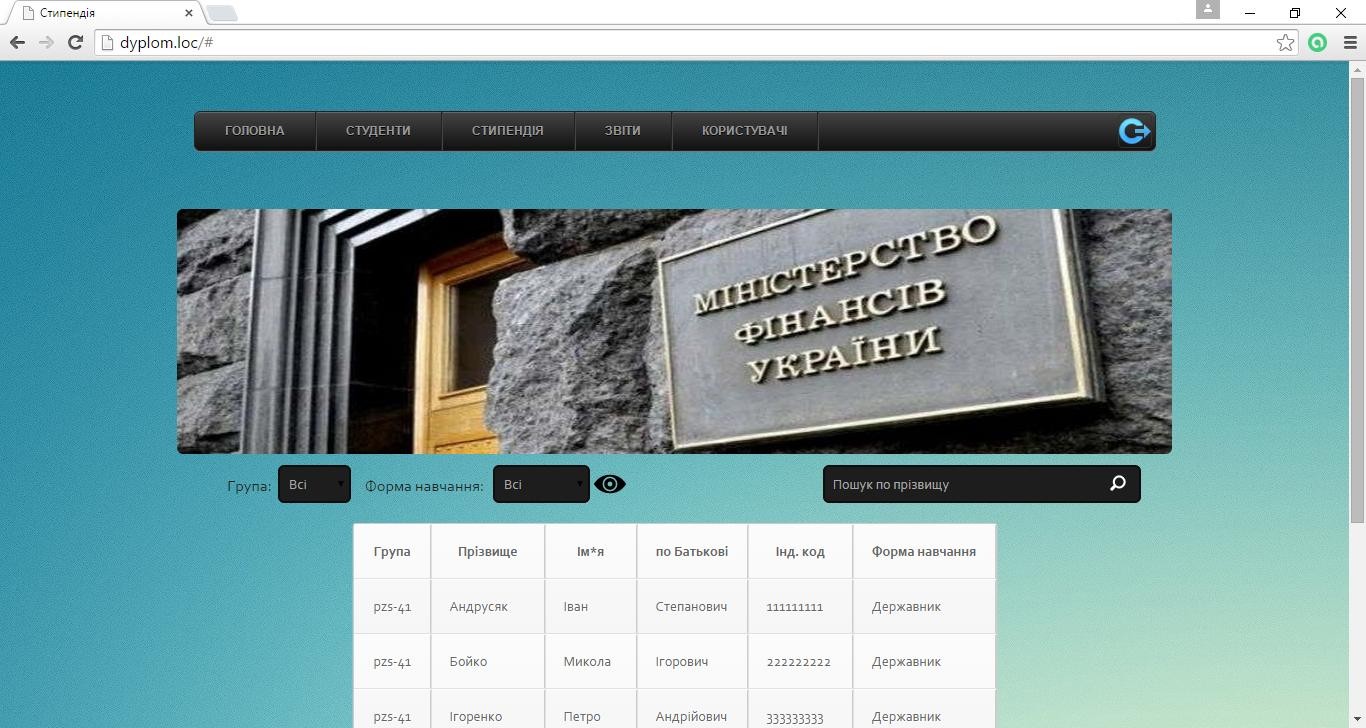


Рисунок 4.11 – Результат виконання функції «Студенти -> Відкрити список»

Для виходу із системи потрібно натиснути на кнопку logout, яка знаходиться у правому куті навігаційного меню (Рисунок 4.12). При наведенні курсора на цю кнопку її піктограма зміниться (Рисунок 4.13) і після натискання здійсниться вихід із системи.



Рисунок 4.12 – Кнопка «logout»



Рисунок 4.13 – Кнопка «logout» при наведенні курсора

Після цієї дії змінні сесії очищуються і сеанс роботи із системою вважається завершеним.

Висновки до розділу ІV

В ході виконання розділу було виконано тестування веб-орієнтованої системи нарахування стипендії. Реальна поведінка усіх функцій системи відповідає тій, що прописана у специфікації. Налаштовано та наповнено тестову бузу даних. Перевірено роботу програмного продукту у різних браузерах. Визначено час з’єднання та час відповіді сервера. Ідентифіковано степінь зручності користувацького інтерфейсу та ступень захисту даних програмного продукту при збереженні функціональності.

Також складено інструкцію користувача та список вимог, дотримання яких необхідно для перетворення звичайного ПК в сервер.

Отож, після проведення тестування веб-орієнтованої системи нарахування стипендії можна зробити висновок про те, що загальний результат

– позитивний. Оскільки всі тестові випадки виконано, вимоги продуктивності задоволено, всі баги виправлено і виправлення підтверджено, тестування безпеки пройдено.

ВИСНОВКИ

Результатом написання дипломної роботи є готовий програмний продукт, який являє собою веб-орієнтовану систему нарахування стипендії. Ключовою перевагою розробленої системи являється спрощення роботи працівника ВНЗ з процесом нарахування стипендії та супроводжуючих цей процес функцій, без прив’язки до робочого місця з встановленим ПЗ.

Розроблена веб-орієнтована система на відміну від її аналогів, майже повністю автоматизує процес нарахування стипендії шляхом автоматичного визначення рейтингу, можливості отримувати підвищену та пільгову стипендію і допуску до отримання стипендії на основі даних, отриманих із заліково-екзаменаційної сесії.

Окрім цього, за допомогою розробленої системи доступна можливість друку автоматично-згенерованого наказу на призначення стипендії на основі вище-згаданих критеріїв. Також реалізовано механізм сортування студентів по групах та формі навчання, пошук по прізвищу. Ще доступна функція формування звіту із стипендійними даними та реалізовано механізм сортування по групі, наявності пільгової категорії та рейтингу. Реалізована функція обрахунку середнього рейтингу та загальної суми стипендії, відповідно до раніше-згадуваних критеріїв.

Важливою особливістю розробленої веб-орієнтованої системи нарахування стипендії є те, що робота з нею можлива з будь-якого місця, де є доступ до мережі Інтернет.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Депонування коштів [Електронний ресурс]. – Режим доступу:<http://leksika.com.ua/17140326/legal/deponuvannya_groshey_koshtiv>
2. Стипендія [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/%D0%A1%D1%82%D0%B8%D0%BF%D0%B5

%D0%BD%D0%B4%D1%96%D1%8F

1. Індексація [Електронний ресурс]. – Режим доступу: https://news.dtkt.ua/ua/tags/indeksacija

|  |  |  |
| --- | --- | --- |
| 4. ПДФО [Електронний ресурс]. – | Режим | доступу: |
| <http://www.vobu.com.ua/ukr/news/view/50743>  5. Загородній А.Г., Партин Г.О. Бухгалтерський | облік: основи | теорії та |

практики: Навч. посіб. — 2-ге вид., перероб. І доп. — K.: Т-во “Знання”,

КОО, 2014. —327 с.

1. Горев А., Макашарипов С. «Эффективная работа с СУБД»: СПб, «Питер», 2012. – 512c.
2. Поль Дюбуа: MySQL. Издательство: Вильямс, 2006. —413 с.
3. Астахова И.Ф., Толстобров А.П., Мельников В.М. SQL в примерах и задачах. Воронеж, 2001. 160с.
4. Codd E.F. Further Normalization of the Data base Relational Model //Data Base Systems.- N.J.: Prentice-Hall, 1972. - P.33-64.
5. Чен П. Модель "сущность-связь" - шаг к единому представлению о данных

//СУБД. - 2005. - №3. - С.137-158.

1. Мартин Д. Планирование развития автоматизированных систем. - М.: Финансы и статистика, 2009. - 196 с.
2. Кириллов В.В. Структуризованный язык запросов (SQL). - СПб.: ИТМО, 2011. - 80 с.
3. Грабер М. Справочное руководство по SQL. - М.: Лори, 1997. - 291 с.
4. Кузнецов С. Д. Основы баз данных. – 2-е изд. –М.: Интернет-университет информационных технологий; БИНОМ. Лаборатория знаний, 2007.
5. Хомоненко А.Д., Цыганков В.М., Мальцев М.Г. Базы данных: Учебник для высших учебных заведений / Под ред. проф. А.Д. Хомоненко. - СПб.: КОРОНА принт, 2004. - 416с.
6. Тиори Т., Фрай Дж. Проектирование структур баз данных. Т.1,2, - М.: Мир,1985. -287с., -320с.
7. Майерс Г. Искусство тестирования программ / Пер. с англ. под ред. Б. А. Позина. – М.: Финансы и статистика, 2009. – 176 с.
8. Mуеrs G. J. A Controlled Experiment in Programm Testing and Code Walk- throughs/Inspections. – Commun. ACM, 1978, 21(9), p. 760–768.
9. Apache HTTP Server. [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Apache\_HTTP\_Server
10. Как разместить сайт в Интернет. Хостинг и домен для сайта. [Електронний ресурс]. – Режим доступу: <http://bitte.net.ua/blog/kak-zalit-sayt-na-hosting/>
11. Загрузка сайта на хостинг. [Електронний ресурс]. – Режим доступу: https://myrusakov.ru/site-to-hosting.html

ДОДАТОК А

ВИХІДНИЙ КОД РОЗРОБКИ ВЕБ-ОРІЄНТОВАНОЇ СИСТЕМИ НАРАХУВАННЯ СТИПЕНДІЇ

Index.php

<?php session\_start();

echo'<link rel="stylesheet" href="../css/style.css"/>

<script type="text/javascript" src="js/jquery.1.7.1.min.js"></script>

<script type="text/javascript" src="js/script1.js"></script>'; class SITE{

function go(){

$connect = mysql\_connect('localhost','root','') or die(mysql\_error()); mysql\_select\_db('dyplom');

if (isset($\_POST['enter'])){

$e\_login = $\_POST['e\_login'];

$e\_password = md5($\_POST['e\_password']);

$query = mysql\_query("SELECT \* FROM users WHERE login='$e\_login'");

$user\_data = mysql\_fetch\_array($query);

if ($user\_data['password']== $e\_password) {

$\_SESSION['usr'] = $e\_login;

$\_SESSION['position'] = $user\_data['position'];

}

else {

echo "<h2 align='center' style='color:#383838 ;background:#C3BEBA; font-family: Lobster, cursive;'>Помилка вводу логіна/пароля!</h2>";

}

}

if(isset($\_POST['logout'])){ unset($\_SESSION['usr']); session\_destroy();

}

$b = new LOADER();

if (isset($\_SESSION['usr']))

{

echo "<title>Стипендія</title>";

$b -> showBlocks();

$b -> loadScript();

}

else

{

$b -> authBuild();

}

}

}

class LOADER{ function loadScript(){ echo"<script>

function showContent(link) {

var cont = document.getElementById('contentBody'); var loading = document.getElementById('loading'); cont.innerHTML = loading.innerHTML;

var http = createRequestObject(); if( http ) {

http.open('get', link); http.onreadystatechange = function () { if(http.readyState == 4) { cont.innerHTML = http.responseText;

}

}

http.send(null);

} else { document.location = link;

}

}

function createRequestObject() {

try { return new XMLHttpRequest() } catch(e) {

try { return new ActiveXObject('Msxml2.XMLHTTP') } catch(e) {

try { return new ActiveXObject('Microsoft.XMLHTTP') } catch(e) { return null; }

}

}

}

</script>";

}

function showBlocks(){

echo'<table id="entered" style="top: 145px;">';

$this -> menuBuild();

$this -> contentBuild();

$this -> footerBuild(); echo'</table>';

echo'<div id="loading" style="display: none">'; echo'<img src="images/load.gif">'; echo'</div>';

}

function authBuild(){

echo'<div id="login\_container">'; echo'<div id="form\_container">';

echo'<p class="login-text">Авторизація на сайті</p>'; echo'<form id="enter" method="POST" action="index.php">';

echo'<input type="text" id="e\_log" name="e\_login" class="text\_input" placeholder=" Логін" required/><br>';

echo'<input type="password" id="e\_pass"class="text\_input" name="e\_password"placeholder="

Пароль"required /><br>';

echo'<input type="submit" id="e\_button" value="Вхід" name="enter" ;/>'; echo'</form>';

echo'</div>'; echo'</div> ';

}

function menuBuild(){ echo "<div id='header'>

<nav id='menu-wrap'>

<ul id='menu'>

<form align = 'right'; id='a\_log' method='post' action='index.php'>

<li><a href='#' onclick=showContent('forms/headerBuild.php');>Головна</a></li>

<li>

<a href=''>Студенти</a>

<ul>

<li>

<a href='#' onclick=showContent('action/students/view\_students\_form\_of\_education\_all.php');>Відкрити список</a>

</li>

<li>

<a href='#' onclick=showContent('action/students/add\_student.php')>Додати нового</a>

</li>

<li>

<a href='#' onclick=showContent('action/students/update\_students.php')>Редагувати</a>

</li>

<li>

<a href='#' onclick=showContent('action/students/delete\_student.php')>Видалити</a>

</li>

</ul>

</li>

<li>

<a href=''>Стипендія</a>

<ul>

<li>

<a href='#' onclick=showContent('action/stypendia/gpaRating\_and\_bursarySum.php');>Нарахування</a>

</li>";

if ($\_SESSION['position'] !== 'Бухгалтер') { echo "<li>

<a href=''>Редагувати</a>

<ul>

<li>

<a href='#' onclick=showContent('action/stypendia/update\_date.php')>Дату дії</a>

<li>

<a href='#' onclick=showContent('action/stypendia/update\_st\_type.php')>Суму підвищеної ст.</a>

</li>

<li>

<a href='#' onclick=showContent('action/stypendia/update\_baz\_st.php')>Базову стипендію</a>

</li>

</ul>

</li>

<li>";

}

echo" </li>

</ul>

</li>

<li>

<a href=''>Звіти</a>

<ul>

<li>

<a href='action/report/showReport.php'>Сформувати наказ</a>

</li>

<li>

<a href='action/report/view\_report\_search\_result.php'>Перегляд</a>

</li>

</ul> ";

if ($\_SESSION['position'] == 'Адмін'){ echo

"

<li>

<a href=''>Користувачі</a>

<ul>

<li>

<a href='#' onclick=showContent('action/users/view\_users.php');>Перегляд</a>

</li>

<li>

<a href='#' onclick=showContent('action/users/update\_users.php')> Редагувати </a>

</li>

<li>

<a href='#' onclick=showContent('action/users/add\_user.php')>Додати нового</a>

</li>

<li>

<a href='#' onclick=showContent('action/users/delete\_user.php')>Видалити</a>

</li>

</ul>

</li>

<input id='exit\_btn' type='submit' name='logout' value='' >";

}

else {

echo "<right><input id='exit\_btn' type='submit' name='logout' value=''style='margin-left: 445px;'/></right>";}

echo"

</form>

</ul>

</nav>

</td>

</div> ";

function contentBuild(){ echo'

<div>

<td align="center">

<div id="contentBody">

</div>

</td>

</div> ';

}

function footerBuild(){ echo'

<tr>

<td id=footer align="center">

<hr color="#373737">

<h5>© 2016 | <a id= "author" href=["http://vk.com](http://vk.com/just1_4fun)/[just1\_4fun"](http://vk.com/just1_4fun) target="\_blank" > Дубчак Руслан</a></h5>

</td>

</tr> ';

}

}

$openSite = new SITE();

$openSite -> go();

?>

Stypendia.php

<?php

class STYPENDIA{ function ConnectToDB(){

$hostname = 'localhost';

$username = 'root';

$password = "";

$dbName = "dyplom";

mysql\_connect($hostname, $username, $password) or die ("Не можливо створити з\*єднання!"); mysql\_select\_db($dbName) or die (mysql\_error());

function CalculateGPA\_and\_bursarySum($load\_data\_query,$max\_students\_id\_query,$update\_GPA\_query){

$max\_students\_id\_query =(" SELECT MAX(students\_id) as m\_s\_id FROM stypendia ");

$res = mysql\_query($load\_data\_query) or die(mysql\_error());

$max\_student\_id = mysql\_query($max\_students\_id\_query) or die(mysql\_error()); while ($m\_s\_id\_arr = mysql\_fetch\_array($max\_student\_id)) {

echo "<tr>\n";

$m\_s\_id = $m\_s\_id\_arr['m\_s\_id']; for ($i=1; $i<=$m\_s\_id;$i++){

$update\_GPA\_query =("UPDATE stypendia SET gpa\_rating = (

Select avg(mark.rating\_fifth\_s) FROM mark

INNER JOIN `session` ON `session`.mark\_id = mark.id WHERE `session`.student\_id = '$i'

)

WHERE stypendia.students\_id ='$i' ");

$res3 = mysql\_query($update\_GPA\_query) or die(mysql\_error());

}

}

echo ('

<table class="simple-little-table" border="1" cellspacing="0">

<tr style=\"border: solid 1px #000\">

<td align="center"><b>Група</b></td>

<td align="center"><b>Прізвище</b></td>

<td align="center"><b>Ім\*я</b></td>

<td align="center"><b>По батькові</b></td>

<td align="center"><b>Пільгова категорія</b></td>

<td align="center"><b>Середній бал</b></td>

<td align="center"><b>Допуск</b></td>

<td align="center"><b>Сума, грн</b></td>

<a id ="1">

</tr> ');

while ($row = mysql\_fetch\_array($res)) { echo "<tr>\n";

echo "<td>".$row['grup']."</td>\n";

echo "<td>".$row['surname']."</td>\n";

echo "<td>".$row['student\_name']."</td>\n"; echo "<td>".$row['f\_name']."</td>\n";

echo "<td>".$row['preferential\_category']."</td>\n"; echo "<td>".$row['gpa\_rating']."</td>\n";

echo "<td>".$row['admission']."</td>\n"; echo "<td>".$row['bursary\_sum']."</td>\n";

}

echo ("</table>\n </a>");

$res = mysql\_query('CALL `calculate\_stypendia`()') or die(mysql\_error());

$max\_student\_id2 = mysql\_query($max\_students\_id\_query) or die(mysql\_error()); while ($m\_s\_id\_arr2 = mysql\_fetch\_array($max\_student\_id2)) {

$m\_s\_id2 = $m\_s\_id\_arr2['m\_s\_id']; for ($j=1; $j<=$m\_s\_id2;$j++){

$update\_bursary\_sum\_query =("UPDATE stypendia SET bursary\_sum = (

SELECT `sum` FROM (

SELECT suma + pref\_cat\_sum + add\_sum as `sum` FROM stypendia

INNER JOIN baz\_st on baz\_st.id = stypendia.baz\_st\_id

INNER JOIN preferential\_category on preferential\_category.id = stypendia.preferential\_category\_id INNER JOIN st\_type on st\_type.id = stypendia.st\_type\_id

WHERE stypendia.students\_id = '$j'

)

AS ResultSet

)

WHERE stypendia.students\_id ='$j' AND stypendia.admission = 'yes' ");

$resultat = mysql\_query($update\_bursary\_sum\_query) or die(mysql\_error());

}

}

mysql\_close();

}

function updateDate(){ if(@$\_POST['edit'])

{

$query = "UPDATE date SET date1='{$\_POST['edit\_date1']}', date2='{$\_POST['edit\_date2']}' WHERE id='{$\_POST['update']}'";

mysql\_query($query) or die (mysql\_error());

die (require\_once "../../forms/well\_done.php");

}

$query = "SELECT \* FROM date";

$res = mysql\_query($query) or die(mysql\_error());

$row = mysql\_num\_rows($res);

while ($row = mysql\_fetch\_array($res))

{

echo "<form action=\"action/stypendia/update\_date.php\" method=\"post\" name=\"edit\_form\">\n"; echo "<input type=\"hidden\" name=\"update\" value=\"".$row["id"]."\" />\n";

echo "<table class=\"simple-little-table\" border=\"1\" cellspacing=\"0\">\n"; echo "<tr>\n";

echo "<td colspan=\"2\" style=\"border-bottom:solid 1px #CCCCCC;\"><b><i><div id=\"num\">#".$row["id"]."</div></b></i></td>\n";

echo "</tr><tr>\n";

echo "<td>Дата початку дії стипендії:</td><td><input type=\"text\" value=\"".$row['date1']."\" name=\"edit\_date1\" /></td>\n";

echo "</tr><tr>\n";

echo "<td>Дата кінця дії стипендії:</td><td><input type=\"text\" value=\"".$row['date2']."\" name=\"edit\_date2\" /></td>\n";

echo "</tr><tr>\n";

echo "<td colspan=\"2\" align=\"center\"><input type=\"submit\" name=\"edit\" class=\"buttons\" value=\"Зберегти зміни\" /></td>\n";

echo "</tr></table></form>\n\n";

}

echo"</div>"; mysql\_close();

}

function update\_st\_type(){ if(@$\_POST['edit'])

{

$query = "UPDATE st\_type SET add\_sum='{$\_POST['edit\_add\_sum']}' WHERE id='{$\_POST['update']}'";

mysql\_query($query) or die (mysql\_error()); die (require\_once "../../forms/well\_done.php");

}

$query = "SELECT \* FROM st\_type";

$res = mysql\_query($query) or die(mysql\_error());

$row = mysql\_num\_rows($res);

while ($row = mysql\_fetch\_array($res))

{

echo "<form action=\"action/stypendia/update\_st\_type.php\" method=\"post\" name=\"edit\_form\">\n"; echo "<input type=\"hidden\" name=\"update\" value=\"".$row["id"]."\" />\n";

echo "<table class=\"simple-little-table\" border=\"1\" cellspacing=\"0\">\n"; echo "<tr>\n";

echo "<td colspan=\"2\" style=\"border-bottom:solid 1px #CCCCCC;\"><b><i><div id=\"num\">#".$row["id"]."</div></b></i></td>\n";

echo "</tr><tr>\n";

echo "<td>Тип стипендії:</td><td><input type=\"text\" disabled value=\"".$row['st\_type\_name']."\" name=\"edit\_st\_type\_name\" /></td>\n";

echo "</tr><tr>\n";

echo "<td>Сума надбавки:</td><td><input type=\"text\" value=\"".$row['add\_sum']."\" name=\"edit\_add\_sum\" /></td>\n";

echo "</tr><tr>\n";

echo "<td colspan=\"2\" align=\"center\"><input type=\"submit\" name=\"edit\" class=\"buttons\" value=\"Зберегти зміни\" /></td>\n";

echo "</tr></table></form>\n\n";

}

echo"</div>"; mysql\_close();

}

function update\_baz\_st(){ if(@$\_POST['edit'])

{

$query = "UPDATE baz\_st SET suma='{$\_POST['edit\_suma']}' WHERE id='{$\_POST['update']}'";

mysql\_query($query) or die (mysql\_error()); die (require\_once "../../forms/well\_done.php");

}

$query = "SELECT \* FROM baz\_st";

$res = mysql\_query($query) or die(mysql\_error());

$row = mysql\_num\_rows($res);

while ($row = mysql\_fetch\_array($res))

{

echo "<form action=\"action/stypendia/update\_baz\_st.php\" method=\"post\" name=\"edit\_form\">\n"; echo "<input type=\"hidden\" name=\"update\" value=\"".$row["id"]."\" />\n";

echo "<table class=\"simple-little-table\" border=\"1\" cellspacing=\"0\">\n"; echo "<tr>\n";

echo "<td colspan=\"2\" style=\"border-bottom:solid 1px #CCCCCC;\"><b><i><div id=\"num\">#".$row["id"]."</div></b></i></td>\n";

echo "</tr><tr>\n";

echo "<td>Освітній рівень:</td><td><input type=\"text\" disabled value=\"".$row['ed\_level']."\" name=\"edit\_ed\_level\" /></td>\n";

echo "</tr><tr>\n";

echo "<td>Сума стипендії:</td><td><input type=\"text\" value=\"".$row['suma']."\" name=\"edit\_suma\"

/></td>\n";

echo "</tr><tr>\n";

echo "<td colspan=\"2\" align=\"center\"><input type=\"submit\" name=\"edit\" class=\"buttons\" value=\"Зберегти зміни\" /></td>\n";

echo "</tr></table></form>\n\n";

}

echo"</div>"; mysql\_close();

}

}

?>

ДОДАТОК Б

РЕЗУЛЬТАТ ФІЗИЧНОГО ПРОЕКТУВАННЯ БД

-- phpMyAdmin SQL Dump

-- version 4.0.10.10

-- [http://www.phpmyadmin.net](http://www.phpmyadmin.net/)

--

-- Хост: 127.0.0.1:3306

-- Время создания: Май 29 2016 г., 23:54

-- Версия сервера: 5.5.45

-- Версия PHP: 5.3.29

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

SET time\_zone = "+00:00";

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8 \*/;

--

-- База данных: `dyplom`

--

DELIMITER $$

--

-- Процедуры

--

CREATE DEFINER=`root`@`%` PROCEDURE `calculate\_stypendia`() MODIFIES SQL DATA

BEGIN

='Звичайна')

='Підвищена')

='Звичайна')

UPDATE stypendia

SET stypendia.st\_type\_id = (SELECT id from st\_type where st\_type.st\_type\_name WHERE stypendia.gpa\_rating >=4 and stypendia.gpa\_rating<5;

UPDATE stypendia

SET stypendia.st\_type\_id = (SELECT id from st\_type where st\_type.st\_type\_name WHERE stypendia.gpa\_rating =5;

UPDATE stypendia

SET stypendia.st\_type\_id = (SELECT id from st\_type where st\_type.st\_type\_name

WHERE stypendia.gpa\_rating >=3 and stypendia.gpa\_rating<4 and

stypendia.preferential\_category\_id != (SELECT id from preferential\_category WHERE preferential\_category.name='Немає');

UPDATE stypendia

SET stypendia.admission = 'no' WHERE stypendia.gpa\_rating <4;

UPDATE stypendia

SET stypendia.admission = 'yes' WHERE stypendia.gpa\_rating >=4;

UPDATE stypendia

SET stypendia.admission = 'yes'

WHERE stypendia.preferential\_category\_id != (SELECT id from preferential\_category WHERE preferential\_category.name='Немає');

UPDATE stypendia

SET stypendia.admission = 'no' WHERE stypendia.gpa\_rating <3;

UPDATE stypendia

SET stypendia.bursary\_sum = 0 WHERE stypendia.admission ='no';

END$$ DELIMITER ;

-- --------------------------------------------------------

--

-- Структура таблицы `baz\_st`

--

CREATE TABLE IF NOT EXISTS `baz\_st` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`ed\_level` varchar(50) NOT NULL,

`suma` double NOT NULL, PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=5 ;

--

-- Дамп данных таблицы `baz\_st`

--

INSERT INTO `baz\_st` (`id`, `ed\_level`, `suma`) VALUES (1, 'Магістр', 916.56),

(2, 'Спеціаліст', 900.15),

(3, 'Бакалавр', 816.44),

(4, 'Молодший спеціаліст', 544.5);

-- --------------------------------------------------------

--

-- Структура таблицы `date`

--

CREATE TABLE IF NOT EXISTS `date` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`date1` date NOT NULL,

`date2` date NOT NULL, PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=2 ;

--

-- Дамп данных таблицы `date`

--

INSERT INTO `date` (`id`, `date1`, `date2`) VALUES (1, '2016-05-08', '2016-12-08');

-- --------------------------------------------------------

--

-- Структура таблицы `form\_of\_education`

--

CREATE TABLE IF NOT EXISTS `form\_of\_education` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`name` varchar(50) NOT NULL, PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=3 ;

--

-- Дамп данных таблицы `form\_of\_education`

--

INSERT INTO `form\_of\_education` (`id`, `name`) VALUES (1, 'Державник'),

(2, 'Платник');

-- --------------------------------------------------------

--

-- Структура таблицы `grup`

--

CREATE TABLE IF NOT EXISTS `grup` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`name` varchar(50) NOT NULL, PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=11 ;

--

-- Дамп данных таблицы `grup`

--

INSERT INTO `grup` (`id`, `name`) VALUES (1, 'pzs-11'),

(2, 'pzs-12'),

(3, 'pzs-21'),

(4, 'pzs-22'),

(5, 'pzs-31'),

(6, 'pzs-32'),

(7, 'pzs-41'),

(8, 'pzs-42'),

(9, 'pzs-51'),

(10, 'pzs-52');

-- --------------------------------------------------------

--

-- Структура таблицы `mark`

--

CREATE TABLE IF NOT EXISTS `mark` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`rating` varchar(50) NOT NULL,

`rating\_bol\_min` int(11) NOT NULL,

`rating\_bol\_max` int(11) NOT NULL,

`rating\_fifth\_s` float NOT NULL, PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=8 ;

--

-- Дамп данных таблицы `mark`

--

INSERT INTO `mark` (`id`, `rating`, `rating\_bol\_min`, `rating\_bol\_max`, `rating\_fifth\_s`) VALUES

(1, 'A', 90, 100, 5),

(2, 'B', 85, 89, 4),

(3, 'C', 75, 84, 4),

(4, 'D', 70, 74, 3),

(5, 'E', 60, 69, 3),

(6, 'FX', 35, 59, 2),

(7, 'F', 0, 34, 2);

-- --------------------------------------------------------

--

-- Структура таблицы `personal\_data`

--

CREATE TABLE IF NOT EXISTS `personal\_data` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`surname` varchar(50) CHARACTER SET cp1251 NOT NULL,

`name` varchar(50) CHARACTER SET cp1251 NOT NULL,

`f\_name` varchar(50) CHARACTER SET cp1251 NOT NULL,

`id\_number` int(11) NOT NULL,

`form\_of\_education\_id` int(11) NOT NULL, PRIMARY KEY (`id`),

KEY `form\_of\_education\_id` (`form\_of\_education\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=cp1251 COLLATE=cp1251\_ukrainian\_ci AUTO\_INCREMENT=42 ;

--

-- Дамп данных таблицы `personal\_data`

--

INSERT INTO `personal\_data` (`id`, `surname`, `name`, `f\_name`, `id\_number`,

`form\_of\_education\_id`) VALUES

(1, 'Назаренко', 'Назар', 'Назарович', 666666666, 2),

(2, 'Мартиненко', 'Григорій', 'Йосипович', 555555555, 1),

(3, 'Андрусяк', 'Іван', 'Степанович', 111111111, 1),

(4, 'Ігоренко', 'Петро', 'Андрійович', 333333333, 1),

(5, 'Потоцький', 'Юрій', 'Сергійович', 777777777, 2),

(40, 'Ковальчук', 'Іван', 'Іванович', 444444444, 1),

(41, 'Бойко', 'Микола', 'Ігорович', 222222222, 1);

-- --------------------------------------------------------

--

-- Структура таблицы `preferential\_category`

--

CREATE TABLE IF NOT EXISTS `preferential\_category` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`name` varchar(50) NOT NULL,

`pref\_cat\_sum` float NOT NULL, PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=5 ;

--

-- Дамп данных таблицы `preferential\_category`

--

INSERT INTO `preferential\_category` (`id`, `name`, `pref\_cat\_sum`) VALUES (1, 'Немає', 0),

(2, 'Сироти', 600),

(3, 'Батьки молодих сімей', 700),

(4, 'Жителі гірських місцевостей', 800);

-- --------------------------------------------------------

--

-- Структура таблицы `session`

--

CREATE TABLE IF NOT EXISTS `session` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`student\_id` int(11) NOT NULL,

`subject\_id` int(11) NOT NULL,

`teacher\_id` int(11) NOT NULL,

`mark\_id` int(11) NOT NULL, PRIMARY KEY (`id`),

KEY `student\_id` (`student\_id`), KEY `subject\_id` (`subject\_id`), KEY `teacher\_id` (`teacher\_id`), KEY `mark\_id` (`mark\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=31 ;

--

-- Дамп данных таблицы `session`

--

INSERT INTO `session` (`id`, `student\_id`, `subject\_id`, `teacher\_id`, `mark\_id`) VALUES (1, 1, 1, 4, 1),

(2, 1, 2, 5, 1),

(3, 1, 3, 2, 1),

(4, 1, 4, 3, 1),

(5, 1, 5, 1, 1),

(6, 1, 6, 5, 1),

(7, 2, 6, 5, 3),

(8, 2, 6, 5, 3),

(9, 2, 6, 5, 3),

(10, 2, 6, 5, 3),

(11, 2, 6, 5, 3),

(12, 2, 6, 5, 3),

(13, 3, 6, 5, 4),

(14, 3, 6, 5, 4),

(15, 3, 6, 5, 4),

(16, 3, 6, 5, 3),

(17, 3, 6, 5, 3),

(18, 3, 6, 5, 3),

(19, 37, 6, 5, 3),

(20, 37, 6, 5, 3),

(21, 37, 6, 5, 3),

(22, 37, 6, 5, 1),

(23, 37, 6, 5, 1),

(24, 37, 6, 5, 1),

(25, 38, 6, 5, 1),

(26, 38, 6, 5, 1),

(27, 38, 6, 5, 1),

(28, 38, 6, 5, 1),

(29, 38, 6, 5, 1),

(30, 38, 6, 5, 1);

-- --------------------------------------------------------

--

-- Структура таблицы `students`

--

CREATE TABLE IF NOT EXISTS `students` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`grup\_id` int(11) NOT NULL,

`personal\_data\_id` int(11) NOT NULL, PRIMARY KEY (`id`),

KEY `personal\_data\_id` (`personal\_data\_id`), KEY `grup\_id` (`grup\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=39 ;

--

-- Дамп данных таблицы `students`

--

INSERT INTO `students` (`id`, `grup\_id`, `personal\_data\_id`) VALUES (1, 7, 3),

(2, 7, 4),

(3, 7, 2),

(4, 7, 1),

(8, 7, 5),

(37, 7, 40),

(38, 7, 41);

-- --------------------------------------------------------

--

-- Структура таблицы `stypendia`

--

CREATE TABLE IF NOT EXISTS `stypendia` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`students\_id` int(11) DEFAULT NULL,

`date\_id` int(11) DEFAULT NULL,

`baz\_st\_id` int(11) DEFAULT NULL,

`preferential\_category\_id` int(11) DEFAULT NULL,

`st\_type\_id` int(11) DEFAULT NULL,

`gpa\_rating` float DEFAULT NULL,

`admission` varchar(4) DEFAULT NULL,

`bursary\_sum` double DEFAULT NULL, PRIMARY KEY (`id`),

KEY `date\_id` (`date\_id`), KEY `baz\_st\_id` (`baz\_st\_id`),

KEY `st\_type\_id` (`st\_type\_id`),

KEY `preferential\_category\_id` (`preferential\_category\_id`), KEY `students\_id` (`students\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=19 ;

--

-- Дамп данных таблицы `stypendia`

--

INSERT INTO `stypendia` (`id`, `students\_id`, `date\_id`, `baz\_st\_id`, `preferential\_category\_id`,

`st\_type\_id`, `gpa\_rating`, `admission`, `bursary\_sum`) VALUES (1, 1, 1, 3, 2, 2, 5, 'yes', 1516.44),

(2, 2, 1, 2, 4, 1, 4, 'yes', 1700.15),

(5, 3, 1, 4, 1, 1, 3.5, 'no', 0),

(17, 37, 1, 3, 1, 1, 4.5, 'yes', 816.44),

(18, 38, 1, 3, 1, 2, 5, 'yes', 916.44);

-- --------------------------------------------------------

--

-- Структура таблицы `st\_type`

--

CREATE TABLE IF NOT EXISTS `st\_type` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`add\_sum` float NOT NULL,

`st\_type\_name` varchar(50) NOT NULL, PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=3 ;

--

-- Дамп данных таблицы `st\_type`

--

INSERT INTO `st\_type` (`id`, `add\_sum`, `st\_type\_name`) VALUES (1, 0, 'Звичайна'),

(2, 100, 'Підвищена');

-- --------------------------------------------------------

--

-- Структура таблицы `subject`

--

CREATE TABLE IF NOT EXISTS `subject` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`name` varchar(50) NOT NULL, PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=9 ;

--

-- Дамп данных таблицы `subject`

--

INSERT INTO `subject` (`id`, `name`) VALUES (1, 'Геометрія'),

(2, 'Алгебра'),

(3, 'Тестування ПЗ'),

(4, 'Мікропроцесорні системи'), (5, 'Основи охорони праці'),

(6, 'Групова динаміка і комунікації'), (7, 'Проектування БД'),

(8, 'Безпека програм та даних');

-- --------------------------------------------------------

--

-- Структура таблицы `teacher`

--

CREATE TABLE IF NOT EXISTS `teacher` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`name` varchar(50) NOT NULL, PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=6 ;

--

-- Дамп данных таблицы `teacher`

--

INSERT INTO `teacher` (`id`, `name`) VALUES (1, 'Костенко І. І.'),

(2, 'Тарасенко М. Ю.'),

(3, 'Миколаєв В. Г.'),

(4, 'Панда І. М.'),

(5, 'Поторчук Л. Н.');

-- --------------------------------------------------------

--

-- Структура таблицы `users`

--

CREATE TABLE IF NOT EXISTS `users` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`login` varchar(50) NOT NULL,

`password` varchar(50) NOT NULL,

`position` varchar(50) DEFAULT NULL, PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=9 ;

--

-- Дамп данных таблицы `users`

--

INSERT INTO `users` (`id`, `login`, `password`, `position`) VALUES (1, 'admin', '21232f297a57a5a743894a0e4a801fc3', 'Адмін'),

(2, 'user1', '24c9e15e52afc47c225b757e7bee1f9d', 'Головний бухгалтер'), (7, 'user2', '7e58d63b60197ceb55a1c487989a3720', 'Бухгалтер'),

(8, 'л', '1a41a6326d10d33048737d57b9fa866c', 'Адмін');

--

-- Ограничения внешнего ключа сохраненных таблиц

--

--

-- Ограничения внешнего ключа таблицы `personal\_data`

--

ALTER TABLE `personal\_data`

ADD CONSTRAINT `form\_of\_education\_id` FOREIGN KEY (`form\_of\_education\_id`) REFERENCES `form\_of\_education` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION;

--

-- Ограничения внешнего ключа таблицы `session`

--

ALTER TABLE `session`

ADD CONSTRAINT `mark\_id` FOREIGN KEY (`mark\_id`) REFERENCES `mark` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION,

ADD CONSTRAINT `student\_id` FOREIGN KEY (`student\_id`) REFERENCES `students` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION,

ADD CONSTRAINT `subject\_id` FOREIGN KEY (`subject\_id`) REFERENCES `subject` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION,

ADD CONSTRAINT `teacher\_id` FOREIGN KEY (`teacher\_id`) REFERENCES `teacher` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION;

--

-- Ограничения внешнего ключа таблицы `students`

--

ALTER TABLE `students`

ADD CONSTRAINT `grup\_id` FOREIGN KEY (`grup\_id`) REFERENCES `grup` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION,

ADD CONSTRAINT `personal\_data\_id` FOREIGN KEY (`personal\_data\_id`) REFERENCES

`personal\_data` (`id`) ON DELETE CASCADE ON UPDATE CASCADE;

--

-- Ограничения внешнего ключа таблицы `stypendia`

--

ALTER TABLE `stypendia`

ADD CONSTRAINT `baz\_st\_id` FOREIGN KEY (`baz\_st\_id`) REFERENCES `baz\_st` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION,

ADD CONSTRAINT `date\_id` FOREIGN KEY (`date\_id`) REFERENCES `date` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION,

ADD CONSTRAINT `preferential\_category\_id` FOREIGN KEY (`preferential\_category\_id`) REFERENCES `preferential\_category` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION,

ADD CONSTRAINT `students\_id` FOREIGN KEY (`students\_id`) REFERENCES `students` (`id`) ON DELETE CASCADE ON UPDATE NO ACTION,

ADD CONSTRAINT `st\_type\_id` FOREIGN KEY (`st\_type\_id`) REFERENCES `st\_type` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

ДОДАТОК В РЕЗУЛЬТАТИ ТЕСТУВАННЯ

Test suite

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd).[w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)>

<html xmln[s="http://www](http://www.w3.org/1999/xhtml).w[3.](http://www.w3.org/1999/xhtml)o[rg/1999/xhtml](http://www.w3.org/1999/xhtml)" xml:lang="en" lang="en">

<head>

<meta content="text/html; charset=UTF-8" http-equiv="content-type" />

<title>Test Suite</title>

</head>

<body>

<table id="suiteTable" cellpadding="1" cellspacing="1" border="1" class="selenium"><tbody>

<tr><td><b>Test Suite</b></td></tr>

<tr><td><a href="autorization\_test.html">Autorization\_test</a></td></tr>

<tr><td><a href="all\_tests.html">Add\_new\_student\_test</a></td></tr>

<tr><td><a href="all\_tests.html">Search\_new\_student\_test</a></td></tr>

<tr><td><a href="1.html">Update\_student\_test</a></td></tr>

<tr><td><a href="2.html">Delete\_student\_test</a></td></tr>

</tbody></table>

</body>

</html>

Add\_new\_student\_test\_case

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd).[w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)>

<html xmln[s="http://www](http://www.w3.org/1999/xhtml).w[3.](http://www.w3.org/1999/xhtml)o[rg/1999/xhtml](http://www.w3.org/1999/xhtml)" xml:lang="en" lang="en">

<head profile="<http://selenium-ide.openqa.org/profiles/test-case>">

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

<link rel="selenium.base" h[ref="h](http://dyplom.loc/)ttp:[//dyplom](http://dyplom.loc/).[loc/"](http://dyplom.loc/) />

<title>Add\_new\_student\_test</title>

</head>

<body>

<table cellpadding="1" cellspacing="1" border="1">

<thead>

<tr><td rowspan="1" colspan="3">Add\_new\_student\_test</td></tr>

</thead><tbody>

<tr>

</tr>

<tr>

</tr>

<tr>

<td>open</td>

<td>/</td>

<td></td>

<td>click</td>

<td>link=Додати нового</td>

<td></td>

<td>type</td>

<td>name=surname</td>

<td>Дубчак</td>

</tr>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

<td>type</td>

<td>name=name</td>

<td>Руслан</td>

<td>type</td>

<td>name=f\_name</td>

<td>Іванович</td>

<td>type</td>

<td>name=id\_number</td>

<td>123456789</td>

<td>select</td>

<td>name=grup\_id</td>

<td>label=пзс-41</td>

<td>select</td>

<td>name=baz\_st\_id</td>

<td>label=Бакалавр</td>

<td>clickAndWait</td>

<td>name=reg\_btn</td>

<td></td>

<td>clickAndWait</td>

<td>link=OK</td>

<td></td>

</tbody></table>

</body>

</html>

Authorization\_test\_case

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd).[w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)>

<html xmln[s="http://www](http://www.w3.org/1999/xhtml).w[3.](http://www.w3.org/1999/xhtml)o[rg/1999/xhtml](http://www.w3.org/1999/xhtml)" xml:lang="en" lang="en">

<head profile="<http://selenium-ide.openqa.org/profiles/test-case>">

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

<link rel="selenium.base" h[ref="h](http://dyplom.loc/)ttp:[//dyplom](http://dyplom.loc/).[loc/"](http://dyplom.loc/) />

<title>Autorization\_test</title>

</head>

<body>

<table cellpadding="1" cellspacing="1" border="1">

<thead>

<tr><td rowspan="1" colspan="3">Autorization\_test</td></tr>

</thead><tbody>

</tr>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

<td>open</td>

<td>/</td>

<td></td>

<td>type</td>

<td>id=e\_log</td>

<td>admin</td>

<td>type</td>

<td>id=e\_pass</td>

<td>admin</td>

<td>clickAndWait</td>

<td>id=e\_button</td>

<td></td>

</tbody></table>

</body>

</html>

Delete\_student\_test\_case

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd).[w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)>

<html xmln[s="http://www](http://www.w3.org/1999/xhtml).w[3.](http://www.w3.org/1999/xhtml)o[rg/1999/xhtml](http://www.w3.org/1999/xhtml)" xml:lang="en" lang="en">

<head profile="<http://selenium-ide.openqa.org/profiles/test-case>">

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

<link rel="selenium.base" h[ref="h](http://dyplom.loc/)ttp:[//dyplom](http://dyplom.loc/).[loc/"](http://dyplom.loc/) />

<title>Delete\_student\_test</title>

</head>

<body>

<table cellpadding="1" cellspacing="1" border="1">

<thead>

<tr><td rowspan="1" colspan="3">Delete\_student\_test</td></tr>

</thead><tbody>

<tr>

<td>open</td>

<td>/action/students/view\_students\_search\_result.php?Search\_select\_form\_of\_education=%25%D0

%B0%25&amp;searchSurname=%D0%B4%D1%83%D0%B1%D1%87&amp;search\_b=&amp;Search\_sele ct\_grup=%25-%25</td>

<td></td>

</tr>

<tr>

</tr>

<tr>

</tr>

<td>clickAndWait</td>

<td>id=back\_b</td>

<td></td>

<td>click</td>

<td>link=Видалити</td>

<td></td>

</tr>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

<td>clickAndWait</td>

<td>xpath=(//a[contains(text(),'Видалити')])[10]</td>

<td></td>

<td>clickAndWait</td>

<td>css=a &gt; img</td>

<td></td>

<td>click</td>

<td>link=Відкрити список</td>

<td></td>

<td>type</td>

<td>id=searchSurname</td>

<td>Дубч</td>

<td>clickAndWait</td>

<td>id=search\_b</td>

<td></td>

</tbody></table>

</body>

</html>

Search\_new\_student\_test\_case

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd).[w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)>

<html xmln[s="http://www](http://www.w3.org/1999/xhtml).w[3.](http://www.w3.org/1999/xhtml)o[rg/1999/xhtml](http://www.w3.org/1999/xhtml)" xml:lang="en" lang="en">

<head profile="<http://selenium-ide.openqa.org/profiles/test-case>">

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

<link rel="selenium.base" h[ref="h](http://dyplom.loc/)ttp:[//dyplom](http://dyplom.loc/).[loc/"](http://dyplom.loc/) />

<title>Search\_new\_student\_test</title>

</head>

<body>

<table cellpadding="1" cellspacing="1" border="1">

<thead>

<tr><td rowspan="1" colspan="3">Search\_new\_student\_test</td></tr>

</thead><tbody>

<tr>

</tr>

<tr>

</tr>

<tr>

<td>open</td>

<td>/index.php</td>

<td></td>

<td>click</td>

<td>link=Відкрити список</td>

<td></td>

</tr>

<tr>

</tr>

<td>type</td>

<td>id=searchSurname</td>

<td>Дубч</td>

<td>clickAndWait</td>

<td>id=search\_b</td>

<td></td>

</tbody></table>

</body>

</html>

Update\_student\_test\_case

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" ["http://www](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd).[w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)>

<html xmln[s="http://www](http://www.w3.org/1999/xhtml).w[3.](http://www.w3.org/1999/xhtml)o[rg/1999/xhtml](http://www.w3.org/1999/xhtml)" xml:lang="en" lang="en">

<head profile="<http://selenium-ide.openqa.org/profiles/test-case>">

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

<link rel="selenium.base" h[ref="h](http://dyplom.loc/)ttp:[//dyplom](http://dyplom.loc/).[loc/"](http://dyplom.loc/) />

<title>Update\_student\_test</title>

</head>

<body>

<table cellpadding="1" cellspacing="1" border="1">

<thead>

<tr><td rowspan="1" colspan="3">Update\_student\_test</td></tr>

</thead><tbody>

<tr>

<td>open</td>

<td>/action/students/view\_students\_search\_result.php?Search\_select\_form\_of\_education=%25%D0

%B0%25&amp;searchSurname=%D0%94%D1%83%D0%B1%D1%87&amp;search\_b=&amp;Search\_sele ct\_grup=%25-%25</td>

<td></td>

</tr>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

<td>clickAndWait</td>

<td>id=back\_b</td>

<td></td>

<td>click</td>

<td>link=Редагувати</td>

<td></td>

<td>select</td>

<td>xpath=(//select[@name='edit\_form\_of\_education\_id'])[8]</td>

<td>label=Платник</td>

<td>clickAndWait</td>

<td>xpath=(//input[@name='edit'])[8]</td>

<td></td>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

<td>clickAndWait</td>

<td>css=a &gt; img</td>

<td></td>

<td>click</td>

<td>link=Відкрити список</td>

<td></td>

<td>type</td>

<td>id=searchSurname</td>

<td>дубч</td>

<td>clickAndWait</td>

<td>id=search\_b</td>

<td></td>

</tbody></table>

</body>

</html>

ДОДАТОК Г

Функціональні тестові випадки

Таблиця 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Модуль | Назва | Кроки | Очікуваний результат | Статус |
| 1 | Auth | Авторизація | 1. Перехід на сторінку авторизації 2.Заповнення форми 3.Натискання кнопки  “вхід” | Відкриття основгого функціоналу системи | Виконано |
| 2 | Student | Перегляд списку студентів | 1. Вибір пункту меню: студенти ->  відкрити список | сформований список студентів та їх даних | Виконано |
| 3 | Student | Пошук студента | 1. Вибір пункту меню: студенти -> відкрити список 2. Введення пошукового запиту 3. Натискання кнопки   “пошук” | Перенаправлення на сторінку із результатами пошуку | Виконано |
| 4 | Student | Додавання студента | 1. Вибір пункту меню: студенти -> додати нового 2. Заповнення форми 3. Натискання кнопки “зберегти” | Успішно додані дані студента у БД | Виконано |
| 5 | Student | Редагування даних студента | 1. Вибір пункту меню: студенти -> редагувати 2. Внесення змін 3. Натискання кнопки “зберегти” | Успішно відредаговані дані студента у БД | Виконано |

Продовження таблиці 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 6 | Student | Видалення студента | 1. Вибір пункту меню: студенти -> видалити 2. Вибір студента 3. Натискання кнопки “видалити” | Успішно видалені дані студента із БД | Виконано |
| 7 | Styp | Нарахування | 1. Вибір пункту меню: стипендія -> нарахування | Успішно обрахований середній бал та сума стипендії, присвоєння дозволу для отримання стипендії і  відображення цих  відомостей | Виконано |
| 8 | Styp | Редагування базових відомостей про стипендію | 1. Вибір пункту меню: стипендія -> редагувати 2. Вибір критерію редагування 3. Внесення змін 4. Натискання кнопки “зберегти” | Успішно відредаговані базові відомості про стипендію у БД | Виконано |
| 9 | Report | Сформувати наказ | 1. Вибір пункту меню: звіт -> сформувати наказ | Сформований для друку наказ для отримання стипендії | Виконано |
| 10 | Report | Перегляд звіту по отримуваній стипендії | 1. Вибір пункту меню: звіти -> перегляд 2. Вибір критеріїв формування звіту 3. Натискання кнопки   “пошук” | Перенаправлення на сторінку із сформованим списком студентів та їх стипендійних даних, відповідно до обраних критеріїв | Виконано |