МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ

Кафедра програмного забезпечення автоматизованих систем

ЗВІТ

з виробничої практики

Студент Кравченко А. О.

Група ПЗ-154

Керівник практики

від підприємства (посада) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Керівник практики

від інституту (посада) асистент кафедри Метелап В. В.

Оцінка

Дата

Черкаси 2018

**ЗМІСТ**

[**ВСТУП** 3](#_Toc506188324)

[**1.** **ОГЛЯД ПІДПРИЄМСТВА** 4](#_Toc506188325)

[**1.1** **СТРУКТУРА ТА НАПРЯМКИ ДІЯЛЬНОСТІ ПІДПРИЄМСТВА** 4](#_Toc506188326)

[**1.1.1** **Загальна інформація про підприємство** 4](#_Toc506188327)

[**1.1.2** **Опис установи** 4](#_Toc506188328)

[**1.2** **КОМП’ЮТЕРИЗОВАНІ ЗАСОБИ ПІДТРИМКИ ДІЯЛЬНОСТІ ПІДПРИЄМСТВА** 4](#_Toc506188329)

[**1.2.1** **Вимоги до виробничого середовища** 4](#_Toc506188330)

[**1.2.2** **Робоче місце при проходженні практики** 5](#_Toc506188331)

[**1.2.3** **Опис комп'ютерної техніки і різновидів використовуваного програмного забезпечення** 5](#_Toc506188332)

[**1.2.4** **Дослідження необхідності вдосконалення програмного та технічного забезпечення підприємства** 5](#_Toc506188333)

[**2.** **ВИКОНАННЯ ЗАВДАННЯ ПРАКТИКИ** 6](#_Toc506188334)

[**2.1** **Налаштування робочого середовища** 6](#_Toc506188335)

[**2.2** **Зміст завдання** 6](#_Toc506188336)

[**2.3** **Виконання поставленого завдання** 7](#_Toc506188337)

[**2.3.1** **Вкладка Assignments** 8](#_Toc506188338)

[**2.3.2** **Вкладка Devices** 9](#_Toc506188339)

[**2.3.3** **Вкладка Servers** 11](#_Toc506188340)

[**2.3.4** **Додатковий функціонал ConfigurationEditor** 16](#_Toc506188341)

[**2.3.5** **Додатковий функціонал UserInfo** 17](#_Toc506188342)

[**3.** **Індивідуальне завдання від університету** 18](#_Toc506188343)

[**3.1** **Система контролю версій** 18](#_Toc506188344)

[**3.2** **Налаштування власного GIT-сервера** 18](#_Toc506188345)

[**ВИСНОВОК** 21](#_Toc506188346)

# **ВСТУП**

Метою виробничої практики є застосування студентом набутих навичок на підприємстві та здобуття нових за спеціальністю “розробка програмного забезпечення”.

Керівник підприємства, після ознайомлення студента зі структурою закладу та процесом роботи, дає індивідуальне завдання, яке студент має виконати в межах даної практики.

Основним завданням розробника є проектування, реалізація та тестування програмних продуктів.

Практика проходила у складі групи студентів, які працювали у складі команди. Керівником практики було поставлене наступне завдання для команди: спроектувати, реалізувати програмними засобами та протестувати веб-додаток.

1. **ОГЛЯД ПІДПРИЄМСТВА**
   1. **СТРУКТУРА ТА НАПРЯМКИ ДІЯЛЬНОСТІ ПІДПРИЄМСТВА**

### **Загальна інформація про підприємство**

“ФОП Васільцова” – підприємство, яке займається розробкою програмного забезпечення. Діяльність даного підприємства полягає у розміщенні даних на веб-вузлах.

Підприємство займається розробкою програмних продуктів на Java, .NET, SalesForce.

### **Опис установи**

Підприємство “ФОП Васільцова” було створене у 2015 році. В будівлі знаходяться офіси працівників на першому, другому та третьому поверхах, конференц-зал, їдальня.

Підприємство співпрацює з IT компаніями на контрактній основі.

* 1. **КОМП’ЮТЕРИЗОВАНІ ЗАСОБИ ПІДТРИМКИ ДІЯЛЬНОСТІ ПІДПРИЄМСТВА**

### **Вимоги до виробничого середовища**

Вимоги до параметрів виробничого середовища включають вимоги до параметрів мікроклімату, освітлення, шуму й вібрації, рівнів електромагнітного та іонізуючого випромінювання.

У виробничих приміщеннях на робочих місця із ПК мають забезпечуватися оптимальні значення параметрів мікроклімату: температури, відносної вологості й рухливості повітря (ГОСТ 12.1.005–88, СН 4088–86).

Під час проходження практики всі вимоги до виробничого середовища було виконано. Температура повітря, вологість, освітленість повністю відповідали встановленим вимогам.

### **Робоче місце при проходженні практики**

Робоче місце складалося з комп’ютерного столу, комп’ютера HP PRODESK 400 G3 MT, монітору, мишки, необхідних кабелів. Було здійнено також підключення до інтернету.

### **Опис комп'ютерної техніки і різновидів використовуваного програмного забезпечення**

На практиці використовувалась така комп’ютерна техніка: комп’ютер із характеристиками, представленими в табл. 1:

На робочих станціях встановлена операційна система Microsoft Windows 10, яка має ліцензію. На робочому місці вже було встановлено Microsoft Office 2016 – дозволяє працювати у текстовому редакторі, електронних таблицях, створювати презентації і бази даних.

Таблиця 1 – Характеристики комп’ютера

|  |  |
| --- | --- |
| Процесор | Intel Core i7 |
| Материнська плата | ASUS Z170-A |
| Пам'ять | 16GB |
| Відео | Intel HD Graphics 4600 |
| Жорсткий диск | 1 TB 5400RPM HDD (Hard Disc Drive) |
| Привод оптичних дисків | ASUS BD-R DVD+-R/RW SLIM |

### **Дослідження необхідності вдосконалення програмного та технічного забезпечення підприємства**

Дана установа повністю забезпечена потрібним ПЗ, які відповідають вимогам її діяльності. Присутні ліцензії на всі програмні продукти.

1. **ВИКОНАННЯ ЗАВДАННЯ ПРАКТИКИ**
   1. **Налаштування робочого середовища**

Одним із завдань практики було сконфігурувати на робочій станції таке програмне забезпечення:

* + - Intellij IDEA 2017.3.4 (для розробки продукту на мові Apex);
    - Git (система контролю версій);
    - GitHub (веб-сервіс для хостингу проектів на базі GIT);
    - SalesForce (CRM системa для розробки продукту та управління базою даних)
  1. **Зміст завдання**

Керівником практики було поставлено завдання розробити веб додаток для інвентаризації офісного майна. Додаток має містити базу даних офісів та список обладнання і серверів, що закріплені за кожним із офісів.

Доступ до інвентарних даних конкретного офісу має лише директор даного офісу. Інформація по всіх офісах одразу, доступна лише головному директору.

Обладнання можуть бути закріплені за певним працівником або знаходитись в резерві. Користувач має можливість переглянути список всіх працівників офісу та обладнання, що надане їм; список всього обладнання в офісі (як закріпленого за кимось, так і того, що знаходиться в резерві) та їх комплектуючих.

Розробити можливість заміни комплектуючих обладнання (таких як материнська плата, CPU, RAM та HDD); можливість замінити закріплене за працівником обладнання на інше. Заміна обладнання може відбуватися як всередині офісу (пристрій та комплектуючі беруться із резерву всередині офісу) так і між офісами.

## **Виконання поставленого завдання**

Для реалізації завдання була обрана CRM система Salesforce. В ній реалізована власна мова програмування Apex, що базується на об’єктно орієнтованій мові java. Apex застосовується для взаємодії бази даних SOQL та фреймворку VisualForce. VisualForce базується на XHTML та надає власні можливості розробки сторінки, що дозволяють ефективно взаємодіяти із ApexController.

Розробка виконувалася у складі команди із 5 осіб. Декомпозицію проекту на окремі завдання та їхній розподіл між членами команди виконувалося менеджером підприємства. Кожен модуль містив як взаємодію із сервером, так і створення інтерфейсу користувача.

Модуль, що розроблявся мною, відповідав за показ усього обладнання в офісі та можливість зміни його конфігурації всередині офісу. Модуль складався із двох сторінок.

Перша сторінка показувала користувачеві список усього обладнання та комплектуючих, що закріплені за офісом (в якому залогінений користувач є директором). Користувач має змогу переглянути конфігурацію обладнання та, за потреби, змінити його комплектуючі.

Друга сторінка є редактором конфігурації обладнання. Користувач має змогу змінити основні компоненти обладнання: материнська плата, CPU, RAM, HDD.

База даних для проекту проектувалася та розроблялася спільно усією командою під наглядом ментора.

Salsforce дозволяє легко розробляти продукт в команді, але не передбачає контроль версій створюваного продукту, тому для зручності розробки команда використовувала веб-сервіс для хостингу проектів на базі GIT – GitHub.

Для надання таблицям зручного вигляду та реалізації деяких функцій пошуку, застосовувався DataTable плагін.

У результати виконання завдання було отримано програмний компонент, який забезпечує функції інвентеризації та моніторингу обладнання в офісах фірми.

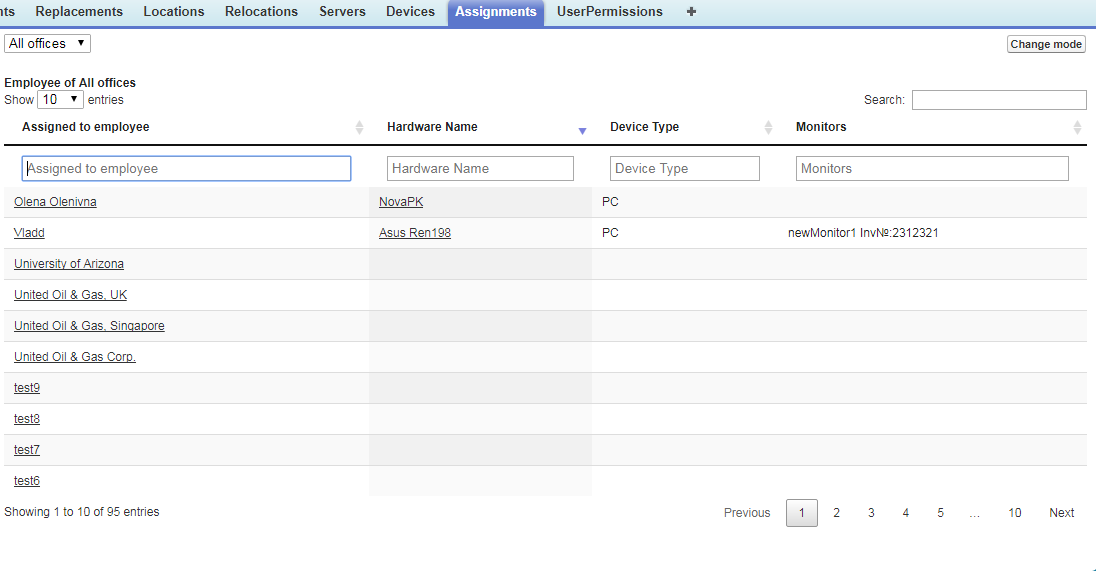
### **Вкладка Assignments**

Вкладка Assignments являє собою головну сторінку системи. Головна ціль вкладки - відобразити всіх співробітників того чи іншого офісу. На вкладці Assignments реалізована логіка прав доступу. Тобто користувачі, що закріплені за конкретними офісами можуть взаємодіяти тільки з співробітниками своїх офісів. Виключенням являється головний адміністратор, який має права доступу до всіх співробітників всіх офісів а також деяких функцій, а саме:

* Випадаючий список для фільтрації співробітників по всім можливим офісам.

В доповнення до повного імені співробітника надається інформація про місцезнаходження його робочого офісу, закріплене за ним апаратне забезпечення, закріпленні за ним монітори, посада в компанії. Є можливість сортування та фільтрації по кожній колонці. Також з функціоналу можна виділити - режим повного так короткого показу інформації про співробітників.

З вкладки Assignments здійснюються всі подальші переходи на вкладки - Devices, Servers, UserInfo.

Рисунок 1 – Вкладка Assignments. Співробітники всіх офісів. Коротка версія таблиці.

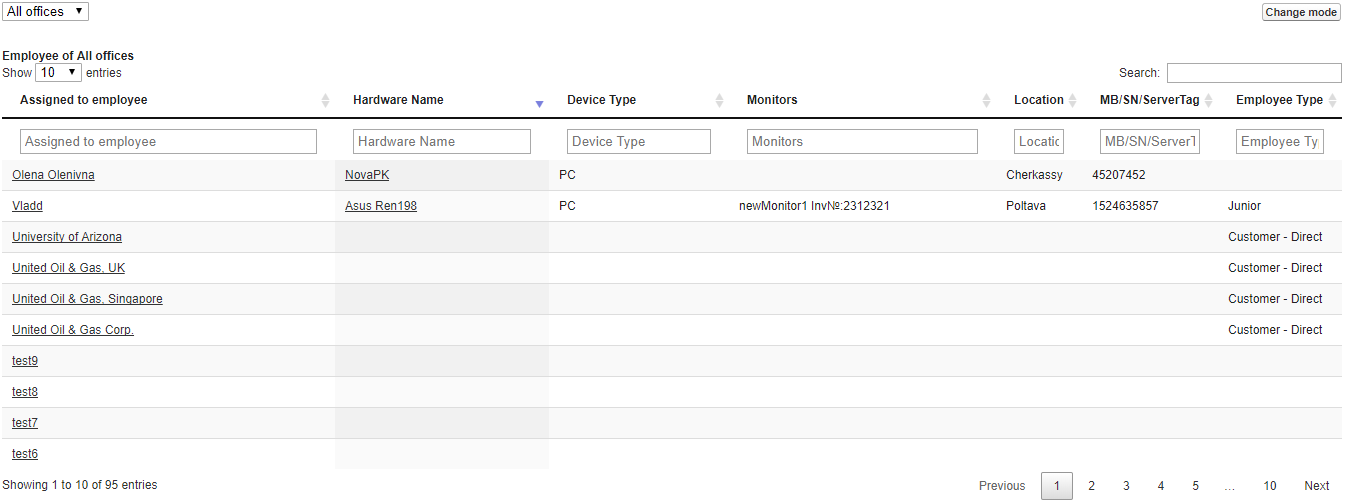
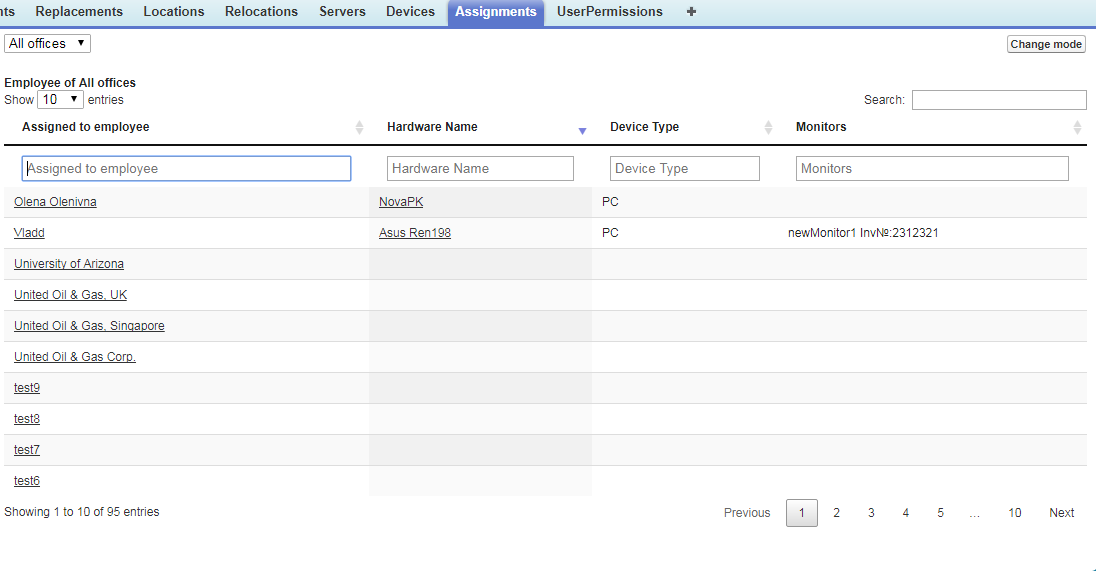


Рисунок 2 – Вкладка Assignments. Співробітники всіх офісів. Повна версія таблиці.

### **Вкладка Devices**

Вкладка Devices дозволяє користувачеві переглянути все обладнання та компоненти, що закріплені за офісом. Так-як мій тестовий аккаунт був закріплиней лише за певним офісом (не головний директор), то інформація в таблицях показана лише для конкретного офісу.

Сторінка складається із двох блоків:

Assemblies – таблиця що показує назву обладнання, користувача що відповідає за нього, код обладнання та його тип. Ця таблиця показує все обладнання та компоненти, що є в резерві офісу. Для списку реалізовано сортування по колонках, фільтрування даних за кожною колонкою та серверну пагінацію.

Configurations – показує компоненти, що входять в обране обладнання. Дана таблиця містить назву компонента, його код та тип. При необхідності конфігурацію обладнання можна змінити, натиснувши на кнопку “EDIT CONFIGURATION”.

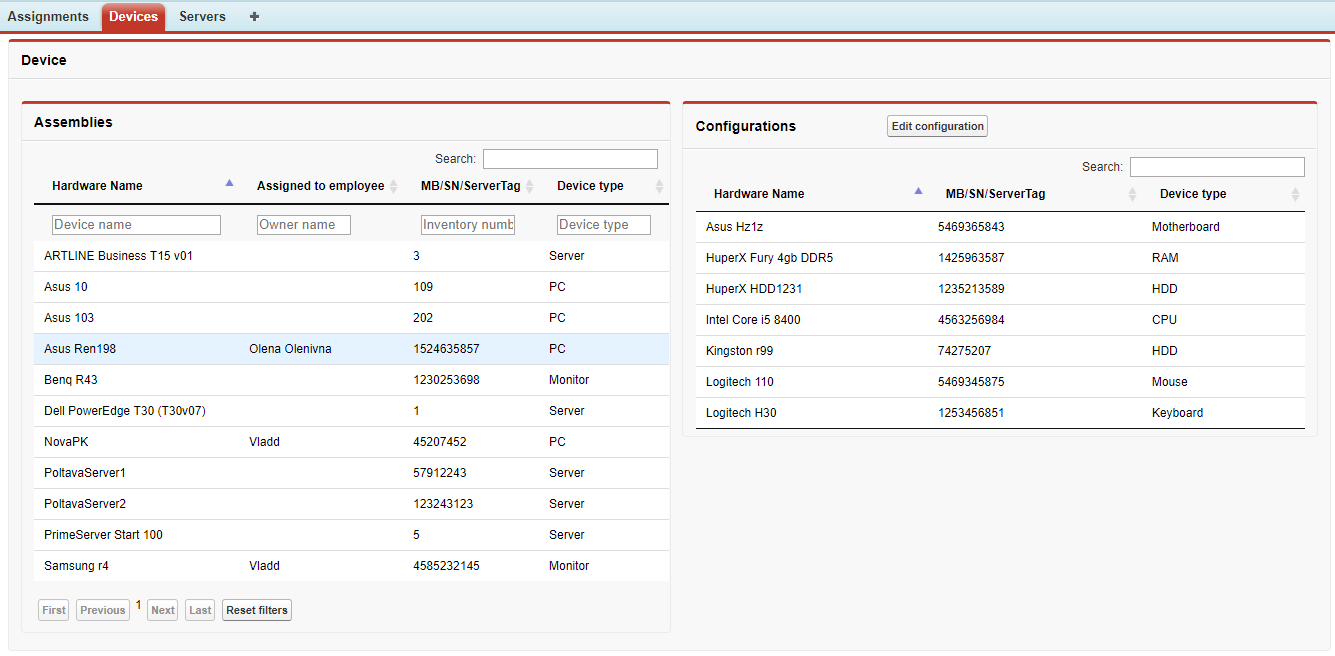


Рисунок 1 – список усіх девайсів офісу без фільтрів. Конфігурація обраного обладнання.

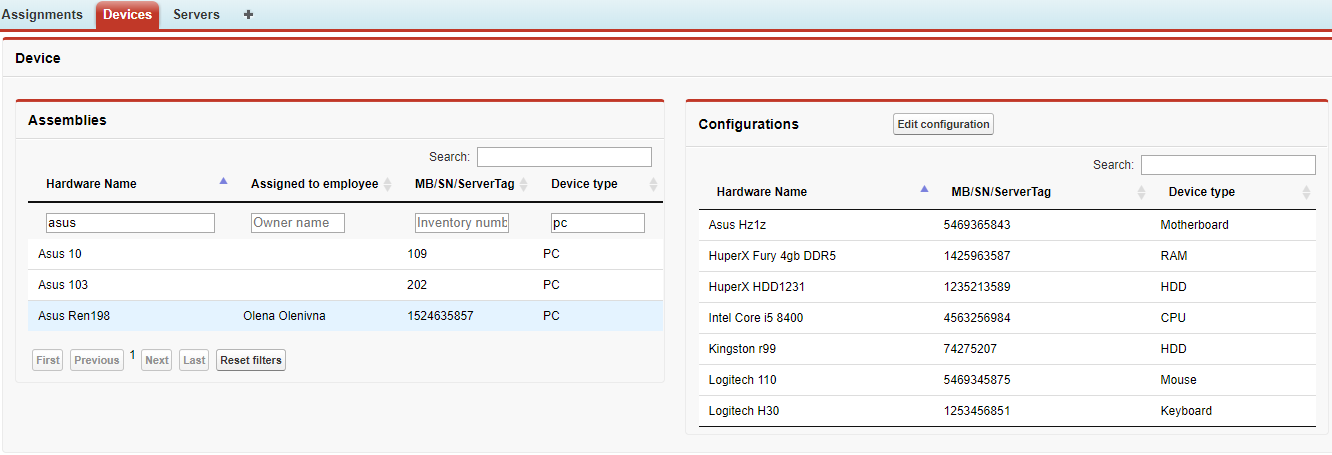


Рисунок 2 – список усіх девайсів офісу із врахуванням фільтрів по назві та типу девайсу. Конфігурація обраного обладнання.

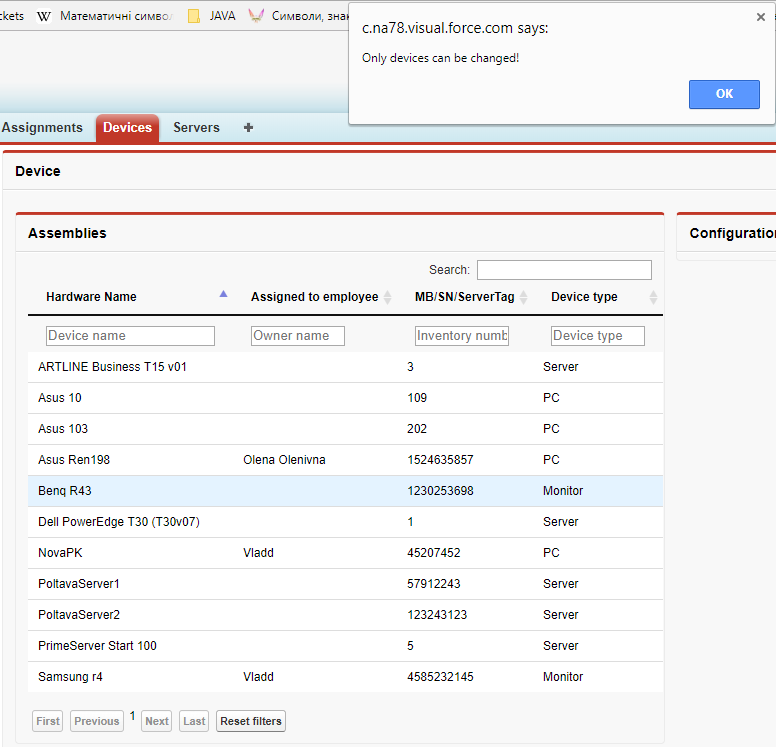


Рисунок 3 – лише обладнання мають конфігурацію, тому компоненти не можуть бути змінені.

### **Вкладка Servers**

Вкладка Servers дозволяє переглянути список діючих серверів офісів компанії відповідно до статуса користувача. У табличці є такі поля:

* назва серверу;
* IP адреса серверу;
* обладнання, що виділене під сервер;
* опис того, що знаходиться на сервері.

Якщо сторінку переглядає локальний адміністратор офісу, то йому доступний список лише тих серверів, що знаходяться у його місті.

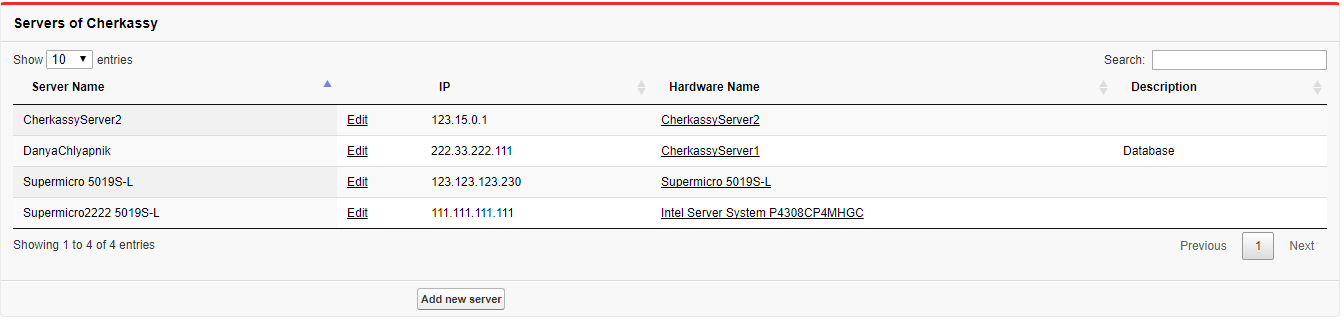


Рисунок 1 – вигляд сторінки Servers локального адміністратора міста Черкас.

Якщо користувач є глобальним адміністратором, то йому відкривається доступ до перегляду доступних серверів усіх офісів компанії.

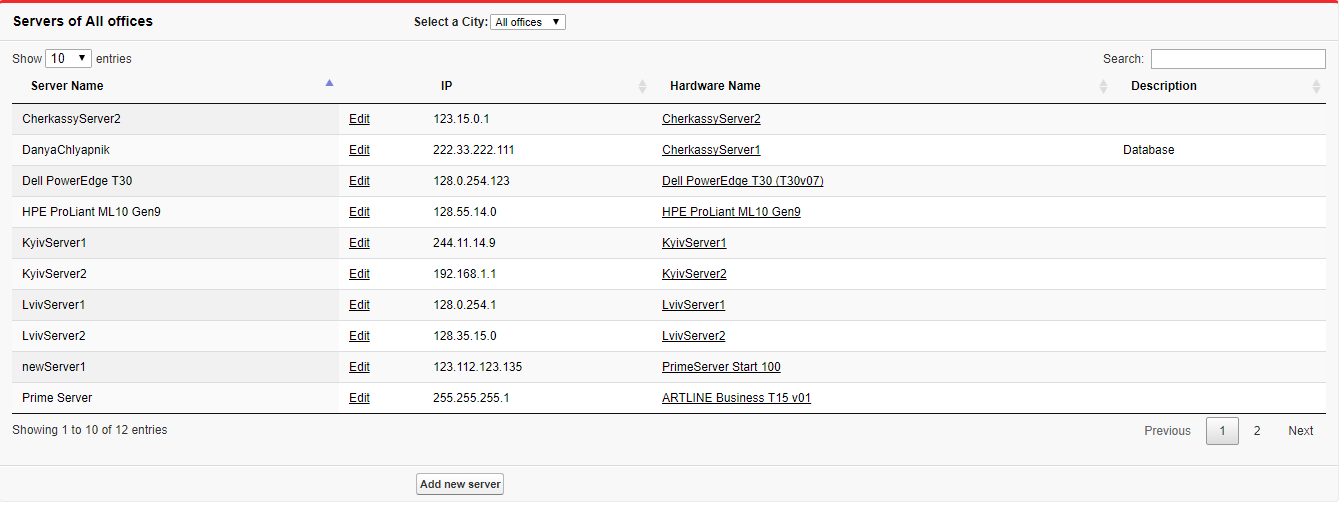


Рисунок 2 – вигляд сторінки Servers глобального адміністратора.

Також глобальний адміністратор може переглядати список серверів конкретного офісу за його бажанням. Для цього у випадаючому списку він моєет вибрати необхідне місто.

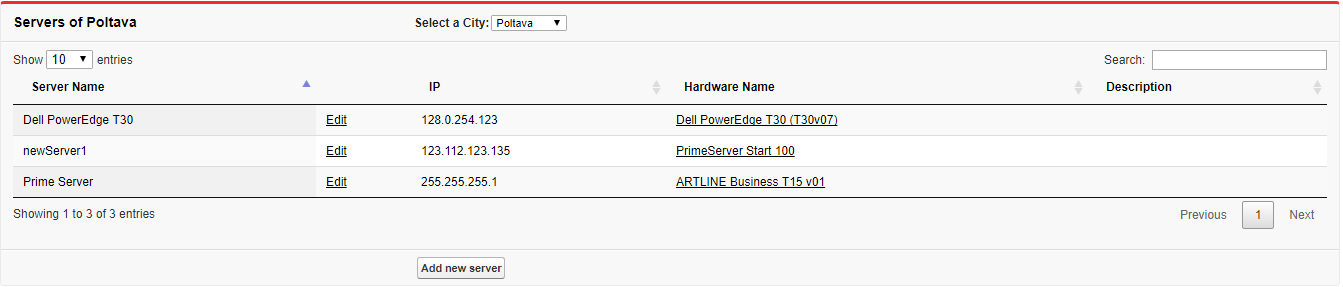


Рисунок 3 – вигляд вкладки Servers глобального адміністратора для перегляду серверів міста Полтава.

Був передбачений функціонал редагування даних про сервера, для цього навпроти кожної назви серверу є посилання «Edit», при натисканні на яке користувач потрапляє на нову сторінку з необхідним для редагування функціоналом.

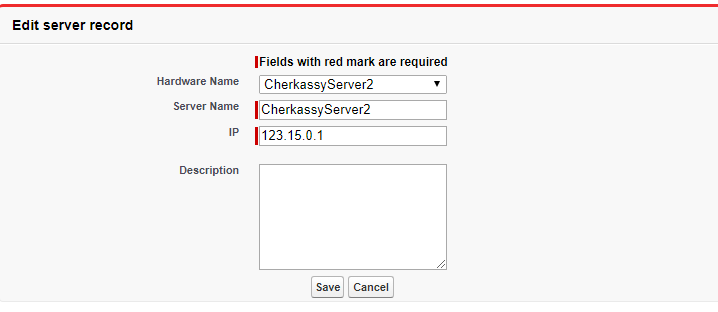


Рисунок 4 – вигляд сторінки Edit server record.

Тут користувач може змінити будь-яке поле обраного серверу. На поле IP додані клієнтська та серверна валідації, якщо користувач спробує редагувати IP адресу на невалідну, то він побачить помилку.

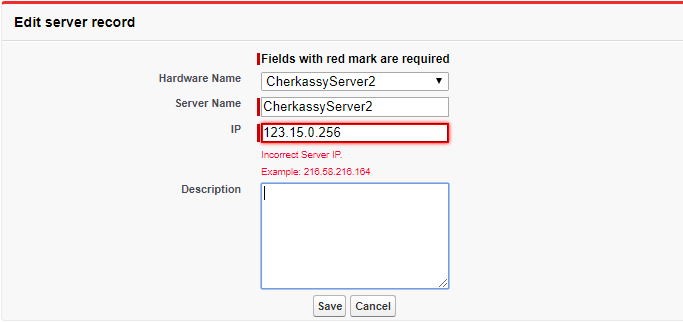


Рисунок 5 – клієнтська валідація поля IP.

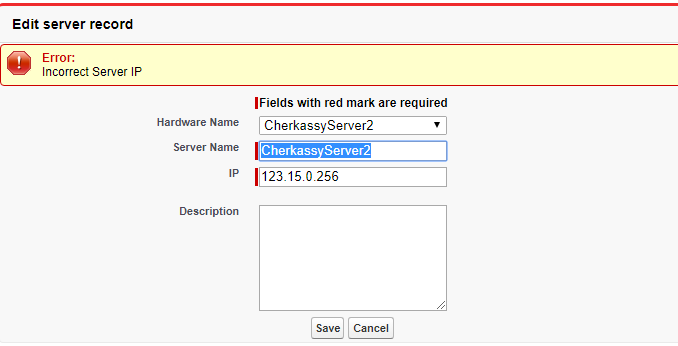


Рисунок 6 – серверна валідація поля IP.

Аналогічно до функції редагування існуючого серверу була реалізована функція додавання серверу, нею можна скористатися натиснувши на кнопку «Add new server». Застосовані валідації поля IP адреси діють і на цій сторінці.

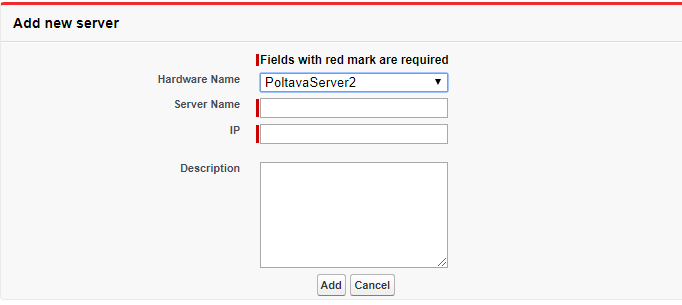


Рисунок 7 – вигляд сторінки Add new server.

Якщо у даний момент немає вільного виділеного під сервер обладнаня, то користувач у списку серверів побачить повідомлення: <No free servers available> та не зможе додати новий сервер.

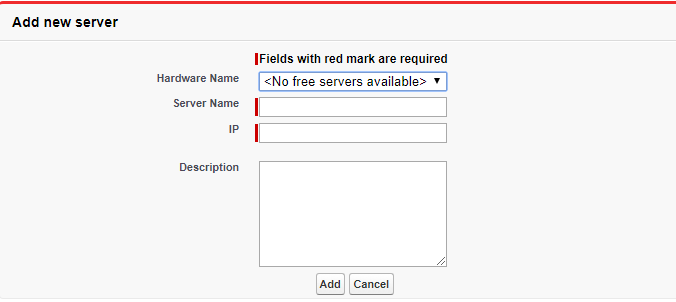


Рисунок 8 – вигляд сторінки Add new server у випадку відсутності вільного виділеного під сервер обладнання.

### **Додатковий функціонал ConfigurationEditor**

Конфігурація обладнання може дуже часто змінюватися. Для зручності моніторингу цього процесу створена дана сторінка. Користувач обравши потрібне обладнання бачить основні компоненти, які можна змінити. При натисненні кнопки “EDIT” під колонкою із потрібним компонентом, відкривається табличка всіх доступних компонентів, що є в резерві даного офісу. Обравши потрібний елемент із списку, користувач натискає «CONFIRM», чим підтверджує свій вибір. Компоненти не заміняються одразу, для цього потрібно натиснути кнопку “SAVE” після проведення всіх необхідних замін. Якщо компонент було додано випадково, можна натиснути на кнопку “”, щоб відмінити вибір. Кнопка “” в рядку початкової конфігурації видаляє елемент. Цю дію можна відмінити, натиснувши на кнопку “”. Елемент буде відновлено в початковій конфігурації.

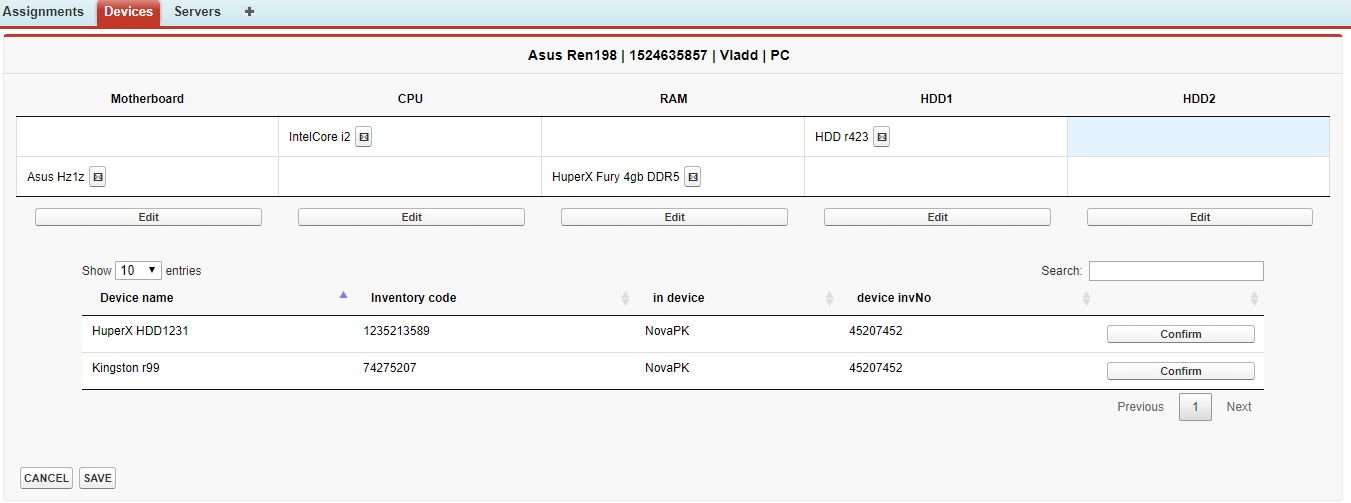


Рисунок 1 – вибір потрібних елементів із резерву.

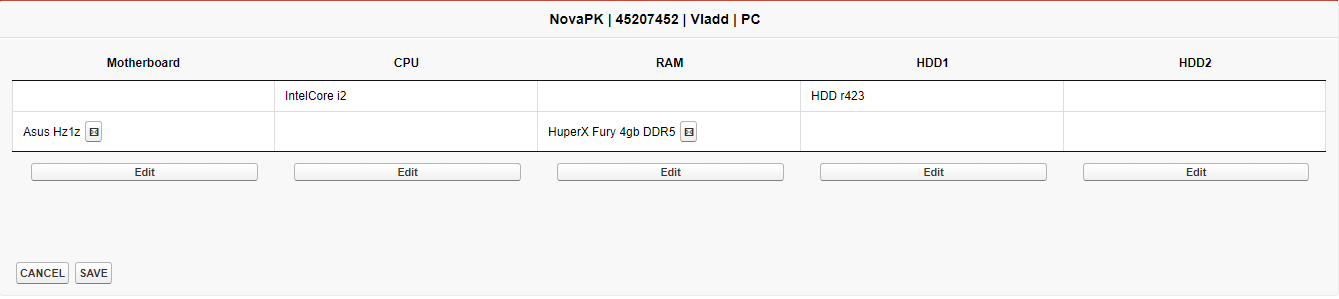


Рисунок 2 – при натиснені на “” обраний компонент видаляється із нової конфігурації.

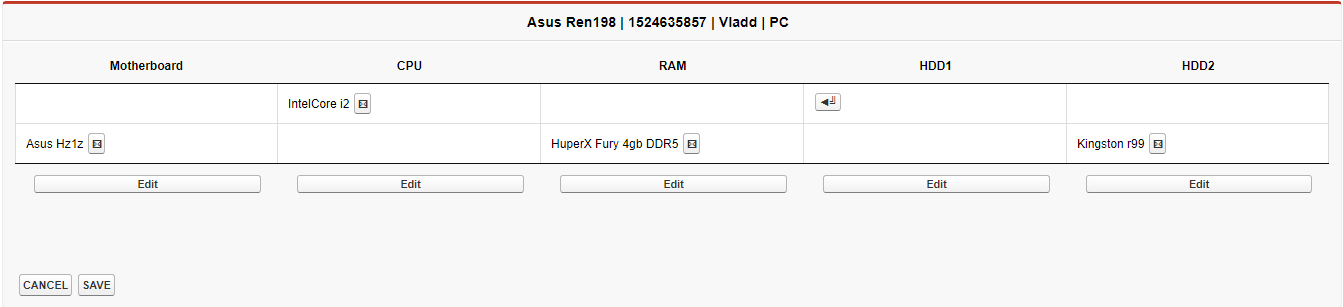


Рисунок 3 - при натиснені на “” обраний компонент видаляється із початкової конфігурації. Кнопка “” відміняє видалення.

### **Додатковий функціонал UserInfo**

Вкладка UserInfo дозволяє переглянути список комп’ютерного обладнання для певного юзера. Основні функції вкладки:

* Перегляд інформації про юзера (ім’я, посада, мережеве ім’я);
* Інформацію про поточний комп’ютер(тип, ім’я девайсу), а також можливість додати новий/редагувати/видалити;
* Інформацію про поточні монітори(ім’я девайсу, його інвентарний номер), а також можливість додати новий/редагувати/видалити;
* Більш детальний список інформації про поточний ПК юзера (OS, MB/SN/Service Tag, CPU, HDD).
* Також є можливість перейти на вкладку “Edit Configuration”, яка дозволяє змінити комплектуючі ПК.

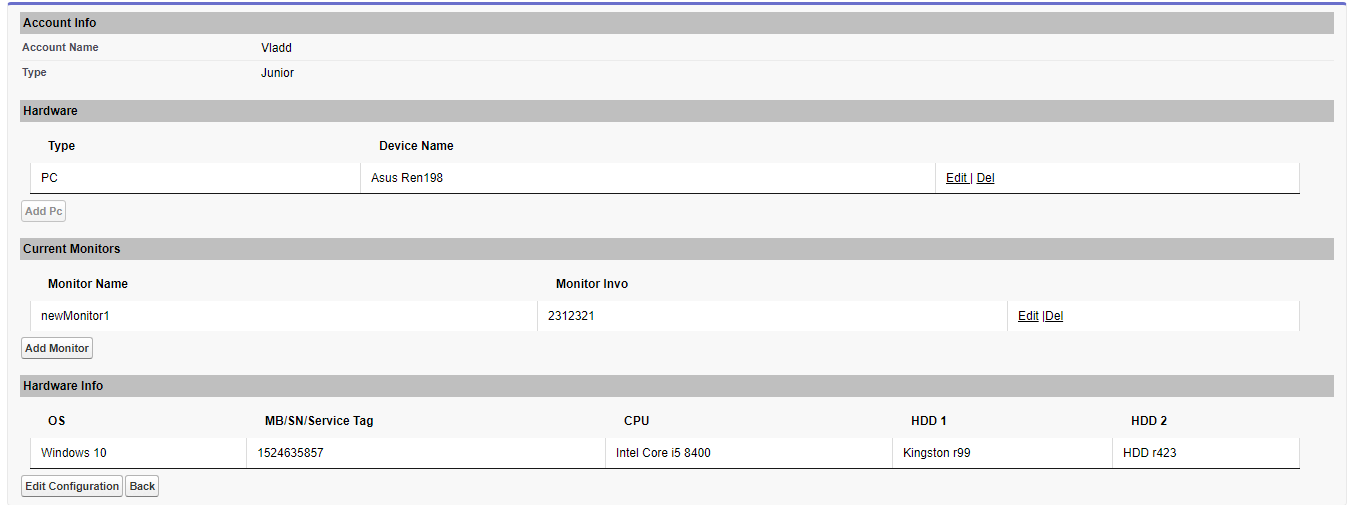


Рисунок 1 – Вигляд сторінки ‘UserInfo’.

# **Індивідуальне завдання від університету**

## **Система контролю версій**

Функція Gіt, яка насправді робить її відмінною від практично будь-якої іншої SCM, є його розгалужена модель.

Git дозволяє і рекомендує вам мати кілька локальних гілок, які можуть бути цілком незалежними одна від одної. GIT дозволяє легко маніпулювати гілками: створення, об'єднання та видалення.

Це означає, що ви можете робити такі речі:

* Бути впевненим, що основний функціонал проекту ніколи не постраждає. Можна створити одну основну гілку на якій тримати лише вже готові та протестовані модулі проекту. Це дозволить розробляти інші модулі не переймаючись за втрату вже проробленої роботи.
* Нові гілки для кожної нової функції, з якою ви працюєте в даний момент. Це дозволить легко переключатися між ними, а потім видалити кожну гілку, коли ця функція об'єднується у основний проект.
* GIT дозволяє випробовувати багато різних ідей одночасно. Можна створити додаткову гілку в якій поексперементувати над ідеями, а потім видалити її, якщо ідея не вдалася.

GitHub – чудовий сервіс на базі GIT, що дозволяє співпрацювати із багатьма учасниками команди одночасно. Однак він накладає певні обмеження. Особливо це помітно для невеликих студій, що не можуть дозволити собі платну підписку. А лише платна версія дозволяє користуватися приватними репозиторіями. До тогож чим більше таких репозиторіїв, тим більшою є ціна в місяць. Але публікувати код проекту у загальний доступ при розробці продукту також неприйнятна річ. В таких випадках багато хто із розробників

В таких випадках багато хто із розробників налаштовує власний GIT сервер. Це дозволяє не тільки заощадити кошти, але й більше контролювати свій сервер. Оскільки ніхто крім самого розробника не має до нього доступу. У більшості випадків більшість передових користувачів Linux вже мають власні сервери, оскільки налаштування GIT серверу не є складним завданням.

## **Налаштування власного GIT-сервера**

Оскільки GIT – це система, що в першу чергу, призвана полегшити контроль версій власного проекту. А це, в більшості випадків, передбачає одночасну роботу кількох осіб над одним проектом. Тому загальноприйнятою практикою є встановлення GIT на віддалений (remote) комп’ютер. Після цього відбувається налаштування локальних комп’ютерів (тобто особистих робочих станцій кожного із розробників).

GIT має бути встановлений на усіх комп'ютерах, які будуть взаємодіяти в системі контролю версій. GIT можна встановити із пакетів, оскільки він вже є доступних із репозиторії дистрибутиву. Для цього потрібно виконати наступну команду:

sudo apt-get install git-core

Після встановлення системи контролю версій, потрібно вказати користувачів:

sudo useradd git  
passwd git

Для більшої зручності доступа до сервера, робочі станції розробників потрібно налаштувати на підключення до віддаленого сервера без пароля. Для цього потрібно створити SSH-ключ на робочій станції та передати його серверу.

ssh-keygen-t rsa

У ньому буде запропоновано вказати місце зберігання ключа, просто натисніть Enter, щоб використати розташування за замовчуванням. Друге питання полягає в тому, щоб вказати пароль, який буде необхідний для доступу до віддаленого сервера. Команда генерує два ключі - відкритий ключ і приватний ключ. Потрібно звернути увагу на розташування відкритого(публічного) ключа, оскільки він знадобиться при введені наступної команди.

Тепер потрібно скопіювати ці ключі на сервер, щоб дві машини могли спілкуватися один з одним. Для цього на робочій станції розробника потрібно виконати наступну команду:

cat ~/.ssh/id\_rsa.pub | ssh git@server "mkdir -p ~/.ssh && cat >> ~/.ssh/authorized\_keys"

Тепер ssh на сервер і створити проектний каталог для Git. Ви можете використовувати потрібний шлях для репо.

Тепер із допомогою SSH потрібно підключитися до віддаленого сервера і створити проектний каталог для GIT. Його можна створити у будь-якій зручній директорії:

git@server:~ $ mkdir -p /home/swapnil/project-1.git

Після того, як каталог був створений, потрібно перейти у нього і вже звідти працювати із системою контролю версій для даного проекту.

cd /home/swapnil/git/project

Тепер потрібно створити порожній репозиторій:

git init --bare

Ініціалізовано порожній репозиторій Git у /home/swapnil/project-1.git 

Тепер нам потрібно створити Git репозиторій на робочій станції програміста:

mkdir -p /home/swapnil/git/project

Щоб було зручніше працювати, перейдемо в створену директорію.

cd /home/swapnil/git/project

Тепер створіть файли, що потрібно буде додати у систему контролю версій. Після цього ініціалізуйте GIT:

git init  
Initialized empty Git repository in /home/swapnil/git/project

Тепер додавайте файли до репозиторія:

git add

Коли користувач вносить якісь зміни у файл чи додає нові файли, він повинен виконувати команду вище. Також до кожного коміта потрібно писати повідомлення. Це загальноприйнята практика і робиться щоб дати іншим розробникам зрозуміти, які саме зміни відбулися у гілці.

git commit -m "message" -a  
 [master (root-commit) 57331ee] message  
 2 files changed, 2 insertions(+)  
 create mode 100644 GoT.txt  
 create mode 100644 writing.txt

У цьому випадку у мене був файл GoT, і я зробив деякі зміни, тому, коли я запустив команду, вона зазначила, що в файл були внесені зміни. У наведеній вище команді опція "-a" означає, що цей опис використовується для всіх файлів у репозиторі. Якщо ви внесли зміни лише до одного файлу, ви можете вказати назву цього файлу, а не використовувати "-a". 

git commit -m "message" GoT.txt  
 [master e517b10] message  
 1 file changed, 1 insertion(+)

До цих пір ми працюємо на локальному сервері. Але щоб із цими файлами та їх змінами могли працювати інші учасники команди, потрібно занести їх на сервер:

git remote add origin ssh://git@remote-server/repo-<wbr< a="">>path-on-server..git

Інші учасники команди мають забрати зроблені кимось зміни із загального сервера:

git clone git@remote-server:/home/swapnil/project.git

Тепер щоб працювати із GIT сервером достатньо лише закомітити локально змінені файли та перенести цю інформацію на сервер:

git commit -m 'corrections in GoT.txt story' -a   
git push origin master

# **ВИСНОВОК**

Практика дозволила застосувати набуті в універсистеті знання та навички в умовах підприємства IT-галузі та здобути новий досвід. Під час проходження практики було виконане індивідуальне завдання у складі команди розробників. Отриманий програмний компонент, використовується замовником для моніторингу обладнання в офісах.

Хоч SalesForce і використовує власні фреймворки та мову програмування, але вони базуються на вже існуючих стандартах та концепціях. Тому знання мови програмування java, досвід роботи із MySQL та навички створення макетів сторінок і їх стилізації засобами HTML + CSS дуже знадобилися при вивченні нової технології. Також досвід роботи із системою контролю версій GIT дуже полегшив розробку проекту в команді. В результаті був створений веб-додаток для моніторингу офісного майна, як в межах конкретного офісу, так і загально по всіх офісах.

Практика надала можливість попрацювати у сучасній компанії та отримати досвід співпраці із висококваліфікованими фахівцями, що поділилися досвідом та навчили розробляти додатки для CRM системи SalesForce.