Зміст

[Анотація 4](#_Toc452648504)

[Abstract 5](#_Toc452648505)

[Вступ 6](#_Toc452648506)

[Розділ 1. Огляд програмного забезпечення для автоматизованого управління університетом 8](#_Toc452648507)

[1.1. Програмні засоби 8](#_Toc452648508)

[1.1.1. Автоматизована система управління вищим навчальним закладом III - IV рівня акредитації Uniteh+ 8](#_Toc452648509)

[1.1.2. АСУ «ВНЗ» 9](#_Toc452648510)

[1.1.3. Пакет програм від Політек-Софт 11](#_Toc452648511)

[Розділ 2. Постановка задачі розробки програмного продукту для формування та друку документації та звітності у ІС «Деканат» 16](#_Toc452648512)

[2.1. Загальна постановка задачі 16](#_Toc452648513)

[2.2. Вибір інструментальних засобів 16](#_Toc452648514)

[2.3. Специфікація вимог до програмного продукту 17](#_Toc452648515)

[2.3.1. Вступ 17](#_Toc452648516)

[2.3.1.1. Призначення, мета 17](#_Toc452648517)

[2.3.1.2. Продукти аналоги 17](#_Toc452648518)

[2.3.2. Загальний опис 18](#_Toc452648519)

[2.3.2.1. Характеристики продукту 18](#_Toc452648520)

[2.3.2.2. Класи користувачів та їх характеристики 18](#_Toc452648521)

[2.3.2.3. Середовище функціонування 18](#_Toc452648522)

[2.3.2.4. Характеристики системи 19](#_Toc452648523)

[2.3.2.4.1. Пошук студентів про групах, інститутах 19](#_Toc452648524)

[2.3.2.4.2. Формування документації та звітності 19](#_Toc452648525)

[2.3.2.4.3. Збереження сформованого документу 20](#_Toc452648526)

[2.3.2.4.4. Друк документу 20](#_Toc452648527)

[2.3.2.4.5. Друк вже існуючого документу 21](#_Toc452648528)

[2.3.2.4.6. Друк статистичних відомостей 21](#_Toc452648529)

[2.3.3. Вимоги зовнішніх інтерфейсів 22](#_Toc452648530)

[2.3.3.1. Користувацькі інтерфейси 22](#_Toc452648531)

[2.3.3.2. Програмні інтерфейси 22](#_Toc452648532)

[2.3.3.3. Комунікаційні інтерфейси 22](#_Toc452648533)

[2.3.4. Нефункціональні вимоги 22](#_Toc452648534)

[2.3.4.1. Вимоги продуктивності 22](#_Toc452648535)

[2.3.4.2. Вимоги безпеки 22](#_Toc452648536)

[2.3.4.3. Вимоги надійності 23](#_Toc452648537)

[Розділ 3. Проектування та архітектура підсистеми для формування та друку документації та звітності у ІС «Деканат» 24](#_Toc452648538)

[3.1. Архітектура програмного продукту 24](#_Toc452648539)

[3.2. Проектування бази даних 24](#_Toc452648540)

[3.2.1. Концептуальна модель бази даних 25](#_Toc452648541)

[3.2.2. Логічна модель бази даних 26](#_Toc452648542)

[3.2.3. Фізична модель бази даних 26](#_Toc452648543)

[3.3. Проектування об’єктної моделі 31](#_Toc452648544)

[3.4. Проектування графічного інтерфейсу користувача 31](#_Toc452648545)

[Розділ 4. Реалізація та тестування підсистеми формування та друку документації та звітності у ІС «Деканат» 33](#_Toc452648546)

[4.1. Реалізація підсистеми 33](#_Toc452648547)

[4.1.1. Проектування звітів 33](#_Toc452648548)

[4.1.2. Проектування інтерфейсу користувача 38](#_Toc452648549)

[4.1.3. Реалізація функціоналу системи 40](#_Toc452648550)

[4.2. Тестування системи. 41](#_Toc452648551)

[Розділ 5. Економічна частина 42](#_Toc452648552)

[5.1. Економічна характеристика проектного рішення (програмного продукту) 42](#_Toc452648553)

[5.2. Інформаційне забезпечення та формування гіпотези щодо потреби розроблення товару. 42](#_Toc452648554)

[5.3. Оцінювання та аналізування факторів зовнішнього та внутрішнього середовищ. 43](#_Toc452648555)

[5.4. Формування стратегічних альтернатив 45](#_Toc452648556)

[5.5. Бюджетування 48](#_Toc452648557)

[5.6. Остаточний вибір стратегії 53](#_Toc452648558)

[Висновки 55](#_Toc452648559)

[Список використаної літератури 56](#_Toc452648560)

[Додаток А 57](#_Toc452648561)

[Додаток Б 65](#_Toc452648562)

[Додаток В 73](#_Toc452648563)

# Анотація

# Abstract

# Вступ

Кожного дня, працівники університетів, викладачі, студенти, секретарі зустрічаються з необхідністю вирішення таких задач:

* Формування навчальних планів
* Підготовка екзаменаційних відомостей
* Формування даних щодо результатів сесії
* Створення різноманітних звітів
* Ведення електронних журналів успішності
* Формування відомостей для друку

Відділ кадрів веде особисті картки працівників та студентів.

Бухгалтерія отримує інформацію про академічну успішність, нарахування стипендій та заробітної платні.

Допомагає полегшити ці процеси, автоматизувати їх та забезпечує весь необхідний функціонал, спеціально розроблена автоматизована система управління університетом.

Звісно, об’єм даних, які обробляються такими системами дуже великий, тому, зазвичай, АСУ складають з декількох пов’язаних між собою модулів, для управлінням різними підрозділами університету (такі як деканат, бухгалтерія, відділ кадрів і т. д.), це допомагає пришвидшити роботу, та забезпечує надійність системи та цілісність даних [1].

Одним, з найбільш затребуваних модулів є такий, що дозволяє формувати документацію, звіти, різного типу довідки, зведені дані про студентів, успішність, особисту інформацію, та надає можливість збереження та друку таких документів.

З часом потрібно удосконалювати ці системи, інколи, шляхом реалізації їх на основі новіших технологій.

В результаті цього виникає потреба написання програмного продукту, який дозволить виконувати основі операції з друком та формуванням документації та звітності у інформаційній системі «Деканат», яка є підсистемою автоматизованого управління «Університет».

# Розділ 1. Огляд програмного забезпечення для автоматизованого управління університетом

## 1.1. Програмні засоби

Розробка програмного забезпечення є дорогою, тому, написання власної АСУ для кожного університету не є вигідним. У зв’язку з цим існує безліч готових рішень, які можна адаптувати до потрібного вищого навчального закладу.

Одними з таких є: АСУ «ВНЗ», Пакет програм від Політек-Софт, Автоматизована система управління вищим навчальним закладом III - IV рівня акредитації Uniteh+.

### 1.1.1. Автоматизована система управління вищим навчальним закладом III - IV рівня акредитації Uniteh+

Система складається з таких автономних модулів [2]:

* Структура ВНЗ
* WEB-сайт ВНЗ
* Навчальна частина
* Кафедра
* Навчальний розклад
* Абітурієнт
* Документообіг
* Деканат
* Кафедра
* Вчені ради ВНЗ

Останні 4 модулі на даний момент знаходяться в стані розробки.

Усі модулі працюють в локальній мережі ВНЗ, мають спільне сховище даних. Кожен модуль має серверну та клієнтську частину, доступ до яких захищено паролем.

Модуль "Структура ВНЗ " є базовим для всієї АСУ "Університет". Він моделює структуру ВНЗ і, в значній мірі, визначає зміст і якість роботи всієї АСУ "Університет".

Модуль «Навчальна частина» вирішує проблеми створення сховища бази даних, розрахунків, аналізу та звітності різним напрямкам роботи навчальної частини ВНЗ.

Модуль «Кафедра» вирішує проблеми створення сховища даних, розрахунків, аналізу та звітності по різним напрямкам роботи кафедри.

Модуль «Навчальний розклад» вирішує проблеми відображення та формування розкладу для викладачів, студентів.

Модуль "Абітурієнт " АСУ "Університет" дозволяє автоматизувати всі напрямки роботу Приймальної комісії ВНЗ, від реєстрації заяви абітурієнта до передачі особових справ зарахованих студентів до деканатів та відділу кадрів.

### 1.1.2. АСУ «ВНЗ»

Дана АСУ підходить для вищих навчальних закладів різного рівня акредитації. Використовується більше ніж 70-тьма ВНЗ, відповідає усім вимогам ISO [3].

Однією з переваг системи є повна взаємодія з програмами ЄДЕБО, УЦОЯО, ІВС «Освіта».

Складається з таких основних автоматизованих систем:

* Приймальна комісія
* Деканат
* Студмістечко

Також доступна підсистема для тестування студентів. АС студмістечко має вбудовану систему електронних платежів, для оплати навчання, проживання тощо.

Кожен навчальний заклад, який використовує АСУ "ВНЗ" може отримати Веб-розклад, а його студенти та викладачі завантажити безкоштовний мобільний додаток "АСУ ВНЗ. Розклад".

АС «Деканат» – це програмно-технологічний комплекс управління навчальним процесом закладу освіти, призначений для організації роботи методистів та зменшення кількості документації на паперових носіях.

В процесі роботи АС «Деканат», до персональних даних студента додаються інші поля, що відображають поточний стан навчання студента згідно навчального плану. Тому доцільно використовувати дану програму у комплексі з АС «Приймальна комісія».

Дані зарахованих студентів пакетами переносяться з АС «Приймальна комісія». Це позбавляє працівників деканатів від необхідності вводити руками дані студентів. Таким чином, персональна картка для кожного студента вводиться лише один раз – під час подання документів до приймальної комісії, далі картка лише набуває додаткових даних.

Складається АС «Деканат» з таких модулів:

* Студенти
* Навчальні плани
* Сесії
* Журнал
* Контракти
* Накази
* Розклад
* Результати сесії
* Відділ кадрів

Загалом система є платною, та за певну ціну університет отримує увесь набір необхідних підсистем.

### 1.1.3. Пакет програм від Політек-Софт

Програмне забезпечення ПП "Політек-СОФТ" призначене для вищих навчальних закладів України від I до IV рівнів акредитації, охоплює майже всі аспекти їх діяльності, зареєстровано в Реєстрі виробників та розповсюджувачів програмного забезпечення, має інформаційні листи від Міністерства освіти і науки, молоді та спорту України щодо використання в вищих навчальних закладах України [4].

Складається з таких пакетів програм:

* Деканат
* Абітурієнт
* Колоквіум
* Бібліограф
* Персонал

Пакет програм "Деканат"

Деканат - пакет програм, що призначений для автоматизації планування та обліку навчального процесу.

Основні можливості Пакету:

* формування даних щодо структури навчального процесу;
* формування даних щодо всіх викладачів та їх планового навантаження, розклад їх роботи;
* формування даних щодо щоденних даних про фактичну роботу кожного викладача по кожній дисципліні;
* формування великого обсягу даних щодо всіх студентів та їх успішності за весь період навчання;
* формування даних щодо наявності корпусів та аудиторій, їх заповнення, розклад занять.

Особливості Пакету:

* великий обсяг та повнота інформації, яка зберігається в базі даних;
* великий обсяг звітів, які можна підготувати на основі даних з бази даних з урахуванням вимог Міністерства освіти і науки, молоді та спорту України;
* інформаційна сумісність з іншими продуктами ПП "Політек-СОФТ".

Пакет програм "ПС-Абітурієнт

ПС-Абітурієнт - пакет програм, що призначений для автоматизації діяльності приймальної комісії.

Основні можливості Пакету:

* формування даних щодо підрозділів;
* формування даних щодо галузей знань, напрямів підготовки, спеціальностей (спеціалізацій) та їх розподілу за підрозділами;
* формування даних щодо переліку дисциплін, з яких приймаються сертифікати зовнішнього незалежного тестування або складаються вступні іспити;
* формування даних щодо кожного абітурієнта;
* формування даних щодо балів сертифікатів абітурієнтів;
* формування даних, що необхідні для щоденної передачі до системи "Конкурс".

Особливості Пакету:

* великий обсяг та повнота інформації, яка зберігається в базі даних;
* великий обсяг звітів, які можна підготувати на основі даних з бази даних;
* інформаційна сумісність з іншими продуктами ПП "Політек-СОФТ".

Пакет програм "Колоквіум"

Колоквіум - пакет програм, що призначений для автоматизації тестування студентів.

* зручна оболонка для формування тестів;
* тестування студентів в різних режимах та на основі результатів тестування можливість оцінювати їх знання;
* контроль процесу тестування в реальному часі;
* забезпечення доступу до бази даних з результатами тестування з метою їх всебічного аналізу;
* генерація звітів за результатами тестування.

Особливості Пакету:

* інтуїтивно зрозумілий інтерфейс;
* великий обсяг інформації, яка зберігається в базі даних;
* велика кількість режимів тестування і оцінювання, що дає можливість підібрати оптимальні параметри тестування для кожного конкретного випадку залежно від вимог викладача та важливості тесту;
* можливість використання в будь-яких навчальних закладах;
* інформаційна сумісність з пакетом програм "Деканат".

Пакет програм "Бібліограф"

Бібліограф - пакет програм, що призначений для для автоматизації діяльності бібліотеки.

Основні можливості Пакету:

* формування даних щодо бібліографічних описів книжок, періодичних видань, статей;
* формування даних щодо електронного варіанту книги сумарного обліку;
* формування даних щодо анкетних даних читачів;
* формування даних щодо електронного варіанту читацького квитка з інформацією щодо видачі та повернення літератури;
* формування даних щодо обігу кожного видання;
* формування даних щодо забезпеченості навчального процесу літературою;
* формування статистики розподілу видачі літератури за розділами та датою (щоденник);
* формування статистики обліку читачів та відвідувань.

Особливості Пакету:

* простота використання, мінімум необхідних дій при роботі з програмами пакету;
* великий обсяг та повнота інформації, яка зберігається в базі даних;
* пристосованість до особливостей функціонування бібліотеки в будь-якому закладі;
* інформаційна сумісність з іншими продуктами ПП "Політек-СОФТ".

Пакет програм "ПС-Персонал"

ПС-Персонал - пакет програм, що призначений для автоматизації обліку в відділі кадрів.

Основні можливості Пакету:

* формування даних щодо освіти робітника;
* формування даних щодо сімейного стану та переліку членів сім'ї робітника;
* формування даних щодо проходження атестацій та підвищення кваліфікації робітника;
* формування даних щодо службових переміщень робітника;
* формування списку наказів, згідно з якими відбувалися переміщення;
* формування даних щодо відпусток;
* формування відомостей про військовий облік;
* формування даних щодо подяк та доган, які отримав робітник.

Особливості Пакету:

* простота використання, мінімум необхідних дій при роботі з програмами пакету;
* великий обсяг та повнота інформації, яка зберігається в базі даних;
* великий обсяг звітів, які можна підготувати на основі даних з бази даних;
* інформаційна сумісність з іншими продуктами ПП "Політек-СОФТ".

# Розділ 2. Постановка задачі розробки програмного продукту для формування та друку документації та звітності у ІС «Деканат»

## 2.1. Загальна постановка задачі

Реалізувати програмний продукт, за допомогою якого можна формувати звітність, про успішність студентів, академічні довідки різних форм, зведені таблиці. Повинна бути можливість друку сформованих документів та/або збереження їх на диск у форматі .pdf. Програма повинна надавати можливість перегляду списку студентів про інститутах, групах та формування документів щодо обраного студента. Після реалізації модуль повинен бути готовим для інтеграції з існуючою ІС «Деканат».

## 2.2. Вибір інструментальних засобів

Розроблення програмного продукту здійснюватиметься в середовищі розробки -Microsoft Visual Studio . VS- серія продуктів фірми Майкрософт, які включають інтегроване середовище розробки програмного забезпечення та ряд інших інструментальних засобів. Ці продукти дозволяють розробляти як консольні програми, так і програми з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, а також веб-сайти, веб-застосунки, веб-служби .

Технології, які використовуватимуться при створенні веб-сервісу можна поділити на дві групи front-end і back-end технології, відповідно до спеціфіки модуля.

Мовою розробки було обрано C#. Версія платформи – .NET Framework 4.5, яка включає в себе усі використані технології. Внаслідок того, що C# представляє собою гібрид із декількох мов, він є таким же синтаксично чистим як і Java, майже таким ж простим, як VB, і практично таким ж потужнім та гнучким, як C++.

Для зберігання і відтворення даних використовуватиметься MS SQL Server - це надійна , захищена та стабільна платформу для роботи програмного забезпечення, важливого для бізнесу. Він дозволяє значно спростити розробку та підтримку такого програмного забезпечення, значно знижуючи затрати часу та затрати на керування даними різних рівнів. Останні версії SQL Server містять значну кількість нововведень, які дозволяють клієнтам отримати доступ до новаторських функцій керування даними, хмарних обчислень та ін.

Наведені вище технології відносяться до back-end і призначені для вирішення питань бізнес-логіки даної предметної області.

JS,HTML, CSS – базові засоби для створення користувацьких інтерфейсів веб-ресурсів і забезпечують динамічність, створення розмітки і стилів оформлення на веб- сторінцію.

Twitter Bootstrap - це чудовий фреймворк для створення сучасних, крос-браузерних і стандартизованих інтерфейсів. Продумана структура коду HTML, JavaScript і CSS дає вам можливість створювати безліч найрізноманітніших елементів інтерфейсу і сітку сайту.

## 2.3. Специфікація вимог до програмного продукту

### 2.3.1. Вступ

#### 2.3.1.1. Призначення, мета

Призначення даного продукту – формування та друк документації та звітності у ІС «Деканат».

Мета – розширити функціонал ІС «Деканат» додатковим модулем.

#### 2.3.1.2. Продукти аналоги

Аналоги:

* Автоматизована система управління вищим навчальним закладом III - IV рівня акредитації Uniteh+
* АСУ «ВНЗ»
* Пакет програм від Політек-Софт

### 2.3.2. Загальний опис

#### 2.3.2.1. Характеристики продукту

Функції, що будуть реалізовані у програмі:

* Пошук студентів про групах, інститутах;
* Формування документації та звітності;
* Збереження сформованого документу;
* Друк документу;
* Друк вже існуючого документу;
* Формування статистичних відомостей.

#### 2.3.2.2. Класи користувачів та їх характеристики

Користувачі, за рівнем доступу до системи, будуть поділятися на такі класи:

* Працівник деканату.
* Працівник деканату, відповідальний за певну групу/потік.
* Керуючий деканатом

Працівник деканату, за рівнем доступу, є найнижче, і може лише переглядати раніше видані документи та довідки, та друкувати їх.

Працівник деканату, відповідальний за певну групу чи потік, може шукати студентів, лише серед доступних йому, та формувати довідки виключно для цих студентів.

Керуючий деканатом, може виконувати усі функції, але виключно в межах інституту.

#### 2.3.2.3. Середовище функціонування

Апаратні вимоги:

– Частота процесора 1 GHz;

– Оперативна пам’ять 512 Mb;

– Процесор Intel Pentium 4 / Athlon 64 або вищі версії з підтримкою SSE2

Системні вимоги:

– Операційна система Microsoft Windows XP або вищі версії ОС;

– Необхідний принтер для можливості друку;

#### 2.3.2.4. Характеристики системи

##### 2.3.2.4.1. Пошук студентів про групах, інститутах

Опис і пріоритет:

Можливість пошуку студента, по якому буде формуватися довідка, відповідно до інституту та відповідно до академічної групи та відповідно до рівня доступу користувача.

Пріоритет – високий.

Послідовність дія\відгук:

Користувач обирає потрібний інститут з усіх існуючих, відповідно до нього появляється список академічних груп, з яких користувач обирає потрібну і тоді, відобразиться список студентів. Також повинна бути можливість пошуку студента після введення прізвища/імені.

Функціональні вимоги:

REQ-1.1: Вибір інституту

REQ-1.1: Вибір академічної групи

REQ-1.1: Вибір студента з списку

##### 2.3.2.4.2. Формування документації та звітності

Опис і пріоритет:

Після вибору студента, користувач, за умови доступу до функції формування нової довідки, вибирає який тип звіту чи яку форму довідки згенерувати, після чого у відповідному вікні відобразить згенерований документ.

Пріоритет – високий.

Послідовність дія\відгук:

Користувач підтверджує вибір студента, у полі з усіма можливими документами, вибирає необхідний, який повинен згенеруватися. Після підтвердження, користувач побачить готовий документ.

Функціональні вимоги:

REQ-2.1: Вибір студента

REQ-2.2: Вибір типу документу

REQ-2.3: Підтвердження та створення документу

##### 2.3.2.4.3. Збереження сформованого документу

Опис і пріоритет:

Після того як документ згенерується, користувач повинен мати можливість зберегти документ на диск.

Пріоритет – високий.

Послідовність дія\відгук:

Користувач переглядає згенерований документ, та , за потреби, зберігає його на диск.

Функціональні вимоги:

REQ-3.1: Користувач натискає кнопку збереження

REQ-3.2: Обирає директорію збереження

REQ-3.3: Підтверджує зберігання

##### 2.3.2.4.4. Друк документу

Опис і пріоритет:

Після того як документ згенерується, користувач повинен мати можливість роздрукувати його.

Пріоритет – високий.

Послідовність дія\відгук:

Користувач переглядає документ, та, за необхідності, друкує його.

Функціональні вимоги:

REQ-4.1: Користувач натискає кнопку друку

REQ-4.2: Обирає необхідний принтер

REQ-4.3: Підтверджує друк

##### 2.3.2.4.5. Друк вже існуючого документу

Опис і пріоритет:

Користувач, що не має доступу до формування нових довідок, обирає раніше сформований документ, та може роздрукувати його.

Пріоритет – високий.

Послідовність дія\відгук:

Користувач переглядає документ, та, за необхідності, друкує його.

Функціональні вимоги:

REQ-5.1: Користувач натискає кнопку друку

REQ-5.2: Обирає необхідний принтер

REQ-5.3: Підтверджує друк

##### 2.3.2.4.6. Друк статистичних відомостей

Опис і пріоритет:

Користувач, може обрати режим формування статистичних відомостей, та згенерувати звіт за різними параметрами.

Пріоритет – високий.

Послідовність дія\відгук:

Користувач входить в режим, обирає необхідні параметри, та формує звіт, який за необхідності може зберегти чи роздрукувати..

Функціональні вимоги:

REQ-6.1: Користувач входить в режим

REQ-6.2: Обирає необхідні параметри

REQ-6.3: Формує звіт

### 2.3.3. Вимоги зовнішніх інтерфейсів

#### 2.3.3.1. Користувацькі інтерфейси

* Інтерфейс користувача – це веб-сервіс, що відображається у всіх відомих Інтернет браузерах, які підтримують HTML 5 та CSS 3, у вигляді сайту.
* Інтерфейс користувача реалізовуються за допомогою компонування такого набору технологій - HTML/CSS/JS.
* За допомогою фреймворка Twitter Bootstrap - сторінки сайту є адаптивними до екранів різної розмірності і мобільних пристроїв.

#### 2.3.3.2. Програмні інтерфейси

Сервер веб-сайту комунікує із зовнішнім сервером бази даних:

MS SQL – Система управління базами даних, що зберігає дані сайту. Комунікація здійснюється через ORM-технологію ADO.NET Entity Framework.

#### 2.3.3.3. Комунікаційні інтерфейси

Клієнт розгортається у веб-браузері. Серверна частина розгортається на сервері IIS. Зв’язок між клієнтською і серверною частиною здійснюється за допомогою протоколу HTTP.

### 2.3.4. Нефункціональні вимоги

#### 2.3.4.1. Вимоги продуктивності

Швидкість роботи програми буде прямо залежати від кількості даних та складності звітів що будуть генеруватися.

#### 2.3.4.2. Вимоги безпеки

* Доступність функціоналу для користувача залежно від його ролі.
* Захист від SQL ін'єкцій
* Заборона використання ресурсів програми іншими клієнтами

#### 2.3.4.3. Вимоги надійності

У випадку збою роботи програми, користувач повинен отримати детальне повідомлення про зміст помилки і усі SQL-транзакції повинні відкотитись не впливаючи на роботу інших користувачів системи

# Розділ 3. Проектування та архітектура підсистеми для формування та друку документації та звітності у ІС «Деканат»

## 3.1. Архітектура програмного продукту

Відповідно до вимог підсистема повинна взаємодіяти з уже існуючою ІС «Деканат» та існуючою базою даних.

Архітектура модуля повинна бути закритою для змін і гнучкою тля можливості розширень у майбутньому. Для забезпечення цих вимог було вирішено застосувати шаблон проектування MVС, який дозволяє поділити відповідальність окремих компонент розробки (контролера, моделі і вигляду), поліпшує тестування, підвищує гнучкість.

Для розробки модуля вибрано об’єктно-орієнтовний парадигму програмування.

Клієнтська частина модуля буде виконана у стилі Single Page Application - це веб-додаток, розміщений на одній веб-сторінці, яка для забезпечення роботи завантажує усю необхідну інформацію разом із завантаженням самої сторінки. Цей підхід дозволяє створювати гнучкий користувацький інтерфейс, істотно збільшує швидкодію роботи програми за рахунок обмеженої взаємодії з серверною частиною і завантаження одного і того ж вмісту на сторінку браузера.

## 3.2. Проектування бази даних

База даних повинна бути спроектована, так, щоб дотримувалися рівні доступу, належний рівень транзакцій, був достатній рівень безпеки, відповідний рівень нормалізації. Усі транзакції в разі неуспішного виконання не повинні вносити змін у базу, а повинні відкочуватися.

База даних ІС «Деканат» знаходиться в робочому стані. Підсистема формування та друку документації та звітності буде доступатись лише до частини таблиць, а саме до тих ,які надають доступ до інформації про студентів.

Для роботи над дипломною роботою, було створено копію бази, на основі скриптів, без жодних реальних даних (Додаток А).

Доступ до таблиць здійснюється через спеціальні View, які містять настройки прав доступу.

### 3.2.1. Концептуальна модель бази даних

Для реалізації функціоналу, щодо роздруку вже існуючих довідок, було прийнято рішення спроектувати додаткові таблиці, та розшити вже існуючу базу даних.

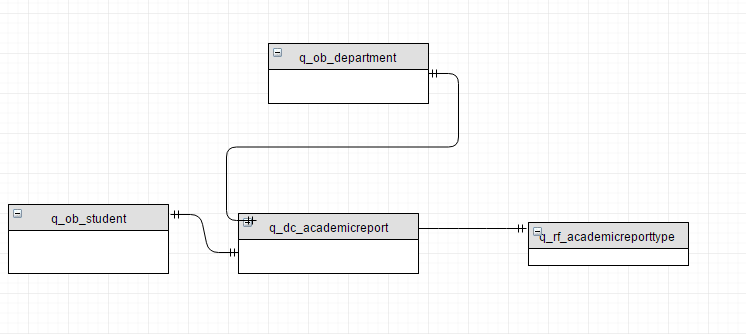


Рис. 3.1. Концептуальна модель бази даних

Щоб з легкістю відновити дані довідки, таблиця повинна мати посилання на усі необхідні дані, та також містити тип довідки.

Створення концептуальної моделі є перши, доволі загальним етапом проектування, наступний етап – логічне проектування. На ньому таблиці деталізуються полями, первинними ключами.

### 3.2.2. Логічна модель бази даних

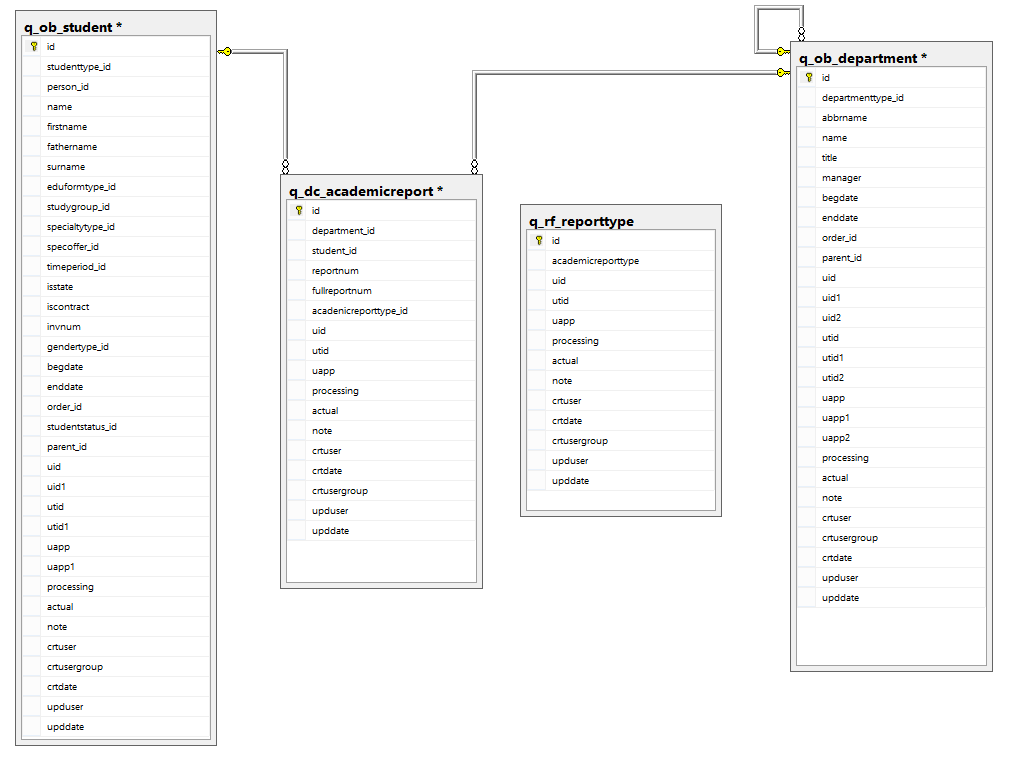


Рис. 3.2. Логічна модель бази даних

Як видно з рисунку, у таблиця «q\_dc\_academicreport» містить поле з посиланням на студента, поле з посиланням на деканат, для якого формувалась довідка, та поле з посиланням на тип довідки, а також поля з номером довідки.

Таблиця «q\_rf\_reporttype» містить назву довідки. Кожна таблиця має первинний ключ id.

### 3.2.3. Фізична модель бази даних

Створення фізичної моделі є останнім етапом при проектування бази даних. Вона більш деталізована за логічну модель, і містить в собі усі назви таблиць, типи даних полів та їх обмеження.

Слід вказати, що кожна таблиця містить в собі такі службові поля:

* uid – числове поле, може бути пустим;
* uid1 – числове поле, може бути пустим;
* utid – текстове поле, може бути пустим;
* utid1 – текстове поле, може бути пустим;
* uapp – текстове поле, може бути пустим;
* uapp1 – текстове поле, може бути пустим;
* processing – числове поле, не може бути пустим (вказує на те чи дані активні (значення поля більше 0), чи поле було видалене (від’ємне значення поля));
* actual – числове поле, не може бути пустим (вказує на актуальність даних);
* note – текстове поле, може бути пустим;
* crtuser – текстове поле, не може бути пустим (встановлюється за замовчуванням, з іменем користувача який створив запис);
* crtusergroup – текстове поле, не може бути пустим;
* crtdate – поле дати створення запису, не може бути пустим;
* upduser – текстове поле, може бути пустим (вказує на користувача, що останнім оновлював дані);
* update – поле дати, не може бути пустим;

Список основних таблиць, з якими проводилась робота:

* Таблиця «q\_rf\_reporttype» («Довідка»)
  + id – чилове поле, первинний ключ;
  + reporttype – текстове поле, тип довідки.
* Таблиця «q\_dc\_academicreport» («Довідка»)
  + id – числове поле, первинний ключ;
  + department\_id – чилове поле – посилання на інститут;
  + student\_id – чилове поле, посилання на студента;
  + reportnum – числове поле, номер довідки;
  + fullreportnum – тестове поле, згенерований код довідки.
* Таблиця «q\_ob\_student» («Студенти»)
  + id – числое поле, первинний ключ таблиці;
  + studenttype\_id – числове поле, ідентифікатор типу студента;
  + person\_id – числове поле, ідентифікатор персони;
  + firstname – текстове поле, ім’я студента;
  + fathername – текстове поле, ім’я по-батькові студента;
  + surname – текстове поле, прізвище студента;
  + eduformtype\_id – числове поле, ідентифікатор форми навчання;
  + studygroup\_id – числове поле, ідентифікатор групи студента;
  + specialtytype\_id – числове поле, ідентифікатор типу спеціальності студента (ОКР);
  + specoffer\_id – числове поле, ідентифікатор пропозиції напрямів та спеціальностей;
  + timeperiod\_id – числове поле, ідентифікатор часових періодів (вказує курс навчання; для випускників навчальний рік випуску);
  + isstate – числове поле, державна форма навчання;
  + iscontract – числове поле, комерційна форма навчання;
  + invnum – числове поле, інвентарний номер;
  + gendertype\_id – числове поле, ідентифікатор статі студента;
  + begdate – поле дати, дата зарахування студента;
  + enddate – поле дати, дата відрахування студента (завершення навчання);
  + order\_id – числове поле, ідентифікатор наказу про зарахування;
  + studentstatus\_id – числове поле, ідентифікатор статусу студента (начвчається, відрахований і т.д.);
  + parent\_id – числове поле, ієрархічний ідентифікатор;
* Таблиця «q\_qt\_studyplanitem» («Інформація про предмет з навчального плану»)
  + id – числове поле, первинний ключ таблиці;
  + studyplan\_id – числове поле, ідентифікатор заголовку начального плану;
  + studyplanpart\_id – числове поле, ідентифікатор частини навчального плану;
  + stydyplancycle\_id – числове поле, ідентифікатор циклу навчального плану;
  + studytype\_id – числове поле, ідентифікатор типу навчання;
  + markrollitem\_id – числове поле, ідентифікатор типу документу з оцінками (відомості, талону);
  + department\_id – числове поле, ідентифікатор підрозділу, який забезпечує навчальний процес по дисципліні;
  + mark – числове поле, кінцева оцінка у 100 бальній шкалі;
  + markscale\_id – числове поле, ідентифікатор шкали оцінювання;
  + markstate – числове поле, кінцева оцінка у 5-бальній шкалі;
  + markstatescale\_id – числове поле, ідентифікатор шкали оцінювання;
  + markecst – числове поле, кінцева оцінка у шкалі ECTS;
  + markscaleecst\_id – числове поле, ідентифікатор шкали оцінювання;
  + attestationnum – числове поле, номер атестації, по які отримано оцінку;
  + destsemestrtimeperiod\_id – числове поле, семестр, до якого відноситься дисципліна згідно з навчальними планами;
  + semestrtimeperiod\_id – числове поле, семестр, у якому студент насправді здає дисципліну;
  + diplomasupplementitem\_id - числове поле, ідентифікатор запису у додатку до диплома;
  + subject\_id – числове поле, ідентифікатор назви дисципліни;
  + semester – числове поле, семестр;
  + studycontrolltype\_id – числове поле, ідентифікатор типу контрольного заходу;
  + hoursall – числове поле, всього годин на вивчення;
  + credits – числове поле, всього кредитів;
  + hourslection – числове поле, всього лекційних годин;
  + hourslabs – числове поле, всього лабораторних занять;
  + hourspractical – числове поле, всього практичних занять;
  + hourshome – числове поле, всього годи на самостійне вивчення;
  + hourscontrol – числове поле, всього годин на кон6трольні заходи;
  + kkr – числове поле, кількість комплексних контрольних робіт;
  + krr – числове поле, кількість комплексних розрахункових робіт;
* Таблиця «q\_dt\_markrollitem» («Контент документу з оцінками»)
  + id – числове поле, первинний ключ таблиці;
  + markroll\_id – числове поле, ідентифікатор заголовку документа з оцінками (відомості, талона);
  + studyplanitem\_id – числове поле, ідентифікатор контенту навчального плану
  + student\_id – числове поле, ідентифікатор студента;
  + attestationnum – числове поле, номер атестації;
  + markwork1 – числове поле, бал за поточку 1;
  + markwork2 – числове поле, бал за поточку 2;
  + markexam1 – числове поле, бал за контрольний захід 1;
  + markexam2 – числове поле, бал за контрольний захід 2;
  + markexam – числове поле, бал за контрольний захід 3;
  + markpart1 – числове поле, сумарний бал за поточку1 та контрольний захід 1;
  + markpart2 – числове поле, сумарний бал за поточку2 та контрольний захід 2;
  + markpart – числове поле, сумарний бал за поточку1, поточку2, контрольний захід 1 та контрольний захід 2;
  + mark\_p1\_p2\_k3 – числове поле, сумарний бал за поточку1, поточку2 та контрольний захід 3;
  + mark – числове поле, кінцева оцінка у 100 бальній шкалі;
  + markscale\_id – числове поле, ідентифікатор шкали оцінювання;
  + markstate – числове поле, кінцева оцінка у 5-бальній шкалі;
  + markstatescale\_id – числове поле, ідентифікатор шкали оцінювання;
  + markecst – числове поле, кінцева оцінка у шкалі ECTS;
  + markscaleecst\_id – числове поле, ідентифікатор шкали оцінювання;
  + absentreason\_id – числове поле, ідентифікатор причини неявки студента на іспит (застосовується для визначення номера атестації студента);
  + markcontroll\_id – числове поле, бал останнього попереднього семестрового контролю
  + markscalecontroll\_id – числове поле, ідентифікатор шкали оцінювання для останнього (попереднього) семестрового контролю;
  + filepath – текстове поле, шлях до файлу в репозиторії;
  + filename – текстове поле, назва файлу;

## 3.3. Проектування об’єктної моделі

Для роботи з об’єктами бази даних, було використано ADO.NET Entity Framework, який автоматично згенерував модель усіх об’єктів бази даних. В основному робота проводилась через класи відображень.

Перед початком розробки було спроектовано діаграму класів системи (Додаток В).

Основні класи системи

* ReportController – інтерфейс, що містить усі методи які повинні бути реалізовані у кожному контролері, для певного репорту;
* FormTwoController – контролер, який ініціалізує дані у репорті Form2;
* FormFourController – контролер, який ініціалізує дані у репорті Form4;
* AcademicReportController – контролер, який ініціалізує дані у репорті AcademicReport;
* StudentCardController – контролер, який ініціалізує дані у репорті StudentCard;

## 3.4. Проектування графічного інтерфейсу користувача

Прототип графічного інтерфейсу було розроблено у безплатному онлайн редакторі NinjaMock. Це один з найпотужніших веб-сервісів даного типу. Великим плюсом сервісу є можливість його інтеграції з такими сервісами як Google Drive, Dropbox, Jira. Також NinjaMock надає шаблони для проектування під всі основні платформи та пристрої.



Рис. 3.3. Дизайн робочої сторінки проекту

У лівому блоці користувач бачить поля вибору студента, по інститутах, потім по групах інституту, і відповідно вже сам список студентів. Нижче розташовані елементи керування, що відповідають за те який звіт згенерується, та, при потребі, появляється додатковий елемент управління з вибором семестру навчання студента.

# Розділ 4. Реалізація та тестування підсистеми формування та друку документації та звітності у ІС «Деканат»

## 4.1. Реалізація підсистеми

### 4.1.1. Проектування звітів

Усі звіти було спроектовано у спеціальному форматі \*.rdlc, кожен з них має певний набір даних, які в ньому розміщуються, звіт може мітити як різні поля, в які можна записати певне значення, так. і таблиці які відображають певний набір даних.

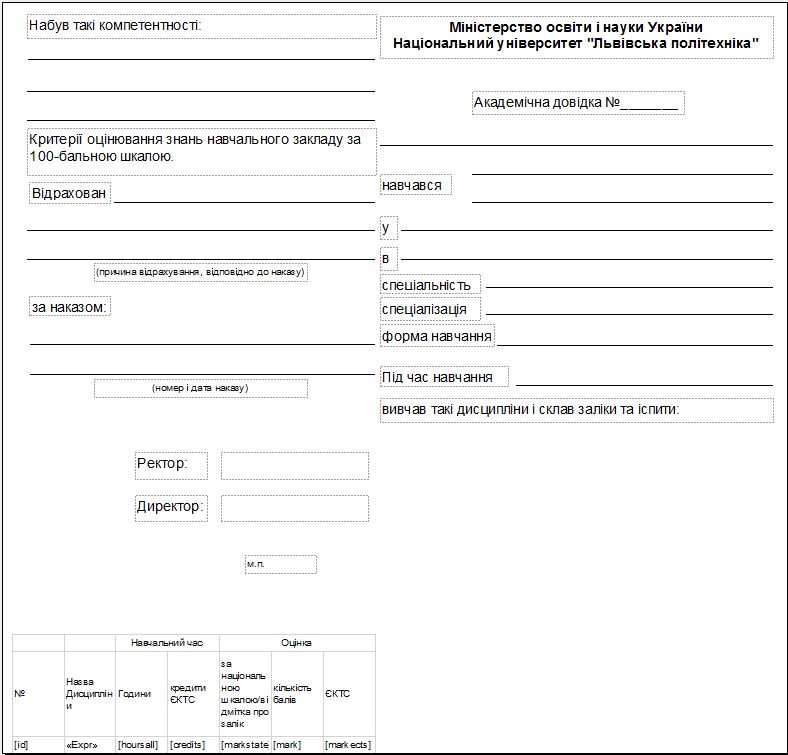


Рис. 4.1. Звіт «Академічна довідка студента»

Академічна довідка містить інформацію про студента, його ім’я, дату вступу, спеціальність, форму навчання, інститут, а також дані про всі складені ним предмети (оцінки по трьох шкалах оцінювання, години предмету).

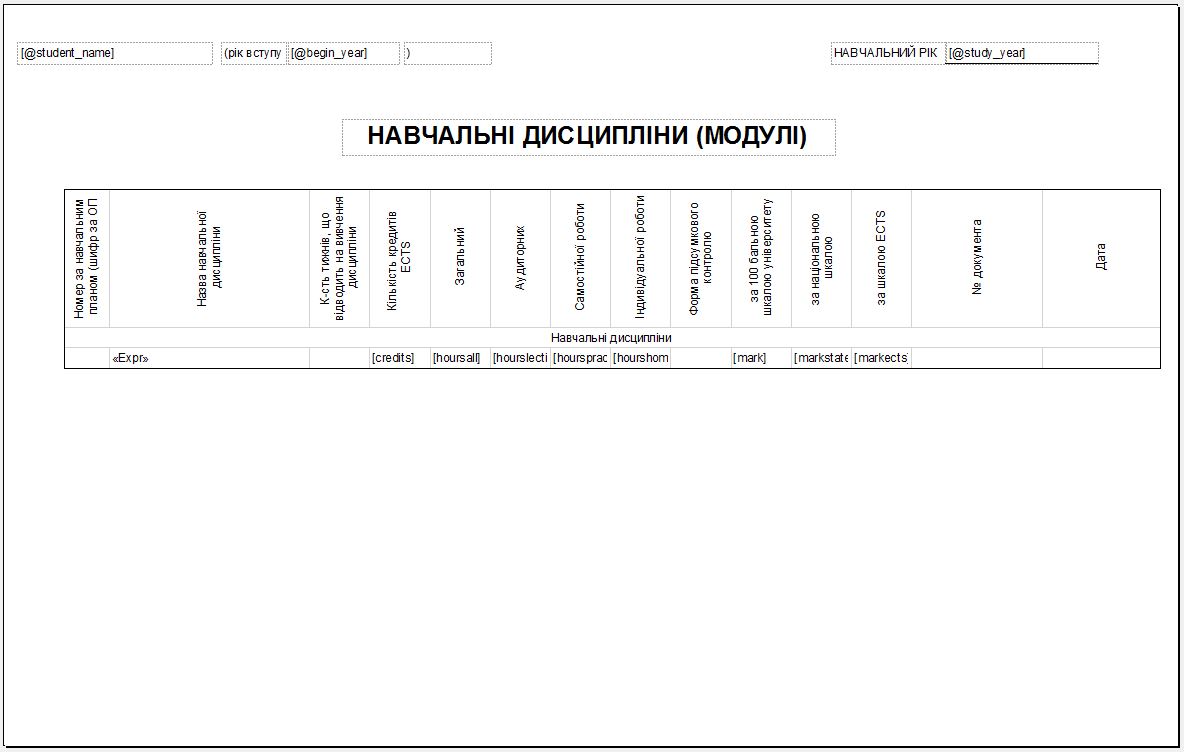


Рис. 4.2. Звіт «Форма 2»

«Форма 2» містить детальну інформацію про вивчені студентом предмети у певному семестрі, детальний поділ годин, інформацію про кредити. Дату здачі предмету, документ який відповідає інформації.

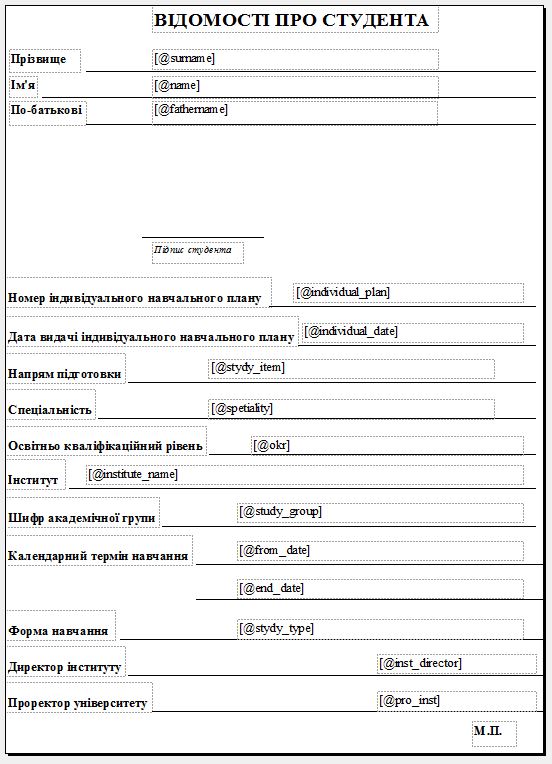


Рис. 4.3. Звіт «Форма 4»

«Форма 4» містить наступну інформацію про студента: повне ім’я, фото, напрям підготовки, спеціальність, ОКР, інститут, академічну групу, терміни та форму навчання.



Рис. 4.4. Звіт «Картка студента»

Звіт «Картка студента» містить в собі найповнішу інформацію про студента, його контактні дані, дані про навчання, накази про зарахування, та виписку з усіма вивченими предметами.

Звіти може настроювати, задаючи певні параметри, фільтри для таблиць. За стандартом, таблиця може мати встановленим лише один DataSet, для того щоб витягнути дані з інших таблиць потрібно, писати спеціальні вирази. Для прикладу:

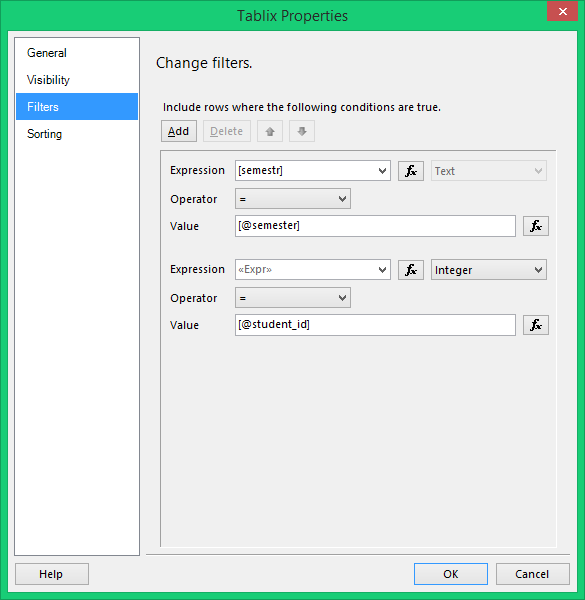


Рис. 4.5. Встановлення фільтру

Як видно з рисунку, на таблицю у «Формі 2» встановлено додаткові обмеження, які дозволяють нам бачити лише предмети певного студента, відповідно до параметру student\_id, та лише ті предмети які вивчалися у семестрі, який задається параметром semester.

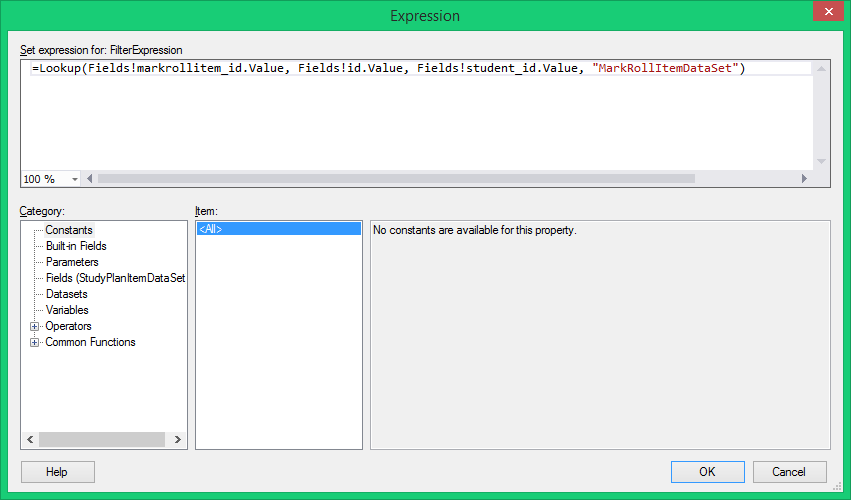


Рис. 4.6. Приєднання даних з інших таблиць

На рисунку 4.6 можна помітити, як саме здійснюється витяг даних з інших таблиць. За допомогою функції Lookup ми вказуємо поля по яких зв’язані таблиці з іншого DataSet (вказується четвертим параметром), та третім параметром вказуємо яке поле необхідно витягти.

Для кращого відображення даних, можна налаштовувати кожне поле у таблиці, під певний тип даних.

### 4.1.2. Проектування інтерфейсу користувача

Відповідно до розробленого прототипу було сформовано і основний інтерфейс користувача, який має наступний вигляд

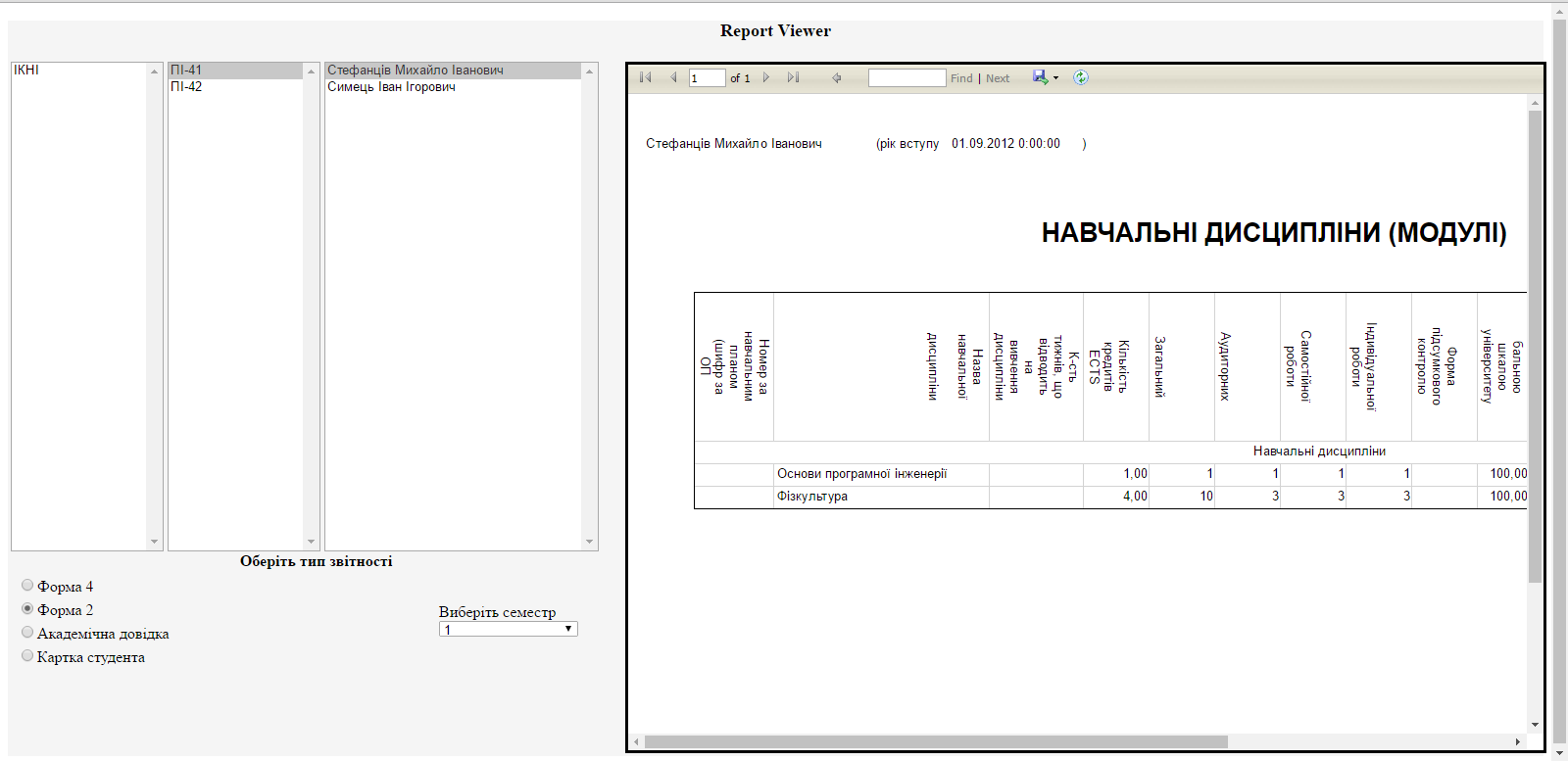


Рис. 4.7. Інтерфейс користувача

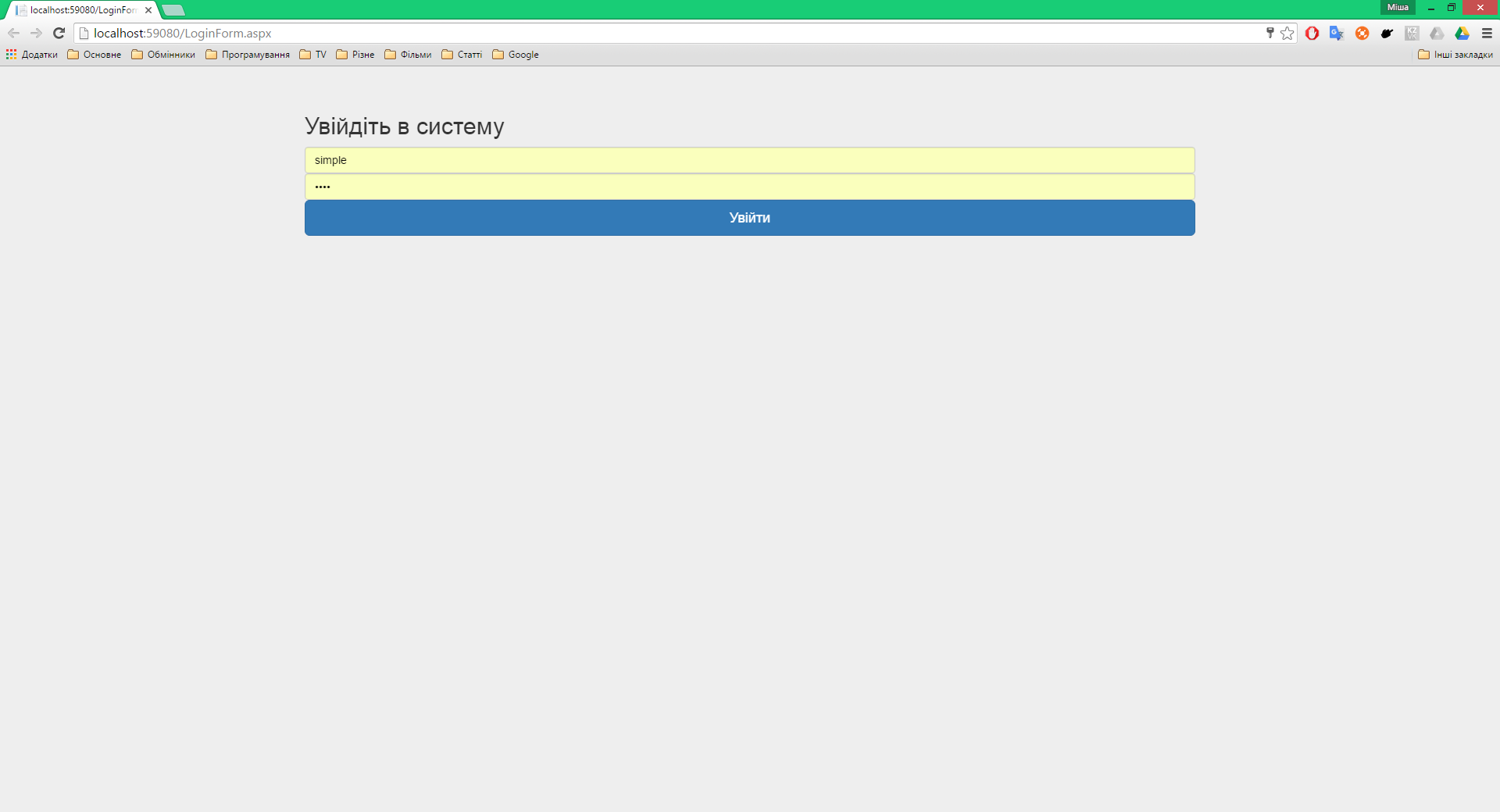


Рис. 4.8. Вікно входу в систему

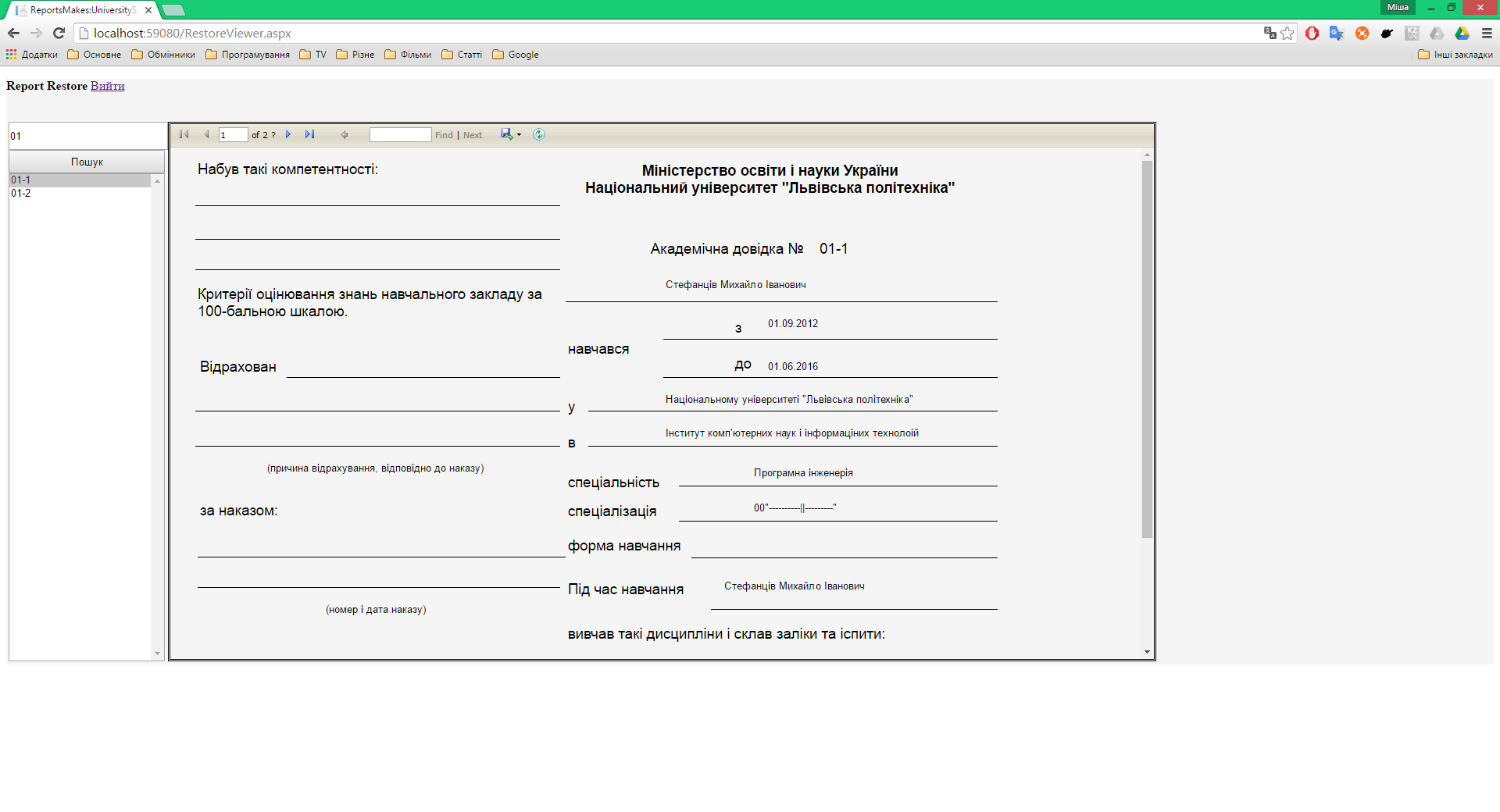


Рис. 4.9. Режим відновлення довідок

На рисунку 4.7. можна побачити як виглядає запущений проект, у якому згенеровано звіт «Форма 2» для першого семестру студента Стефанціва Михайла Івановича.

### 4.1.3. Реалізація функціоналу системи

Усі дії над елементами сторінки обробляються класами, які прив’язані до веб-форми (Додаток Б). Який в свою чергу використовує наступні класи:

**ReportController.cs**

interface ReportController

{

List<ReportParameter> setParameters(int studentId, int semester);

void initReport();

}

Містить основні функції, які повинні бути реалізовані кожним контролером для звіту.

**FormTwoController.cs**

Клас ініціалізує усі параметри звіту «Форма 2».

**FormFourController.cs**

Клас ініціалізує усі параметри звіту «Форма 4».

**AcademicReportController.cs**

Клас ініціалізує усі параметри звіту «Академічна довідка».

**AcademicReportRestoreController.cs**

Клас створено для реалізації функціоналу, пов’язаного з відновленням уже створених довідок. Він перевіряє чи є вже створені довідки, за певни номером, та при наявності, дозволяє відновити її.

**StudentCardController.cs**

Клас ініціалізує усі параметри звіту «Картка студента».

**PageLoader.cs**

namespace UniversityDiploma

{

public class PageLoader

{

public static int loadTimes = 0;

public static int restoreMode = 0;

public static String num;

}

}

Клас містить три службові параметри, для перевірки режиму роботи, для управління завантаженням списків, та параметр номеру довідки.

## 4.2. Тестування системи.

# Розділ 5. Економічна частина

## 5.1. Економічна характеристика проектного рішення (програмного продукту)

Дана бакалаврська кваліфікаційна робота передбачала розробуку підсистеми формування та друку документації та звітності у ІС «Деканат». Після завершення розробки отримуємо готовий програмний продукт, що дозволяє формувати різного типу довідки та зведені таблиці з даними студентів та їхньою успішністю.

Даний програмний продукт призначений для працівників деканатів університету, він повинен інтегрувати у вже існуючу інформаційну систему.

Розроблення даного модуля є економічно вигідне та доцільне, оскільки існуючий модуль застарілий, а аналогічні модулі на ринку не готові до інтеграції з уже існуючою системою.

## 5.2. Інформаційне забезпечення та формування гіпотези щодо потреби розроблення товару.

Зараз на ринку автоматизованих інформаційних систем для управління університетом, українського виробництва, є лише декілька продуктів, які можна використати для навчального закладу, вони платні, та вони не можуть інтегрувати з існуюючою системою.

Ось декілька із них:

* Автоматизована система управління вищим навчальним закладом III - IV рівня акредитації Uniteh+
  + Система складається з набору модулів. Усі модулі працюють в локальній мережі ВНЗ, мають спільне сховище даних. Кожен модуль має серверну та клієнтську частину, доступ до яких захищено паролем.
* АСУ «ВНЗ»
  + Дана АСУ підходить для вищих навчальних закладів різного рівня акредитації. Використовується більше ніж 70-тьма ВНЗ, відповідає усім вимогам ISO [3]. Однією з переваг системи є повна взаємодія з програмами ЄДЕБО, УЦОЯО, ІВС «Освіта».
* Пакет програм від Політек-Софт
  + Програмне забезпечення ПП "Політек-СОФТ" призначене для вищих навчальних закладів України від I до IV рівнів акредитації, охоплює майже всі аспекти їх діяльності, зареєстровано в Реєстрі виробників та розповсюджувачів програмного забезпечення, має інформаційні листи від Мiнiстерства освiти і науки, молоді та спорту України щодо використання в вищих навчальних закладах України [4].

Проаналіхувавши ринок автоматизованих систем управління університетом, я виявив потребу у розробці модуля формування та друку документації та звітності у ІС «Деканат».

## 5.3. Оцінювання та аналізування факторів зовнішнього та внутрішнього середовищ.

Результати оцінок та аналізу впливу факторів зовнішнього та внутрішнього середовищ на організацію наведено у табл. 5.1.

Таблиця 5.1

Результати експертних оцінок

|  |  |  |  |
| --- | --- | --- | --- |
| Фактори | Середня експертна оцінка, бали | Середня вагомість факторів | Зважений рівень впливу, бали |
| 1 | 2 | 3 | 4 |
| *Фактори зовнішнього середовища* |  | | |
| Споживачі | 4 | 0,11 | 0,44 |
| Постачальники | 0 | 0,1 | 0 |
| Конкуренти | 0 | 0,1 | 0,3 |
| Державні органи влади | 0 | 0,05 | 0 |
| Інфраструктура | 1 | 0,06 | 0,3 |
| Законодавчі акти | 0 | 0,1 | 0 |
| Профспілки, партії та інші громадські організації | 0 | 0,05 | 0 |
| Система економічних відносин в державі | 1 | 0,06 | 0,06 |
| Організації-сусіди | -2 | 0,01 | -0,02 |
| Міжнародні події | 0 | 0,01 | 0 |
| Міжнародне оточення | 1 | 0,03 | 0,03 |
| Науково-технічний прогрес | 3 | 0,07 | 0,21 |
| Політичні обставини | 0 | 0,06 | 0 |
| Соціально-культурні обставини | 1 | 0,05 | 0,1 |
| Рівень техніки та технологій | 3 | 0,04 | 0,12 |
| Особливості міжнародних економічних відносин | 0 | 0,02 | 0 |
| Стан економіки | 0 | 0,08 | 0 |
| Загальна сума | — | 1 | 0,70 |

*Продовження табл. 5.1*

*Фактори внутрішнього середовища*

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Цілі | 2 | 0,11 | 0,22 |
| Структура | 4 | 0,16 | 0,64 |
| Завдання | 3 | 0,07 | 0,21 |
| Технологія | 2 | 0,2 | 0,4 |
| Працівники | 4 | 0,21 | 0,84 |
| Ресурси | 3 | 0,25 | 0,75 |
| Загальна сума | — | 1 | 3,06 |

Одже, в результаті проведення оцінки та аналізу зовнішнього та внутрішнього середовища були зроблені такі висновки.

Споживачі та науковово-технічний прогрес є зовнішніми факторами, що мають найбільший вплив. Все більше і більше вищих навчальних закладів намагаються осучаснити свої інформаційні системи, особливо, це стосується осучаснення ведення документації.

Щодо внутрішніх факторів, то найбільший вплив на продук мають структура, працівники та ресурси, що логічно, оскільки продукт має пряме відношення до цих пунктів.

## 5.4. Формування стратегічних альтернатив

Стратегічні альтернативи, група №1.

Критеріями поділу альтернативних стратегій розвитку є існуючий продукт (програмний засіб) та новий, а також відповідні послуги.



Рис. 5.1. Стратегічні альтернативи, група №1

Стратегія розробки нового продукту (проектного рішення) - це стратегія, яка характеризується реалізацією абсолютно нового програмного забезпечення, якого немає на ринку і яке дає змогу вирішувати певні потреби користувача.

Стратегії розвитку нового продукту з супутніми послугами –це підхід, який включає в себе розробку нового програмного рішення і додаткові послуги.

Стратегіїя розвитку існуючого програмного продукту – це стратегія, яка передбачає модифікації та удосконалення програмного продукту.

Стратегія розвитку наявного на ринку продукту (проектного рішення) з супутніми послугами – це стратегія, яка передбачає удосконалення та оптимізацію програмного забезпечення із додатковими послугами, такими як встановлення, супроводження тощо.

Стратегічні альтернативи, група №2.

Основними критеріями, за якими поділяються альтернативні стратегії другої групи це існуючий або новий продукт (проектне рішення), а також наявний чи відсутній ринок.



Рис. 5.2. Група стратегічних альтернатив №2

Стратегія глибокого проникнення на ринок (концентрація) - загальна маркетингова стратегія, яка полягає в знаходженні шляхів збільшення збуту своїх товарів, що випускаються на вже освоєних ринках, та за допомогою більш агресивного маркетингу.

Стратегія розвитку товару передбачає збільшення обсягів збуту завдяки вдосконаленню наявних і розробці нових товарів для наявних ринків. Щодо ризикованості, то дана стратегія займає не низьку позицію.

**Стратегія розвитку ринку передбачає збільшення обсягу збуту завдяки виходу на новий ринок фірми з наявним товаром.** При цьому використовуються дві альтернативи - вихід на нові географічні ринки або орієнтація на нові сегменти ринку.

Стратегія диверсифікації передбачає розробку нових товарів для нових ринків. Зазначимо, що ця стратегія є найбільш ризиковою, разом з тим дозволяє перерозподілити ризики завдяки діяльності фірми на різних товарних ринках.

Проаналізувавши обидві групи стратегій, було обрано по одній стратегій, з кожної групи, які підходять для даного програмного продукту.

З першої групи було обрано стратегію розвитку нового продукту зі супетніми послугами, оскільки потрібно реалізувати продукт який буде забезпечувати увесь необхідний функціонал, а також потібно забезпечити його підтримку надалі.

У другій групі найкращою стратегією є стратегі ярозвитку товару, оскільки продукт може розвиватися, розширюватися, а також модуль може в майбутньому інтегрувати з іншими системами.

## 5.5. Бюджетування

Бюджетування – це необхідна складова фінансового планування, оскільки основою будь-якого оперативного чи поточного фінансового плану є система відповідних бюджетів. Як і поточне планування в цілому, бюджетування спрямоване на поетапну трансформацію стратегічного фінансового плану в систему поточних планів, послідовне їх виконання з метою досягнення стратегічних цілей підприємства.

Таблиця 5.2

Бюджет витрат матеріалів та комплектуючих виробів

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва матеріалів та комплектуючих | Марка, тип, модель | Фактична кількість, шт. | Ціна за одиницю, грн. | Разом, грн. |
| Ноутбук | ASUS VX7SX | 1 | 9 100 | 9 100 |
| Середовище розробки | MS Visual Studio Premium  2012 | 1 | 8000 | 8000 |
| Разом: | | | | 17 100 |

Таблиця 5.3

Бюджет витрат на оплату праці

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Посада,  спеціальність | Кількість працівників, осіб | Час роботи, дні | Денна заробітна плата працівників, грн. | Сума витрат на оплату праці, грн. |
| Основна заробітна плата | | | | |
| Програміст | 1 | 14 | 400 | 5 600 |
| Тестер | 1 | 5 | 200 | 1 000 |
| Разом: | | | | 6 000 |

Таблиця 5.4

Бюджет загальновиробничих витрат

|  |  |
| --- | --- |
| Статті витрат | Сума, грн. |
| Змінні загальновиробничі витрати, у т.ч.: | |
| Заробітна плата допоміжного персоналу | 800 |
| Витрати на МШП | 120 |
| Витрати на електроенергію та технологічні цілі | 180 |
| Разом змінних витрат: | 1100 |
| Постійні загальновиробничі витрати, у т.ч.: |  |
| Комунальні послуги | 520 |
| Витрати на оренду | 2100 |
| Разом постійних витрат: | 2620 |
| Разом загальновиробничих витрат: | 3720 |

Таблиця 5.5

Бюджет адміністративних витрат та витрат на збут

|  |  |
| --- | --- |
| Статті витрат | Сума, грн. |
| Адміністративні витрати, у т.ч.: | |
| Витрати на МШП | 200 |
| Витрати на сплату податків і зборів | 450 |
| Знос адміністративного обладнання | 200 |
| Разом адміністративних витрат: | 800 |
| Витрати на збут, у т.ч.: |  |
| Заробітна плата менеджерів зі збуту | 2000 |
| Витрати на рекламу | 2100 |
| Разом витрат на збут: | 4100 |

Таблиця 5.6

Зведений кошторис витрат на розробку проектного рішення продукту

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Статті витрат | Одиниці виміру | Фактична кількість, шт. | Ціна одиниці, грн. | Разом, грн. |
| 1 | 2 | 3 | 4 | 5 |
| Сировина і матеріали | - | - | - | - |
| Комплектуючі вироби | грн | 3 | - | 17 100 |

Продовження табл. 5.7

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| Зворотні відходи | - | - | - | - |
| Паливо та електроенергія на технологічні цілі | кВт/год | 400 | 0,45 | 180 |
| Основна заробітна плата | грн | - | - | 6000 |
| Додаткова заробітна плата | грн | - | - | - |
| Відрахування на соціальне страхування | % | 36,6 | - | 2196 |
| Витрати на утримання й експлуатацію устаткування | грн | - | - | - |
| Загальновиробничі витрати, у т.ч.: | грн | - | - | - |
| - змінні; | грн | - | - | 1100 |
| - постійні; | грн | - | - | 3720 |
| *Разом виробничих витрат:* | грн | - | - | 13196 |
| Адміністративні витрати | грн | - | - | 800 |
| Витрати на збут | грн | - | - | 4100 |
| Інші операційні витрати | грн | - | - | - |
| *Разом виробничих і операційних витрат:* | грн | - | - | 18096 |

Щоб визначити фінансові результати, нас потрібно обчислити ціну розроблюваного продукту. Вона визначається на основі суми операційних та виробничих видатків із взяттям до уваги рентабельності виробництва.

Ц = СБ \* Р.

де Ц – ціна одинці продукту, грн.

СБ – собівартість продукту, грн.

Р – рентабельність виробництва, 30%

Ц = 18096\* 0,30 = 5 430 грн.

Таблиця 5.8

Бюджет фінансових результатів

|  |  |
| --- | --- |
| Показники | Сума, грн. |
| Дохід від реалізації продукції (10 шт) | 54 300 |
| Податок на додану вартість (20%) | 10 860 |
| Чистий дохід від реалізації продукції | 43 440 |
| Собівартість реалізованої продукції | 18 096 |
| Валовий прибуток | 25 344 |
| Операційні витрати | |
| Адміністративні витрати | 800 |
| Витрати на збут | 4 100 |
| Фінансовий результат від операційної діяльності | 20 444 |
| Податок на прибуток (18%) | 3 679 |
| Чистий прибуток (збиток) | 16 769 |

## 5.6. Остаточний вибір стратегії

Провівши економічний аналіз доцільності розробки програмного продукту, можна зробити висновок, що розробка є прибутковую. Стратегією розробки було вибрано стратегію розробки нового продуку з супутніми послугами.

Даний продукт має основного кінцевого користувача – Національний університет «Львівська політехніка». Проте, модуль має перспективи до розвитку та розширення, та може знайти свою нішу на ринку серед автоматизованих систем управління.

Отже, виконавши бюджетування продукту, було встановлено собівартість продукту, яка становить 18 096 грн.  Планується реалізація 10 примірників модуля і відповідно до обрахунків бюджету чистий прибуток становитиме 16 769 грн при коефіцієнті рентабельності 30%.

# Висновки

Під час роботи я ознайомився з аналогічними системами,проаналізував їх, ознайомився з структурою існуючої ІС «Деканат».

Ознайомився із середовищами розробки MS Visual Studio 2012 та СУБД MS SQL Server 2014. Навчився налаштовувати їх. Поглибив свої знання у вивченні мови C#, а особливо технології ASP.NET MVC.

Навчився складати план робіт та слідував за ним. Виконуючи роботу я намагався максимально ефективно використовувати свій час та виконувати її як найкраще.

# Список використаної літератури

1. <https://uk.wikipedia.org/wiki/Автоматизована_система_керування_технологічним_процесом>
2. <http://www.unitex.com.ua/products/commercial-software/automated-system-for-higher-education-institution/>
3. <http://vuz.osvita.net/ua/>
4. <http://www.politek-soft.kiev.ua/>

# Додаток А

q\_ob\_student

CREATE TABLE [dbo].[q\_ob\_student](

[id] [bigint] NOT NULL,

[studenttype\_id] [bigint] NOT NULL,

[person\_id] [bigint] NOT NULL,

[name] [varchar](200) NOT NULL,

[firstname] [varchar](200) NULL,

[fathername] [varchar](200) NULL,

[surname] [varchar](200) NULL,

[eduformtype\_id] [bigint] NOT NULL,

[studygroup\_id] [bigint] NULL,

[specialtytype\_id] [bigint] NOT NULL,

[specoffer\_id] [bigint] NULL,

[timeperiod\_id] [bigint] NOT NULL,

[isstate] [int] NULL,

[iscontract] [int] NULL,

[invnum] [varchar](50) NULL,

[gendertype\_id] [bigint] NOT NULL,

[begdate] [datetime] NULL,

[enddate] [datetime] NULL,

[order\_id] [bigint] NULL,

[studentstatus\_id] [bigint] NOT NULL,

[parent\_id] [bigint] NULL,

[uid] [bigint] NULL,

[uid1] [bigint] NULL,

[utid] [varchar](50) NULL,

[utid1] [varchar](50) NULL,

[uapp] [varchar](50) NULL,

[uapp1] [varchar](50) NULL,

[processing] [int] NOT NULL,

[actual] [int] NOT NULL,

[note] [varchar](250) NULL,

[crtuser] [varchar](50) NOT NULL,

[crtusergroup] [varchar](50) NOT NULL,

[crtdate] [datetime] NOT NULL,

[upduser] [varchar](50) NOT NULL,

[upddate] [datetime] NULL,

CONSTRAINT [PK\_q\_ob\_student] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

q\_dt\_studyplanitem

CREATE TABLE [dbo].[q\_dt\_studyplanitem](

[id] [bigint] NOT NULL,

[studyplan\_id] [bigint] NOT NULL,

[studyplanpart\_id] [bigint] NOT NULL,

[studyplancycle\_id] [bigint] NOT NULL,

[studytype\_id] [bigint] NOT NULL,

[markrollitem\_id] [bigint] NULL,

[department\_id] [bigint] NOT NULL,

[mark] [decimal](10, 4) NULL,

[markscale\_id] [bigint] NULL,

[markstate] [decimal](10, 4) NULL,

[markscalestate\_id] [bigint] NULL,

[markects] [varchar](50) NULL,

[markscaleects\_id] [bigint] NULL,

[attestationnum] [int] NULL,

[destsemestrtimeperiod\_id] [bigint] NOT NULL,

[semestrtimeperiod\_id] [bigint] NULL,

[diplomasupplementitem\_id] [bigint] NULL,

[subject\_id] [bigint] NOT NULL,

[semestr] [int] NULL,

[studycontroltype\_id] [bigint] NOT NULL,

[hoursall] [int] NOT NULL,

[credits] [decimal](15, 2) NOT NULL,

[hourslection] [int] NOT NULL,

[hourslabs] [int] NOT NULL,

[hourspractical] [int] NOT NULL,

[hourshome] [int] NOT NULL,

[hourscontrol] [int] NOT NULL,

[kkr] [int] NOT NULL,

[krr] [int] NOT NULL,

[qualificationworkthema] [varchar](400) NULL,

[parent\_id] [bigint] NULL,

[uid] [bigint] NULL,

[utid] [varchar](50) NULL,

[uapp] [varchar](50) NULL,

[uid1] [bigint] NULL,

[utid1] [varchar](50) NULL,

[uapp1] [varchar](50) NULL,

[processing] [int] NOT NULL,

[actual] [int] NOT NULL,

[note] [varchar](250) NULL,

[crtuser] [varchar](50) NOT NULL,

[crtusergroup] [varchar](50) NOT NULL,

[crtdate] [datetime] NOT NULL,

[upduser] [varchar](50) NOT NULL,

[upddate] [datetime] NULL,

CONSTRAINT [PK\_q\_dt\_studyplanitem] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

q\_dt\_markrollitem

CREATE TABLE [dbo].[q\_dt\_markrollitem](

[id] [bigint] NOT NULL,

[markroll\_id] [bigint] NOT NULL,

[studyplanitem\_id] [bigint] NULL,

[student\_id] [bigint] NOT NULL,

[attestationnum] [int] NOT NULL,

[markwork1] [decimal](10, 4) NULL,

[markwork2] [decimal](10, 4) NULL,

[markexam1] [decimal](10, 4) NULL,

[markexam2] [decimal](10, 4) NULL,

[markexam] [decimal](10, 4) NULL,

[markpart1] [decimal](10, 4) NULL,

[markpart2] [decimal](10, 4) NULL,

[markpart] [decimal](10, 4) NULL,

[mark\_p1\_p2\_k3] [decimal](10, 4) NULL,

[mark] [decimal](10, 4) NULL,

[markscale\_id] [bigint] NOT NULL,

[markstate] [decimal](10, 4) NULL,

[markscalestate\_id] [bigint] NOT NULL,

[markects] [decimal](10, 4) NULL,

[markscaleects\_id] [bigint] NOT NULL,

[absentreason\_id] [bigint] NULL,

[markcontrol] [decimal](10, 4) NULL,

[markscalecontrol\_id] [bigint] NULL,

[filepath] [varchar](200) NULL,

[filename] [varchar](200) NULL,

[parent\_id] [bigint] NULL,

[uid] [bigint] NULL,

[utid] [varchar](50) NULL,

[uapp] [varchar](50) NULL,

[processing] [int] NOT NULL,

[actual] [int] NOT NULL,

[note] [varchar](250) NULL,

[crtuser] [varchar](50) NOT NULL,

[crtusergroup] [varchar](50) NULL,

[crtdate] [datetime] NULL,

[upduser] [varchar](50) NOT NULL,

[upddate] [datetime] NULL,

CONSTRAINT [PK\_q\_dt\_markrollitem] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

q\_ob\_department

CREATE TABLE [dbo].[q\_ob\_department](

[id] [bigint] NOT NULL,

[departmenttype\_id] [bigint] NOT NULL,

[abbrname] [varchar](50) NULL,

[name] [varchar](200) NOT NULL,

[title] [varchar](200) NULL,

[manager] [varchar](50) NULL,

[begdate] [datetime] NULL,

[enddate] [datetime] NULL,

[order\_id] [bigint] NULL,

[parent\_id] [bigint] NULL,

[uid] [bigint] NULL,

[uid1] [bigint] NULL,

[uid2] [bigint] NULL,

[utid] [varchar](50) NULL,

[utid1] [varchar](50) NULL,

[utid2] [varchar](50) NULL,

[uapp] [varchar](50) NULL,

[uapp1] [varchar](50) NULL,

[uapp2] [varchar](50) NULL,

[processing] [int] NOT NULL,

[actual] [int] NOT NULL,

[note] [varchar](255) NULL,

[crtuser] [varchar](50) NOT NULL,

[crtusergroup] [varchar](50) NOT NULL,

[crtdate] [datetime] NOT NULL,

[upduser] [varchar](50) NOT NULL,

[upddate] [datetime] NULL,

CONSTRAINT [PK\_q\_ob\_department] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

q\_dc\_academicreport

use [ReportServiceDB]

BEGIN TRANSACTION

SET QUOTED\_IDENTIFIER ON

SET ARITHABORT ON

SET NUMERIC\_ROUNDABORT OFF

SET CONCAT\_NULL\_YIELDS\_NULL ON

SET ANSI\_NULLS ON

SET ANSI\_PADDING ON

SET ANSI\_WARNINGS ON

COMMIT

BEGIN TRANSACTION

GO

CREATE TABLE dbo.q\_dc\_academicreport

(

id bigint NOT NULL,

department\_id bigint NOT NULL,

student\_id bigint NOT NULL,

reportnum bigint NOT NULL,

fullreportnum varchar(50) NOT NULL,

acadenicreporttype\_id bigint NOT NULL,

uid bigint NULL,

utid varchar(50) NULL,

uapp varchar(50) NULL,

processing int NULL,

actual int NULL,

note varchar(50) NULL,

crtuser varchar(50) NULL,

crtdate datetime NULL,

crtusergroup varchar(50) NULL,

upduser varchar(50) NULL,

upddate datetime NULL

) ON [PRIMARY]

GO

ALTER TABLE dbo.q\_dc\_academicreport ADD CONSTRAINT

PK\_q\_dc\_academicreport PRIMARY KEY CLUSTERED

(

id

) WITH( STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

GO

ALTER TABLE q\_dc\_academicreport SET (LOCK\_ESCALATION = TABLE)

GO

COMMIT

q\_rf\_reporttype

use [ReportServiceDB]

BEGIN TRANSACTION

SET QUOTED\_IDENTIFIER ON

SET ARITHABORT ON

SET NUMERIC\_ROUNDABORT OFF

SET CONCAT\_NULL\_YIELDS\_NULL ON

SET ANSI\_NULLS ON

SET ANSI\_PADDING ON

SET ANSI\_WARNINGS ON

COMMIT

BEGIN TRANSACTION

GO

CREATE TABLE dbo.q\_rf\_reporttype

(

id int NOT NULL,

academicreporttype varchar(50) NOT NULL,

uid bigint NULL,

utid varchar(50) NULL,

uapp varchar(50) NULL,

processing int NULL,

actual int NULL,

note varchar(50) NULL,

crtuser varchar(50) NULL,

crtdate datetime NULL,

crtusergroup varchar(50) NULL,

upduser varchar(50) NULL,

upddate datetime NULL

) ON [PRIMARY]

GO

ALTER TABLE dbo.q\_rf\_reporttype ADD CONSTRAINT

PK\_q\_rf\_reporttype PRIMARY KEY CLUSTERED

(

id

) WITH( STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

GO

ALTER TABLE dbo.q\_rf\_reporttype SET (LOCK\_ESCALATION = TABLE)

GO

COMMIT

# Додаток Б

Default.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Data;

using System.Web.UI.WebControls;

using Microsoft.Reporting.WebForms;

using UniversityDiploma.ReportControllers;

namespace UniversityDiploma

{

public partial class \_Default : Page

{

ReportServiceDBEntities entities = new ReportServiceDBEntities();

FormTwoController formTwoController = new FormTwoController();

FormFourController formFourControlelr = new FormFourController();

AcademicReportController academicReportController = new AcademicReportController();

StudentCardController studentCardController = new StudentCardController();

private void loadInstitutesList()

{

PageLoader.loadTimes = 1;

List<String> institutesList;

if (LoginController.institute\_id == 0)

institutesList = entities.ob\_department.Where(x => x.departmenttype\_id == 2).Select(x => x.abbrname).ToList();

else

institutesList = entities.ob\_department.Where(x => x.departmenttype\_id == 2 && x.id == LoginController.institute\_id).Select(x => x.abbrname).ToList();

foreach (var item in institutesList)

{

InstitutesBox.Items.Add(item);

}

}

protected void Page\_Load(object sender, EventArgs e)

{

if (PageLoader.loadTimes == 0)

loadInstitutesList();

}

protected void ListBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

var instituteNameId = InstitutesBox.SelectedIndex;

var instituteName = InstitutesBox.Items[instituteNameId].ToString();

var instituteId = entities.ob\_department.Where(inst => inst.abbrname == instituteName).Select(x => x.id).ToList(); ;

var convertedId = Convert.ToInt32(instituteId[0].ToString());

var groupsList = entities.ob\_studygroup.Where(group => group.department\_id == convertedId).Select(x => x.name).ToList();

var distinctGroupsList = groupsList.Distinct();

GroupsBox.Items.Clear();

foreach (var item in distinctGroupsList)

{

GroupsBox.Items.Add(item);

}

}

protected void GroupsBox\_SelectedIndexChanged(object sender, EventArgs e)

{

var groupNameId = GroupsBox.SelectedIndex;

var groupName = GroupsBox.Items[groupNameId].ToString();

var groupId = entities.ob\_studygroup.Where(g => g.name == groupName).Select(x => x.id).FirstOrDefault();

var convertedId = Convert.ToInt32(groupId.ToString());

var studentsList = entities.ob\_student.Where(s => s.studygroup\_id == convertedId).Select(x => x.name).ToList();

StudentsBox.Items.Clear();

foreach (var item in studentsList)

{

StudentsBox.Items.Add(item);

}

}

protected void StudentsBox\_SelectedIndexChanged(object sender, EventArgs e)

{

var studentNameId = StudentsBox.SelectedIndex;

var studentName = StudentsBox.Items[studentNameId].ToString();

var studentId = entities.ob\_student.Where(s => s.name == studentName).Select(x => x.id).ToList();

var convertedId = Convert.ToInt32(studentId[0].ToString());

List<ReportParameter> list = new List<ReportParameter>();

if (RadioButtonList1.SelectedIndex == 0)

{

list = formFourControlelr.setParameters(convertedId, 0);

ReportViewer1.LocalReport.SetParameters(list);

}

else if (RadioButtonList1.SelectedIndex == 1)

{

var semester = DropDownList1.Items[DropDownList1.SelectedIndex].Value.ToString();

list = formTwoController.setParameters(convertedId, Convert.ToInt32(semester));

ReportViewer1.LocalReport.SetParameters(list);

}

else if (RadioButtonList1.SelectedIndex == 2)

{

list = academicReportController.setParameters(convertedId, 0);

ReportViewer1.LocalReport.SetParameters(list);

}

else if (RadioButtonList1.SelectedIndex == 3)

{

list = studentCardController.setParameters(convertedId, 0);

ReportViewer1.LocalReport.SetParameters(list);

}

else

{

ClientScript.RegisterStartupScript(this.GetType(), "myalert", "alert('Оберіть тип звітності');", true);

}

}

protected void RadioButtonList1\_SelectedIndexChanged(object sender, EventArgs e)

{

if (RadioButtonList1.SelectedIndex == 0)

{

ReportViewer1.LocalReport.ReportPath = Server.MapPath("~/Form4.rdlc");

}

else if (RadioButtonList1.SelectedIndex == 1)

{

ReportViewer1.LocalReport.ReportPath = Server.MapPath("~/Form2.rdlc");

DropDownList1.Visible = true;

DropLabel.Visible = true;

}

else if (RadioButtonList1.SelectedIndex == 2)

{

ReportViewer1.LocalReport.ReportPath = Server.MapPath("~/AcademicReport.rdlc");

}

else

{

ReportViewer1.LocalReport.ReportPath = Server.MapPath("~/StudentCard.rdlc");

}

}

}

}

ReportRestore.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using Microsoft.Reporting.WebForms;

using UniversityDiploma.ReportControllers;

namespace UniversityDiploma

{

public partial class Restore\_Viewer : System.Web.UI.Page

{

AcademicReportController academicReportController = new AcademicReportController();

ReportServiceDBEntities entities = new ReportServiceDBEntities();

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void Button1\_Click(object sender, EventArgs e)

{

ListBox1.Items.Clear();

var number = TextBox1.Text;

var foundList = entities.dc\_academicreporttype.Where(x => x.fullreportnum.Contains(number) || x.fullreportnum == number).Select(x => x.fullreportnum).ToList();

foreach (var item in foundList)

{

ListBox1.Items.Add(item);

}

}

protected void ListBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

PageLoader.restoreMode = 1;

var itemSelected = ListBox1.SelectedIndex;

var itemName = ListBox1.Items[itemSelected].ToString();

var report = entities.dc\_academicreporttype.Where(x => x.fullreportnum == itemName).Select(x => x.student\_id).FirstOrDefault();

PageLoader.num = itemName;

List<ReportParameter> list = academicReportController.setParameters(Convert.ToInt32(report), 0);

ReportViewer1.LocalReport.SetParameters(list);

}

}

}

FormTwoController.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using Microsoft.Reporting.WebForms;

namespace UniversityDiploma.ReportControllers

{

public class FormTwoController : ReportController

{

private ReportServiceDBEntities entities = new ReportServiceDBEntities();

public List<ReportParameter> setParameters(int studentId, int semester)

{

List<ReportParameter> list = new List<ReportParameter>();

var student = entities.ob\_student.Where(x => x.id == studentId).ToList().FirstOrDefault();

DateTime beginDate = (DateTime) student.begdate;

list.Add(new ReportParameter("student\_name", student.name, true));

list.Add(new ReportParameter("begin\_year", beginDate.Year.ToString(), true));

list.Add(new ReportParameter("study\_year", beginDate.Year.ToString(), true));

list.Add(new ReportParameter("semester", semester.ToString(), true));

list.Add(new ReportParameter("student\_id", studentId.ToString(), true));

return list;

}

public void initReport()

{

throw new NotImplementedException();

}

}

}

FormFourController.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using Microsoft.Reporting.WebForms;

namespace UniversityDiploma.ReportControllers

{

public class FormFourController : ReportController

{

private ReportServiceDBEntities entities = new ReportServiceDBEntities();

public List<Microsoft.Reporting.WebForms.ReportParameter> setParameters(int studentId, int semester)

{

List<ReportParameter> list = new List<ReportParameter>();

var student = entities.ob\_student.Where(x => x.id == studentId).ToList().FirstOrDefault();

var okr = entities.rf\_specialtytype.Where(x => x.id == student.specialtytype\_id).Select(x => x.name).FirstOrDefault();

var group = entities.ob\_studygroup.Where(x => x.id == student.studygroup\_id).Select(x => x.name).FirstOrDefault();

var groupId = entities.ob\_studygroup.Where(x => x.id == student.studygroup\_id).Select(x => x.id).FirstOrDefault();

var institute = entities.ob\_department.Where(x => x.id == groupId).Select(x => x.title).FirstOrDefault();

var studyplanNum = entities.dc\_studyplan.Where(x => x.student\_id == studentId).Select(x => x.docnum).FirstOrDefault();

var studyplanDate = entities.dc\_studyplan.Where(x => x.student\_id == studentId).Select(x => x.docdate).FirstOrDefault();

var instituteDirector = entities.ob\_department.Where(x => x.id == groupId).Select(x => x.manager).FirstOrDefault();

var specOffer = entities.ob\_specoffer.Where(x => x.id == student.specoffer\_id).Select(x => x.specialty\_id).FirstOrDefault();

var study\_item = entities.ob\_specialty.Where(x => x.id == specOffer).Select(x => x.name).FirstOrDefault();

var study\_item\_number = entities.ob\_specialty.Where(x => x.id == specOffer).Select(x => x.cipher).FirstOrDefault();

DateTime beginDate = (DateTime) student.begdate;

DateTime endDate = (DateTime) student.enddate;

DateTime studyplanShortDate = (DateTime) studyplanDate;

list.Add(new ReportParameter("name", student.firstname, true));

list.Add(new ReportParameter("surname", student.surname, true));

list.Add(new ReportParameter("fathername", student.fathername, true));

list.Add(new ReportParameter("from\_date", beginDate.Date.ToShortDateString(), true));

list.Add(new ReportParameter("end\_date", endDate.Date.ToShortDateString(), true));

list.Add(new ReportParameter("okr", okr.ToString(), true));

list.Add(new ReportParameter("institute\_name", institute.ToString(), true));

list.Add(new ReportParameter("study\_group", group.ToString(), true));

list.Add(new ReportParameter("individual\_plan", studyplanNum.ToString(), true));

list.Add(new ReportParameter("individual\_date", studyplanShortDate.ToShortDateString(), true));

list.Add(new ReportParameter("inst\_director", instituteDirector.ToString(), true));

list.Add(new ReportParameter("stydy\_item", study\_item\_number + " " + study\_item, true));

return list;

}

public void initReport()

{

throw new NotImplementedException();

}

}

}

AcademicReportController.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using Microsoft.Reporting.WebForms;

namespace UniversityDiploma.ReportControllers

{

public class AcademicReportController : ReportController

{

private ReportServiceDBEntities entities = new ReportServiceDBEntities();

private AcademicReportRestoreController restoreController = new AcademicReportRestoreController();

public List<Microsoft.Reporting.WebForms.ReportParameter> setParameters(int studentId, int semester)

{

List<ReportParameter> list = new List<ReportParameter>();

var student = entities.ob\_student.Where(x => x.id == studentId).ToList().FirstOrDefault();

var group = entities.ob\_studygroup.Where(x => x.id == student.studygroup\_id).Select(x => x.name).FirstOrDefault();

var groupId = entities.ob\_studygroup.Where(x => x.id == student.studygroup\_id).Select(x => x.id).FirstOrDefault();

var institute = entities.ob\_department.Where(x => x.id == groupId).Select(x => x.title).FirstOrDefault();

var instituteId = entities.ob\_department.Where(x => x.id == groupId).Select(x => x.id).FirstOrDefault();

var instituteDirector = entities.ob\_department.Where(x => x.id == groupId).Select(x => x.manager).FirstOrDefault();

var specOffer = entities.ob\_specoffer.Where(x => x.id == student.specoffer\_id).Select(x => x.specialty\_id).FirstOrDefault();

var study\_item = entities.ob\_specialty.Where(x => x.id == specOffer).Select(x => x.name).FirstOrDefault();

var study\_item\_number = entities.ob\_specialty.Where(x => x.id == specOffer).Select(x => x.cipher).FirstOrDefault();

DateTime fromDate = (DateTime) student.begdate;

DateTime endDate = (DateTime) student.enddate;

list.Add(new ReportParameter("student\_id", student.id.ToString(), true));

list.Add(new ReportParameter("student\_name", student.name, true));

list.Add(new ReportParameter("university", "Національному університеті \"Львівська політехніка\"", true));

list.Add(new ReportParameter("institute", institute.ToString(), true));

list.Add(new ReportParameter("manager", instituteDirector.ToString(), true));

list.Add(new ReportParameter("fromDate", fromDate.Date.ToShortDateString(), true));

list.Add(new ReportParameter("endDate", endDate.Date.ToShortDateString(), true));

list.Add(new ReportParameter("speciality", study\_item, true));

String num;

if (PageLoader.restoreMode == 0)

num = restoreController.saveGeneratedReport(Convert.ToInt32(instituteId), studentId);

else

num = entities.dc\_academicreporttype.Where(x => x.fullreportnum == PageLoader.num).Select(x => x.fullreportnum).FirstOrDefault();

list.Add(new ReportParameter("num", num, true));

return list;

}

public void initReport()

{

throw new NotImplementedException();

}

}

}

AcademicReportRestore.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace UniversityDiploma.ReportControllers

{

public class AcademicReportRestoreController

{

private ReportServiceDBEntities entities = new ReportServiceDBEntities();

public String saveGeneratedReport(int departmentId, int studentId) {

var maxId = entities.dc\_academicreporttype.Max(x => x.id);

var lastNumber = entities.dc\_academicreporttype.Where(x => x.id == maxId).Select(x => x.reportnum).FirstOrDefault();

String firstNumberPart = "";

var newReport = new q\_dc\_academicreport();

newReport.crtdate = DateTime.UtcNow;

newReport.department\_id = departmentId;

newReport.student\_id = studentId;

newReport.reportnum = Convert.ToInt32(lastNumber) + 1;

if (departmentId < 10)

{

firstNumberPart = "0" + departmentId.ToString();

}

else

{

firstNumberPart = departmentId.ToString();

}

String secondNumberPart = "";

if (newReport.reportnum < 10)

secondNumberPart = "000";

else secondNumberPart = "00";

secondNumberPart += newReport.reportnum.ToString();

newReport.fullreportnum = firstNumberPart + "-" + secondNumberPart;

newReport.id = maxId + 1;

entities.q\_dc\_academicreport.Add(newReport);

entities.SaveChanges();

return newReport.fullreportnum;

}

}

}

StudentCardController.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using Microsoft.Reporting.WebForms;

namespace UniversityDiploma.ReportControllers

{

public class StudentCardController : ReportController

{

ReportServiceDBEntities entities = new ReportServiceDBEntities();

public List<ReportParameter> setParameters(int studentId, int semester)

{

List<ReportParameter> list = new List<ReportParameter>();

var student = entities.ob\_student.Where(x => x.id == studentId).ToList().FirstOrDefault();

var group = entities.ob\_studygroup.Where(x => x.id == student.studygroup\_id).Select(x => x.name).FirstOrDefault();

var specOffer = entities.ob\_specoffer.Where(x => x.id == student.specoffer\_id).Select(x => x.specialty\_id).FirstOrDefault();

var study\_item = entities.ob\_specialty.Where(x => x.id == specOffer).Select(x => x.name).FirstOrDefault();

var study\_item\_number = entities.ob\_specialty.Where(x => x.id == specOffer).Select(x => x.cipher).FirstOrDefault();

var personInfo = entities.ob\_person.Where(x => x.id == student.person\_id).Select(x => new { x.marriedtype\_id, x.citizencountry\_id, x.docseries, x.docnum, x.birthplace, x.begdate }).FirstOrDefault();

DateTime birth = (DateTime) personInfo.begdate;

if (personInfo.citizencountry\_id != null)

{

list.Add(new ReportParameter("nationality", "має", true));

}

else

{

list.Add(new ReportParameter("nationality", "-", true));

}

var married = entities.rf\_marriedtype.Where(x => x.id == personInfo.marriedtype\_id).Select(x => x.name).FirstOrDefault();

var home\_phone = entities.od\_personcontact.Where(x => x.person\_id == student.person\_id && x.contacttype\_id == 1).Select(x => x.strvalue).FirstOrDefault();

var cell\_phone = entities.od\_personcontact.Where(x => x.person\_id == student.person\_id && x.contacttype\_id == 2).Select(x => x.strvalue).FirstOrDefault();

var email = entities.od\_personcontact.Where(x => x.person\_id == student.person\_id && x.contacttype\_id == 4).Select(x => x.strvalue).FirstOrDefault();

list.Add(new ReportParameter("student\_name", student.name, true));

list.Add(new ReportParameter("specialty", study\_item + ", " + study\_item\_number, true));

list.Add(new ReportParameter("specialization", "00\"-----||-----\"", true));

list.Add(new ReportParameter("study\_group", group.ToString(), true));

list.Add(new ReportParameter("birth\_date", birth.ToShortDateString(), true));

list.Add(new ReportParameter("birth\_place", personInfo.birthplace.ToString(), true));

list.Add(new ReportParameter("family\_status", married.ToString(), true));

list.Add(new ReportParameter("school\_degree", "-", true));

list.Add(new ReportParameter("work", "-", true));

list.Add(new ReportParameter("home\_address", personInfo.birthplace.ToString(), true));

list.Add(new ReportParameter("passport", personInfo.docnum + personInfo.docseries, true));

list.Add(new ReportParameter("home\_phone", home\_phone.ToString(), true));

list.Add(new ReportParameter("cell\_phone", cell\_phone.ToString(), true));

list.Add(new ReportParameter("email", email.ToString(), true));

return list;

}

public void initReport()

{

throw new NotImplementedException();

}

}

}

# Додаток В

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | | | |
| Виконав: | Стефанців М.І. |  |  | UML діаграма класів | Сторінка 1 з 1 |
| Перевірив: | Квятковський Б.О. |  |  |
| ІКНІ кафедра ПЗ ПІ-41 | | | |