НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. Ігоря Сікорського»

Кафедра

Автоматизованих систем обробки інформації і управління

**КУРСОВА РОБОТА**

з технологій розробки програмного забезпечення

на тему: «Біржа праці: ведення особових справ безробітних в центрі

зайнятості»

|  |  |  |
| --- | --- | --- |
| Керівник: Крамар Ю. М. |  | Виконала:  Мощицька Ольга  студентка гр. ІС-83, ФІОТ  3 курс  № IС-8316  Підпис: \_\_\_\_\_\_\_\_\_\_ |
| Допущена до захисту  «\_\_\_» \_\_\_\_\_\_\_\_ 2020 \_\_\_\_\_\_\_\_\_\_  підпис |  |
| Захистила з оцінкою  \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  оцінка підпис  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 |  |

Київ 2020

**ЛИСТ ЗАВДАННЯ**

Національний технічний університет України “КПІ ім. І. Сікорського”

(назва вищого навчального закладу)

Кафедра автоматизованих систем обробки інформації і управління

Дисципліна [Технології розробки програмного забезпечення](http://wiki.kpi.ua/index.php/%D0%A2%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D1%96%D1%97%20%D1%80%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B8%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F%20-%202.%20%D0%A2%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D1%96%D1%97%20%D1%80%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B8%20WEB-%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%B2%D0%B0%D0%BD%D1%8C%20%D0%BD%D0%B0%20%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D1%96%20Microsoft.NET/Java/Node.js_(20202020))

Напрям "Інформаційні системи та технології"

Курс 3 Група ІС-83 Семестр 5

**ЗАВДАННЯ**

**на курсову роботу студента**

Мощицька Ольга Миколаївна

(прізвище, ім’я, по батькові)

1. Тема роботи Біржа праці: ведення особових справ безробітних в центрі

зайнятості\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Строк здачі студентом закінченої роботи 24.12.2020

3. Вхідні дані до роботи база даних

4. Вихідні дані до роботи сторінка веб-сервісу

5. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці) Постановка завдання, опис завдання, опис шарів проекту, розробка програмного забезпечення

6. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

7. Дата видачі завдання 22.10.2020

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Назва етапів курсової роботи | Термін виконання етапів роботи | Підписи керівника, студента |
| 1. | Отримання теми курсової роботи | 22.10.2020 |  |
| 2. | Підготовка ТЗ | 25.10.2020 |  |
| 3. | Пошук та вивчення літератури з питань курсової роботи | 26.10.2020 |  |
| 4. | Розробка сценарію роботи програми | 01.11.2020 |  |
| 6. | Узгодження сценарію роботи програми з керівником | 02.11.2020 |  |
| 5. | Розробка (вибір) алгоритму рішення задачі | 12.11.2020 |  |
| 6. | Узгодження алгоритму з керівником | 14.11.2020 |  |
| 7. | Узгодження з керівником інтерфейсу користувача | 17.11.2020 |  |
| 8. | Розробка програмного забезпечення | 20.11.2020 |  |
| 9. | Налагодження розрахункової частини програми | 25.11.2020 |  |
| 10. | Розробка та налагодження інтерфейсної частини програми | 27.11.2020 |  |
| 11. | Узгодження з керівником набору тестів для контрольного прикладу | 04.12.2020 |  |
| 12. | Тестування програми | 07.12.2020 |  |
| 13. | Підготовка пояснювальної записки | 15.12.2020 |  |
| 14. | Здача курсової роботи на перевірку | 20.12.2020 |  |
| 15. | Захист курсової роботи | 24.12.2020 |  |

Студент Мощицька О. М.

(підпис) (прізвище, ім’я, по батькові)

Керівник Крамар Ю.М.

(підпис) (прізвище, ім’я, по батькові)

" " 20 р.

Оглавление

[Вступ 4](#_Toc58882282)

[1 Постановка задачі 5](#_Toc58882283)

[2 Опис реалізації проекту 7](#_Toc58882284)

[2.1 Опис завдання 7](#_Toc58882285)

[3 Опис шарів проекту 9](#_Toc58882286)

[3.1 Загальний опис шарів проекту 9](#_Toc58882287)

[3.2 Шар доступу до даних (DAL) 9](#_Toc58882288)

[3.2.1 Моделі даних 9](#_Toc58882289)

[3.2.2 Класи, призначені для взаємодії з моделями даних 9](#_Toc58882290)

[3.3 Шар бізнес-логіки (BLL) 10](#_Toc58882291)

[3.3.1 Об’єкти 10](#_Toc58882292)

[3.4 Шар представлення (Web) 11](#_Toc58882293)

[3.4.1 Моделі 11](#_Toc58882294)

[3.4.2 Контролери 11](#_Toc58882295)

[Висновки 13](#_Toc58882296)

[Перелік посилань 14](#_Toc58882297)

**АНОТАЦІЯ**

Пояснювальна записка до курсової роботи: 16 сторінок, 3 рисунків, 5 посилань.

Об’єкт дослідження: технологія ASP.NET MVC .Net Framework 4.7 та багатошарові застосунки.

Мета роботи: розробка веб-застосунку, що реалізує ведення особових справ безробітних в центрі зайнятості. У теоретичних викладках було розглянуто теоретичні відомості про ASP.NET MVC .Net Framework 4.7, Entity Framework та технологію створення баз даних Code First, опис шаблонів Репозиторій (Repository) та Одиниця Роботи (Unit of Work).

В описі програмного забезпечення було наведено інформацію про багатошарові веб-додатки та особливості їхньої архітектури.

Вивчено ряд функцій технології ASP.NET MVC та практично закріплено їх використання при написанні програмного коду курсової роботи.

Виконана програмна реалізація веб-застосунку, що вказана у завданні.

Вступ

Метою курсової роботи є розробка програмного забезпечення на основі технологій веб-інтерфейсу з використанням технологій ASP.NET MVC .Net Framework 4.7.

Завданням курсової роботи було створити веб-застосунок мовою С#, використовуючи технології ASP.NET MVC .Net Framework 4.7 та Entity Framework Core, що реалізує ведення особових справ безробітних в центрі зайнятості.

Згідно вимог поставленого завдання, розроблена програмна система повинна мати багатошарову архітектуру, де шарами архітектури є:

* шар (рівень) доступу до даних;
* шар бізнес-логіки;
* шар представлення (інтерфейс програмної системи).

ASP.NET MVC .NET Framework 4.7 – це фреймворк для сучасної платформи .Net Framework 4.3, що використовується для розробки різних веб-застосунків. В основі цього фреймворку лежить популярний паттерн MVC (Model-View-Controller) - архітектурний шаблон, який використовується під час проектування та розробки програмного забезпечення.

Мета шаблону — гнучкий дизайн програмного забезпечення, який повинен полегшувати подальші зміни чи розширення програм, а також надавати можливість повторного використання окремих компонентів програми. Крім того використання цього шаблону у великих системах сприяє впорядкованості їхньої структури і робить їх більш зрозумілими за рахунок зменшення складності.

Необхідно зазначити, що ми можемо також писати і клієнтську частину, використовуючи мову розмітки HTML, стилі CSS, а також JavaScript, що допомагає нам реалізувати якісь дії та обробники на клієнтській стороні аби розвантажити сервер. Є можливість підключення різних зовнішніх бібліотек для роботи з клієнтською частиною, а саме bootstrap, vue.js, angular, ReactJs, JQuery. Підключити ці бібліотеки ми можемо за допомогою технології CDN, або завантажити ці бібліотеки в каталог застосунку, за домомогою NPM, Bower, Nuget.

# Постановка задачі

Спроектувати та реалізувати систему класів, в основу якої покладено логічну структуру даних, наведену у варіанті, для накопичення та обробки даних домену відповідно варіанту курсової роботи. Варіант: «Біржа праці: ведення особових справ безробітних в центрі зайнятості».

Код програмної системи має складатись не менш як з трьох частин відповідно до багатошарової архітектури системи, де шарами архітектури є:

1. Шар доступу до даних (DAL) організувати таким чином, щоб він забезпечував збереження даних у БД із застосуванням Entity Framework. Тип програмного модулю – DLL. У проектуванні шару використати шаблони Репозиторій та Одиниця роботи.
2. В шарі бізнес-логіки (BLL) побудувати класи, що представляють об’єкти та дії над ними відповідно до предметної області. Якщо дані потрібно отримати зі сховища даних або зберегти у сховищі, шар бізнес-логіки звертається до шару доступу до даних, передаючи/отримуючи збережені дані. Тип програмного модулю – DLL.
3. Шар представлення (PL) – це додаток типу відповідного до обраного інтерфейсу (ASP.NET MVC чи Web API). Для виконання дій на вимогу користувача шар представлення звертається до шару бізнес-логіки, передаючи в нього уведені користувачем дані та команди на виконання певних дії. Результати виконаних дій, отримані від шару бізнес-логіки, шар представлення відображує користувачу у представленнях (web-сторінках). Тип програмного модулю – web-застосування.

Реалізувати обов’язковий контроль коректності уведених даних в шарі представлення.

Розроблене програмне забезпечення повинне відповідати наступним функціональним вимогам:

1. Управління вакансіями і резюме

1.1. Можливість додавати вакансію і резюме до категорії

1.2. Можливість видаляти вакансію і резюме з категорії

1.3. Можливість змінювати дані вакансій і резюме

1.4. Можливість переглянути дані конкретної вакансії і резюме

1.5. Можливість переглянути відсортований список всіх вакансій

1.6. Можливість переглянути відсортований список всіх резюме

2. Управління безробітними

2.1. Можливість додавати безробітних

2.2. Можливість видаляти безробітних

2.3. Можливість змінювати дані безробітних

2.4. Можливість переглянути дані конкретного безробітного

2.5. Можливість переглянути список всіх безробітних

2.5.1. Можливість відсортувати список по імені

2.5.2. Можливість відсортувати список по прізвищу

3. Управління фірмами замовниками

3.1. Можливість додавати замовників

3.2. Можливість видаляти замовників

3.3. Можливість змінювати дані замовників

3.4. Можливість переглянути дані конкретного замовника

3.5. Можливість переглянути список всіх замовників

3.5.1. Можливість відсортувати список по імені

3.5.2. Можливість відсортувати список по прізвищу

4. Пошук

4.1. Можливість пошуку по ключовому слову серед вакансій

4.2. Можливість пошуку по ключовому слову серед безробітних

# Опис реалізації проекту

## Опис завдання

За умовою поставленої задачі, необхідно було розробити веб-застосунок із багатошаровою (multilayer) архітектурою.

Окрім звичайних переваг модульного програмного забезпечення з чітко визначеними інтерфейсами, трьохрівнева архітектура призначена для того, щоб дозволити модернізувати будь-який з трьох рівнів.

Класична трьохрівнева система складається із наступних рівнів (рис. 3.1).:

|  |
| --- |
|  |
| Рисунок 3.1 Багатошарова архітектура |

* Presentation layer (рівень представлення) – це рівень, з яким безпосередньо взаємодіє користувач. Цей рівень включає компоненти користувацького інтерфейсу, механізм отримання вводу від користувача. Стосовно до ASP.NET MVC на даному рівні розташовані всі представлення і всі ті компоненти, які складають користувацький інтерфейс (стилі, статистичні сторінки html, javascript), а також моделі представлень, контролери, об’єкти контексту запиту.
* Business layer (рівень бізнес-логіки) – це рівень, який містить набір компонентів, що відповідають за обробку отриманих від рівня представлення даних, реалізує всю необхідну логіку застосунку, всі обчислення, взаємодіє з базою даних і передає рівню представлення результат обробки.
* Data Access layer (рівень доступу до даних) – це рівень, який зберігає моделі, що описують сутності, що використовуються, також тут розміщені специфічні класи для роботи з різними технологіями доступу до даних, наприклад, клас контексту даних Entity Framework. Тут також зберігаються репозиторії, через які рівень бізнес-логіки взаємодіє з базою даних.

При цьому важливим є те, що крайні рівні не можуть взаємодіяти між собою, тобто рівень представлення (стосовно ASP.NET MVC, контролери) не можуть напряму звертатись до бази даних і навіть до рівня доступу до даних, а тільки через рівень бізнес-логіки.

Рівень доступу до даних не залежить від інших рівнів, рівень бізнес-логіки залежить від рівня доступу до даних, а рівень представлення – від рівня бізнес‑логіки.

Компоненти, як правило, повинні бути слабозв’язними (loose coupling), тому невід’ємною ланкою багатошарових застосунків є впровадження залежностей.

У даній курсовій роботі кожен логічний рівень є окремим проектом. Всі проекти об’єднані в одному рішенні і представляють собою цілісне програмне забезпечення, функціонал якого утворює єдину логічну ланку.

# Опис шарів проекту

## Загальний опис шарів проекту

Програмний продукт складається з 3 проектів, два з яких – бібліотеки класів (DAL та BLL), та ще один – ASP.NET Web Application (PL), які є обов’язковими згідно поставленого завдання.

3.1 Шар доступу до даних (DAL)

Даний шар програми представлений DLL модулем (DAL.dll).

Він містить у собі усі моделі даних, які необхідно зберігати в БД, а також класи, призначені для взаємодії з цими даними.

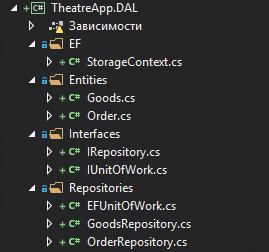
### Моделі даних

### Класи, призначені для взаємодії з моделями даних

Були використані шаблони Репозиторій та Одиниця роботи. Таким чином були розроблені класи IRepository(інтерфейс, призначений для роботи з різними репозиторіями проекту містить у собі методи додавання, зміни, видалення, отримання даних та збереження змін у базі), IUnitOfWork (інтерфейс, призначений для роботи з Unit of Work, через цей інтерфейс надається доступ до репозиторії, які містяться у Unit of Work), UnitOfWork( клас, що реалізує IUnitOfWor через нього буде відбуватись уся взаємодія з репозиторіями. Містить у собі усі репозиторії проекту, контекст даних та метод збереження змін у базі даних).Підключення до бази даних відбувається за допомогою класу Context. Він пов’язує усі моделі з сутностями в базі даних.

Приклад кінцевої структури проекту, що реалізує рівень даних, виглядає наступним чином (рис.4.1):

Рисунок 4.1 Приклад структура проекту DAL з лабораторних



## Шар бізнес-логіки (BLL)

Даний шар програми представлений BLL модулем (BLL.dll).

Він представляє собою набір сервісів, які надають певну логіку для представлення. Також він має свої моделі (Data Transfer Object), які містять тільки логіку, без зайвих компонентів. За допомогою цих моделей відбувається передача інформації на рівень представлення.

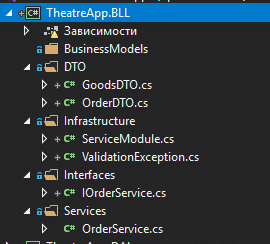
### Об’єкти

Рівень бізнес логіки має бути незалежним, так як і попередні рівні, тому він оперує власними DTO (Data transfer object) моделями та зв’язаний з DAL рівнем через інтерфейси. Саме це йому дозволяє бути незалежним і при зміні представлення або рівню доступу до даних, ми його можем не змінювати. Класи, які описують бізнес-логіку.

Для реалізації бізнес-логіки проекту були створені «сервіси» - класи, які містять у собі методи для обробки даних, отриманих від користувача, перед їх додаванням у базу. А також спрощення даних, отриманих з бази перед передачею їх до користувача.

Приклад кінцевої структури проекту, що реалізує рівень бізнес-логіку, виглядає наступним чином (рис.4.5):

Рисунок 4.5 Приклад структури проекту BLL з лабораторних



## Шар представлення (Web)

Даний рівень представляє собою web-проект.

Він містить у собі моделі, контролери та представлення, які реалізують інтерфейс додатку, видимий користувачу. Також містить функціонал, який забезпечує валідацію даних, додавання, редагування та видалення даних з бази.

### Моделі

Моделі, дублюють моделі з шару BLL. Крім цього, створені моделі, що є буфером під час редагування об’єктів. Додатково в моделях прописані умови валідації даних, їх необхідність, відображення на сайті.

### Контролери

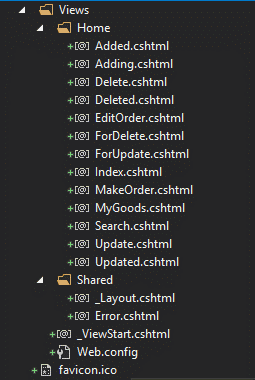
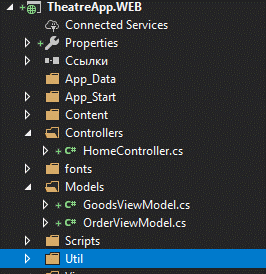
Шар містить 1 контролер, який відповідає за свою логічну частину програми.

1. HomeController. Містить методи для реалізації відображення домашньої сторінки.

Шар містить представлення, які відповідають кожному методу контролера, розбиті по папкам з назвою контролера.

Приклад кінцевої структури проекту, що реалізує рівень представлення, зображена на рисунку 4.9:

Рисунок 4.9 Приклад структури проекту PL із лабораторних



Висновки

У ході виконання курсової роботи було проаналізовано методи розробки веб-додатків з багатошаровою архітектурою.

Також було створено такий веб-застосунок з використанням технологій ASP.NET MVC .Net Framework.

На практиці було реалізовано програмний продукт, що реалізує роботу сервісу ведення особових справ безробітних в центрі зайнятості. Даний додаток розв’язує поставлену в курсовій роботі задачу у повному обсязі. На етапі тестування було визначено, що програма вірно обробляє всі вхідні дані та видає очікуваний результат.

Результатом курсової роботи є створений веб-додаток, який можна використовувати для електронного обліку біржі праці.

По закінченню написання програми було створено пояснювальну записку до даної курсової роботи.

Таким чином, я, реалізувавши всі вищезгадані завдання, розробила програмне забезпечення із графічним веб-інтерфейсом з використанням технологій ASP.NET MVC, що повністю задовольняє мету курсової роботи.

Перелік посилань

* 1. Adam Freeman «Pro ASP.NET MVC 5 Platform» // Apress. – 2014. - 428 ст.
  2. ASP.NET MVC. Урок 0. Вступление [Електронні ресурси] // Режим доступу: https://habr.com/post/175999/. Перевірено: 04.06.2018
  3. Руководство по ASP.NET MVC 5 [Електронні ресурси] // Режим доступу: https://metanit.com/sharp/mvc5/. Перевірено: 04.06.2018
  4. ASP.NET MVC 4. Разработка реальных веб-приложений с помощью ASP.NET MVC / [[Джесс Чедвик](https://www.ozon.ru/person/19635157/), [Тодд Снайдер](https://www.ozon.ru/person/19635171/), [Хришикеш Панда](https://www.ozon.ru/person/19635191/)] – Вильямс, 2013. – 432 ст.
  5. Рихтер Дж. **«**CLR via C#. Программирование на платформе Microsoft .NET Framework 4.0 на языке C#» // Питер. – 2012. – 928 ст.