Національний технічний університет України “КПІ”

(назва вищого навчального закладу)

Кафедра автоматизованих систем обробки інформації і управління

Дисципліна Крос-платформене програмування

Напрям "Комп’ютерні науки", спеціальність "інформаційні управляючі системи та технології"

Курс 3 Група ІС-24 Семестр 6\_

**ЗАВДАННЯ**

**на розрахункову роботу студента**

Вереня Олександр Ігорович

(прізвище, ім’я, по батькові)

1. Тема проекту (роботи) Система складання колендарного плану виготовлення товарів

2. Строк здачі студентом закінченого проекту (роботи)

3. Вихідні дані до проекту (роботи) програмний код написаний мовою Java EE OS: Windows XP/7/8, база даних MySQL.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)

1. Технічне завдання. 2.Пояснювальна записка 3. Інструкція програміста. 4. Інструкція системного програміста. 5. Інструкція користувача

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

Відео копії результатів, схема таблиць бази даних, діаграма варіантів використання,   
діаграма компонентів, DFD діаграма.

6. Дата видачі завдання 28. 02.2015 р.

 Національний технічний університет України “КПІ”

(назва вищого навчального закладу)

Кафедра автоматизованих систем обробки інформації і управління

Дисципліна Крос-платформене програмування

Напрям "Комп’ютерні науки", спеціальність "інформаційні управляючі системи та технології"

Курс 3 Група ІС-24 Семестр 6\_

**ЗАВДАННЯ**

**на розрахункову роботу студента**

Кмець Максим Ігорович

(прізвище, ім’я, по батькові)

1. Тема проекту (роботи) Система складання колендарного плану виготовлення товарів

2. Строк здачі студентом закінченого проекту (роботи)

3. Вихідні дані до проекту (роботи) програмний код написаний мовою Java EE OS: Windows XP/7/8, база даних MySQL.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)

1. Технічне завдання. 2.Пояснювальна записка 3. Інструкція програміста. 4. Інструкція системного програміста. 5. Інструкція користувача

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

Відео копії результатів, схема таблиць бази даних, діаграма варіантів використання,   
діаграма компонентів, DFD діаграма.

6. Дата видачі завдання 28. 02.2015 р.

Національний технічний університет України “КПІ”

(назва вищого навчального закладу)

Кафедра автоматизованих систем обробки інформації і управління

Дисципліна Крос-платформене програмування

Напрям "Комп’ютерні науки", спеціальність "інформаційні управляючі системи та технології"

Курс 3 Група ІС-24 Семестр 6\_

**ЗАВДАННЯ**

**на розрахункову роботу студента**

Сидоров Максим Олегович

(прізвище, ім’я, по батькові)

1. Тема проекту (роботи) Система складання колендарного плану виготовлення товарів

2. Строк здачі студентом закінченого проекту (роботи)

3. Вихідні дані до проекту (роботи) програмний код написаний мовою Java EE OS: Windows XP/7/8, база даних MySQL.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)

1. Технічне завдання. 2.Пояснювальна записка 3. Інструкція програміста. 4. Інструкція системного програміста. 5. Інструкція користувача

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

Відео копії результатів, схема таблиць бази даних, діаграма варіантів використання,   
діаграма компонентів, DFD діаграма.

6. Дата видачі завдання 28. 02.2015 р.

Національний технічний університет України “КПІ”

(назва вищого навчального закладу)

Кафедра автоматизованих систем обробки інформації і управління

Дисципліна Крос-платформене програмування

Напрям "Комп’ютерні науки", спеціальність "інформаційні управляючі системи та технології"

Курс 3 Група ІС-24 Семестр 6\_

**ЗАВДАННЯ**

**на розрахункову роботу студента**

Тимчук Андрій Олександрович

(прізвище, ім’я, по батькові)

1. Тема проекту (роботи) Система складання колендарного плану виготовлення товарів

2. Строк здачі студентом закінченого проекту (роботи)

3. Вихідні дані до проекту (роботи) програмний код написаний мовою Java EE OS: Windows XP/7/8, база даних MySQL.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)

1. Технічне завдання. 2.Пояснювальна записка 3. Інструкція програміста. 4. Інструкція системного програміста. 5. Інструкція користувача

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

Відео копії результатів, схема таблиць бази даних, діаграма варіантів використання,   
діаграма компонентів, DFD діаграма.

6. Дата видачі завдання 28. 02.2015 р.

# **ЗМІСТ**

[1 ПОСТАНОВКА ЗАДАЧІ 9](#_Toc389538481)

[1.1 Опис предметного середовища 9](#_Toc389538482)

[1.2 Вхідні дані 9](#_Toc389538483)

[1.3 Вихідні дані 11](#_Toc389538484)

[2 РІШЕННЯ З ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ 12](#_Toc389538485)

[2.1 Склад інформаційної бази 12](#_Toc389538486)

[2.2 Організація інформаційної бази 12](#_Toc389538487)

[2.2.1 Принцип побудови інформаційної бази 12](#_Toc389538488)

[2.2.2 Структура інформаційної бази 12](#_Toc389538489)

[3 ІНСТРУКЦІЯ ПРОГРАМІСТА 17](#_Toc389538490)

[3.1 Призначення та умови застосування сервісу 17](#_Toc389538491)

[3.2 Характеристики сервісу 18](#_Toc389538492)

[3.3 Звернення до системи 18](#_Toc389538493)

[3.3.1 Початок роботи 18](#_Toc389538494)

[3.3.2 Робота з сервісом 18](#_Toc389538495)

[3.3.3 Завершення роботи з сервісом 19](#_Toc389538496)

[3.4 Вхідні та вихідні дані 19](#_Toc389538497)

[3.4.1 Вхідні дані 19](#_Toc389538498)

[3.4.2 Вихідні дані 20](#_Toc389538499)

[3.5 Повідомлення 20](#_Toc389538500)

[4 ІНСТРУКЦІЯ СИСТЕМНОГО ПРОГРАМІСТА 23](#_Toc389538501)

[4.1 Загальні відомості про програму 23](#_Toc389538502)

[4.1.1 Призначення програми 23](#_Toc389538503)

[4.1.2 Функції сервісу 23](#_Toc389538504)

[4.1.3 Мінімальний склад технічних засобів 24](#_Toc389538505)

[4.1.4 Мінімальний склад програмних засобів 25](#_Toc389538506)

[4.1.5 Вимоги до персоналу (системного адміністратора) 25](#_Toc389538507)

[4.2 Структура сервісу 25](#_Toc389538508)

[4.2.1 Відомості про структуру сервісу 25](#_Toc389538509)

[4.2.2 Відомості про складові частини програми 25](#_Toc389538510)

[4.2.3 Відомості про зв'язки між складовими частинами програми 26](#_Toc389538511)

[4.2.4 Відомості про зв'язки з іншими програмами 26](#_Toc389538512)

[4.3 Налаштування програми 26](#_Toc389538513)

[4.3.1 Налаштування сервера 26](#_Toc389538514)

[4.3.2 Налаштування на стороні користувача. 26](#_Toc389538515)

[4.4 Перевірка програми 26](#_Toc389538516)

[4.4.1 Постановка задачі тестування 26](#_Toc389538517)

[4.4.2 Створення коду програми 27](#_Toc389538518)

[4.4.3 Тестування 28](#_Toc389538519)

[4.4.5 Повний код програми 34](#_Toc389538520)

[5 ІНСТРУКЦІЯ КОРИСТУВАЧА 37](#_Toc389538521)

[5.1 Вступ 37](#_Toc389538522)

[5.1.1 Область застосування 37](#_Toc389538523)

[5.1.2 Короткий опис можливостей 37](#_Toc389538524)

[5.1.3 Рівень підготовки користувача 37](#_Toc389538525)

[5.1.4 Перелік експлуатаційної документації 37](#_Toc389538526)

[5.2 Призначення і умови застосування 37](#_Toc389538527)

[5.2.1 Види діяльності, функції 37](#_Toc389538528)

[5.2.2 Програмні та апаратні вимоги до системи 38](#_Toc389538529)

[5.3 Опис операцій 38](#_Toc389538530)

[6 ВИСНОВКИ 43](#_Toc389538531)

[6.1 Результати роботи 43](#_Toc389538532)

[6.2 Виконана функціональність відповідно до ТЗ 43](#_Toc389538533)

[6.3 Невиконана функціональність відповідно до ТЗ 43](#_Toc389538534)

[6.4 Перспективи розвитку програми 44](#_Toc389538535)

[7 ПЕРЕЛІК ПОСИЛАНЬ 45](#_Toc389538536)

[ДОДАТОК А.1 46](#_Toc389538537)

[Use Case діаграма 46](#_Toc389538538)

[ДОДАТОК А.2 47](#_Toc389538539)

[Концептуальної моделі БД 47](#_Toc389538540)

[ДОДАТОК А.3 48](#_Toc389538541)

[Діаграма DFD 48](#_Toc389538542)

[ДОДАТОК А.4 49](#_Toc389538543)

[Діаграма компонентів 49](#_Toc389538544)

# 1 ПОСТАНОВКА ЗАДАЧІ

## **1.1 Опис предметного середовища**

Система «СКПВТ» призначена як для великих організацій, так і для звичайних користувачів, які хочуть зібрати собі комп’ютер, який відповідає усім їхнім потребам та можливостям. Цей ресурс дозволяє зареєстрованим користувачам створювати нові замовлення на збір копютерів (одного чи багатьох) та задавати строки на виконання цих замовлень. Також ця система дозволяє отримати своєчасну та швидку інформацію стосовно статусу нового замовлення, можливість чи неможливість його виконання, а також дату закінчення виконання замовлення. Кожен користувач, який створив нове замовлення може відмінити його, якщо воно ще не перейшло в стан виконуваного. У разі відмови у виконанні або успішному закінченню замовлення, користувач отримує своєчасне повідомлення на електронну пошту, зазначену при реєстрації.

Основними цілями створення системи «СКПВТ» є:

* Удосконалення роботи підприємства з виготовлення та збору комп’ютерів;
* Створення простого та інтуїтивно зрозумілого сервісу оформлення замовлень для клієнта заводу зі складання комп’ютерів за допомогою:
* Надання графічного конструктора складання типу комп’ютера;
* Надання критеріїв оцінки готового комп’ютера(вартість, потужність, якість);
* Підвищення продуктивності роботи планувальників заводу зі складання комп’ютерів за допомогою:
* Надання зручного інтерфейсу зворотнього зв’язку з замовником стосовно розглянутих замовлень;
* Надання автоматизованої системи розподілу працівників за прийнятими замовленнями;
* Подача розподілу замовлень у вигляді календарного плану;
* Створення інтерфейсу керування ресурсами(робітники, деталі) заводу зі складання комп’ютерів

## **1.2 Вхідні дані**

Первісні дані вводяться в систему користувачем, через веб-інтерфейс системи.

**Дані, які надходять від користувача**. Першим чином користувач при реєстрації створює ***логін*** та ***пароль***. Далі, якщо користувач захоче створити нове замовлення, то він передає інформацію про бажане наповнення комп’ютера, якість, потужність, ціну та кількість таких комп’ютерів. Якщо користувач хоче відмінити замовлення, він надає інформацію про відмову від замовлення.

**Дані, які надходять від планувальника**. Планувальник відповідає за модерацію існуючих замовлень. Тому дані які надаватимуться планувальником такі:

* Відправлені на планування та репланування замовлення;
* Замовлення, які планувальник відмінив;
* Замовлення, які пропонуються змінити користувачем;
* та ін.

**Дані, які надходять від менеджера ресурсів**. Менеджер ресурсів відповідає за регулювання таких ресурсів як робітники та деталі, тому дані, які надає менеджер ресурсів наступні:

* інформація про деталь, яку необхідно додати до ситеми;
* інформація про деталь, яку необхідно видалити з ситеми;
* інформацію про нового робітника системи;
* інформацію про робітника, якого звільняють;

## **1.3 Вихідні дані**

Вся вихідна інформація надається користувачеві через вікно браузера. Це може бути інформація про поточне замовлення, його статус. Також сплановані замовлення відображаються у календарі планувальника. У відділі ресурсів можна отримати інформацію про існуючих найнятих робітників та деталі, які може виготовляти система.

# **2 РІШЕННЯ З ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ**

## 2.1 Склад інформаційної бази

До складу інформаційної бази входять: дані про користувачів, робітників, деталі, замалення, контракти з робітниками.

## 2.2 Організація інформаційної бази

### 2.2.1 Принцип побудови інформаційної бази

БД побудована таким чином, шо кожному об’єкту системи відповідає таблиця. Якщо користувач створює замовлення, то воно зберігається у відповідній таблиці. Інформація про склад комп’ютерів замовлення, робітників, та котракти на виконання замовлення з робітниками зберігаються у відповідних таблицях.

Детальну інформацію про БД дивитися у слідуючому пункті.

### 2.2.2 Структура інформаційної бази

Таблиця 2.1 – Опис таблиць БД

| **Назва таблиці** | **Поля таблиці** | **Тип даних поля** | **Логічний зміст поля** |
| --- | --- | --- | --- |
| user | Id | Integer | Id – який відповідає користувачу |
| first\_name | Characters(45) | Ім’я користувача |
| last\_name | Characters(45) | Прізвище користувача |
| enabled | tinyint(1) | Чи був користувач варифікован |
| password | Characters(60) | Пароль користувача |
| email | Characters(45) | Email користувача |
| role | id | Integer | Id ролі |
| role | varchar(45) | Назва ролі |
| order\_data | Id | INT(11) | Id замовлення |
| deadline | DATETIME | Дата дедлайну по замовленню |
| price | INT(11) | Загальна ціна замовлення |
| count\_computers | INT(11) | Кількість комп’ютерів у замовленні |
| performance\_time | INT(11) | Загальний час на виконання замовлення |
| propose | VARCHAR(500) | Якщо поле null, то поланування замовлення успішне, інакше містить пропозицію про зміну замовлення. |
| status | Id | INT | Id статусу замовлення |
| name | VARCHAR(45) | Назва статусу |
| computer | Id | Integer | Id комп’ютера |
| price | FLOAT | Ціна одного комп’ютера заданої конфігурації |
| power | FLOAT | Бажана потужність комп’ютера |
| quality | FLOAT | Бажана якість комп’ютера |
| detail | Id | Integer | Id – який відповідає статті |
| price | INT | Ціна деталі |
| quality | VARCHAR(45) | Якість деталі |
| power | Integer | Потужність деталі |
| name | VARCHAR(45) | Назва деталі |
| detail\_type | Id | Integer | Id типу деталі |
| name | VARCHAR(45) | Назва типу деталі |
| produce\_time | INT | Час виготовлення деталі |
| specialization | Id | INT | Id спеціалізації |
| detail\_type\_id | INT | id типу деталі |
| worker\_id | INT | id робітника |
| worker | Id | INT | Id робітника |
| name | VARCHAR(45) | Ім’я робітника |
| sename | VARCHAR(45) | Прізвище робітника |
| cash | INT | Заробітна плата робітника |
| contract | Id | INT | Id кнтракту |
| start\_date | DATETIME | Початок виконання завдання |
| end\_date | DATETIME | Кінець виконання завдання |
| order\_id | INT | id замовлення |
| worker\_id | INT | id робітника |

\_\_\_\_\_\_\_\_\_\_\_\_Інструкція програміста\_\_\_\_\_\_\_\_\_\_\_\_

(найменування програми (документа)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(вид носія даних)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(обсяг програми (документа), арк., Кб)

*студента групи ІС-24 ІІІ курсу*

*Вереня О.І.*

*студента групи ІС-24 ІІІ курсу*

*Кмець М.І.*

*студент групи ІС-24 ІІІ курсу*

*Сидоров М.О.*

*студент групи ІС-24 ІІІ курсу*

*Тимчук А.О.*

2015

ЗАТВЕРДЖУЮ:

Керівник курсового проекту

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_*ас. Сперкач М.О.\_\_\_\_*

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 20\_\_р.

АРКУШ ЗАТВЕРДЖЕННЯ

\_\_\_\_\_\_\_\_\_\_\_\_Інструкція програміста\_\_\_\_\_\_\_\_\_\_\_\_

(найменування програми (документа)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(вид носія даних)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(обсяг програми (документа), арк., Кб)

*студента групи ІС-24 ІІІ курсу*

*Вереня О.І.*

*студента групи ІС-24 ІІІ курсу*

*Кмець М.І.*

*студент групи ІС-24 ІІІ курсу*

*Сидоров М.О.*

*студент групи ІС-24 ІІІ курсу*

*Тимчук А.О.*

2015

# **3 ІНСТРУКЦІЯ ПРОГРАМІСТА**

## 3.1 Призначення та умови застосування сервісу

Основним призначенням системи «СКПВТ» є:

* Удосконалення роботи підприємства з виготовлення та збору комп’ютерів;
* Створення простого та інтуїтивно зрозумілого сервісу оформлення замовлень для клієнта заводу зі складання комп’ютерів за допомогою:
* Надання графічного конструктора складання типу комп’ютера;
* Надання критеріїв оцінки готового комп’ютера(вартість, потужність, якість);
* Підвищення продуктивності роботи планувальників заводу зі складання комп’ютерів за допомогою:
* Надання зручного інтерфейсу зворотнього зв’язку з замовником стосовно розглянутих замовлень;
* Надання автоматизованої системи розподілу працівників за прийнятими замовленнями;
* Подача розподілу замовлень у вигляді календарного плану;
* Створення інтерфейсу керування ресурсами(робітники, деталі) заводу зі складання комп’ютерів

Система «СКПВТ» призначена як для великих організацій, так і для звичайних користувачів, які хочуть зібрати собі комп’ютер, який відповідає усім їхнім потребам та можливостям. Цей ресурс дозволяє зареєстрованим користувачам створювати нові замовлення на збір копютерів (одного чи багатьох) та задавати строки на виконання цих замовлень. Також ця система дозволяє отримати своєчасну та швидку інформацію стосовно статусу нового замовлення, можливість чи неможливість його виконання, а також дату закінчення виконання замовлення. Кожен користувач, який створив нове замовлення може відмінити його, якщо воно ще не перейшло в стан виконуваного. У разі відмови у виконанні або успішному закінченню замовлення, користувач отримує своєчасне повідомлення на електронну пошту, зазначену при реєстрації.

**Вимоги до програмного забезпечення**

Вимоги до програмного забезпечення серверу:

* Java EE 7;
* Tomcat 7 або вище;

Вимоги до програмного забезпечення серверу бази даних:

* СУБД MySQL

Вимоги до програмного забезпечення клієнта:

* Браузер IE 10 або вище, оновлені Firefox, Chrome, Opera, Safari або будь-який інший браузер.

**Вимоги до технічного забезпечення**

Вимоги до технічного забезпечення серверу:

* Процесор – 1 Ггц, 1 ядро ЦП або краще
* Оперативна пам’ять не менш ніж 256 Мб
* Не менше ніж 5 ГБ ПЗУ
* 10 Мбіт/сек доступу до мережі Інтернет

Вимоги до технічного забезпечення серверу бази даних:

* Процесор – 1 Ггц, 1 ядро ЦП або краще
* Оперативна пам’ять не менш ніж 256 Мб
* Не менше ніж 5 ГБ ПЗУ
* 10 Мбіт/сек доступу до мережі Інтернет

## 3.2 Характеристики сервісу

Робота з сервісом можлива в декількох режимах: в режимі зареєстрованого користувача, планувальника, менеджера ресурсів. Режим планувальника дозволяє переглядати замовлення, відправляти на планування нові замовлення, відміняти нові замовлення. Режим менеджера ресурсів дозволяє додавати та видаляти нову інформацію про робітників та деталі.

При збої в роботі апаратури сервера відновлення нормальної роботи системи має здійснюватися після:

* перезавантаження сервера;
* запуску системи.

При помилках в роботі апаратних засобів (крім носіїв даних та програм) відновлення функції системи покладається на ОС.

При помилках, пов’язаних з програмним забезпеченням (ОС та драйвери пристроїв), відновлення працездатності покладається на ОС.

## 3.3 Звернення до системи

### 3.3.1 Початок роботи

Сервіс можна запустити ввівши http://localhost:8080 в адресний рядок браузера.

### 3.3.2 Робота з сервісом

Робота з сервісом відбувається за сценарієм, що задає користувач.

### 3.3.3 Завершення роботи з сервісом

Завершити роботу можна одним зі способів:

* однократним натисканням на кнопку закриття вікна браузера у верхньому правому куті вікна;
* однократним одно часовим натисканням клавіш «Alt» і «F4»;
* закривши вкладку браузера в якій був відкритий сервіс.

## 3.4 Вхідні та вихідні дані

### **3.4.1 Вхідні дані**

Первісні дані вводяться в систему користувачем, через веб-інтерфейс системи.

**Дані, які надходять від користувача**. Першим чином користувач при реєстрації створює ***логін*** та ***пароль***. Далі, якщо користувач захоче створити нове замовлення, то він передає інформацію про бажане наповнення комп’ютера, якість, потужність, ціну та кількість таких комп’ютерів. Якщо користувач хоче відмінити замовлення, він надає інформацію про відмову від замовлення.

**Дані, які надходять від планувальника**. Планувальник відповідає за модерацію існуючих замовлень. Тому дані які надаватимуться планувальником такі:

* Відправлені на планування та репланування замовлення;
* Замовлення, які планувальник відмінив;
* Замовлення, які пропонуються змінити користувачем;
* та ін.

**Дані, які надходять від менеджера ресурсів**. Менеджер ресурсів відповідає за регулювання таких ресурсів як робітники та деталі, тому дані, які надає менеджер ресурсів наступні:

* інформація про деталь, яку необхідно додати до ситеми;
* інформація про деталь, яку необхідно видалити з ситеми;
* інформацію про нового робітника системи;
* інформацію про робітника, якого звільняють;

***3.4.2 Вихідні дані***

Вся вихідна інформація надається користувачеві через вікно браузера. Це може бути інформація про поточне замовлення, його статус. Також сплановані замовлення відображаються у календарі планувальника. У відділі ресурсів можна отримати інформацію про існуючих найнятих робітників та деталі, які може виготовляти система.

## 3.5 Повідомлення

Користувач отримуватиме повідомлення від системи, після того як він подав заявку на верифікацію. Це повідомлення містить ключ для підтвердження реєстрації. Також користувач отримуватиме повідомлення в разі відмови у виконанні замовлення, або якщо замовлення було успішно завершено.

\_\_\_\_\_\_\_\_\_\_\_\_Інструкція системного програміста\_\_\_\_\_\_\_\_\_\_\_\_

(найменування програми (документа)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(вид носія даних)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(обсяг програми (документа), арк., Кб)

*студента групи ІС-24 ІІІ курсу*

*Вереня О.І.*

*студента групи ІС-24 ІІІ курсу*

*Кмець М.І.*

*студент групи ІС-24 ІІІ курсу*

*Сидоров М.О.*

*студент групи ІС-24 ІІІ курсу*

*Тимчук А.О.*

2015

ЗАТВЕРДЖУЮ:

Керівник курсового проекту

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_*ас. Сперкач М.О.\_\_\_\_*

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 20\_\_р.

АРКУШ ЗАТВЕРДЖЕННЯ

\_\_\_\_\_\_\_\_\_\_\_\_Інструкція системного програміста\_\_\_\_\_\_\_\_\_\_\_\_

(найменування програми (документа)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(вид носія даних)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(обсяг програми (документа), арк., Кб)

*студента групи ІС-24 ІІІ курсу*

*Вереня О.І.*

*студента групи ІС-24 ІІІ курсу*

*Кмець М.І.*

*студент групи ІС-24 ІІІ курсу*

*Сидоров М.О.*

*студент групи ІС-24 ІІІ курсу*

*Тимчук А.О.*

2015

# **4 ІНСТРУКЦІЯ СИСТЕМНОГО ПРОГРАМІСТА**

## 4.1 Загальні відомості про програму

### 4.1.1 Призначення програми

Основним призначенням системи «СКПВТ» є:

* Удосконалення роботи підприємства з виготовлення та збору комп’ютерів;
* Створення простого та інтуїтивно зрозумілого сервісу оформлення замовлень для клієнта заводу зі складання комп’ютерів за допомогою:
* Надання графічного конструктора складання типу комп’ютера;
* Надання критеріїв оцінки готового комп’ютера(вартість, потужність, якість);
* Підвищення продуктивності роботи планувальників заводу зі складання комп’ютерів за допомогою:
* Надання зручного інтерфейсу зворотнього зв’язку з замовником стосовно розглянутих замовлень;
* Надання автоматизованої системи розподілу працівників за прийнятими замовленнями;
* Подача розподілу замовлень у вигляді календарного плану;
* Створення інтерфейсу керування ресурсами(робітники, деталі) заводу зі складання комп’ютерів

### 4.1.2 Функції сервісу

Розробка системи будувалася на основі принципу модульності її підсистем з чітким розділенням функцій між ними.

Перелік підсистем системи «СКПВТ»:

Реєстрації, авторизації та верифікація користувачів – здійснює реєстрацію нових користувачів на основі даних, що вводяться ним, та авторизацію згідно обраного логіна та пароля, верифікація користувачів здійснюється шляхом відправлення на електронну пошту повідомлення з посиланням на підтвердження реєстрації.

Перелік функцій:

* Реєстрація
* Авторизація
* Підтвердження особи
  + Перхід по посиланню, що було вислане на електронну пошту, що була вказана при реєстрації

Відображення і взаємодія зі сторінками користувачів – відображення сторінок звичайних користувачів, планувальника чи менеджера ресурсів.

Перелік функцій:

* Перегляд інформації статус замовлення
* Перегляд складу замовлення
* Створення нового замовлення
* Додавання чи видалення інформації про робітників чи деталі

Складання розкладу – планування нових замовлень та репланування тих замовлень, які знаходяться у стані виконання.

Перелік функцій:

* Визначення періоду часу обробки кожної деталі кожним робітником над кожним
* Отримання Планувальником календарного плану вразі вдалого репланування
* Повідомлення користувача та планувальника вразі невдалого репланування
* Пошук робітників, що не виконують замовлень

Керування замовленням – підсистема, що включає в себе можливість керування замовленням як зі сторони користувача, так і зі сторони планувальника, що включає в себе створення, перегляд, додавання у заплановані, скасування замовлення.

Перелік функцій:

* Створення нового замовлення
* Скасування замовлення користувачем, що не перейшло в стан виконуваного
* Перегляд замовлень
* Скасування замовлення планувальником, перед відправкою на планування
* Додавання замовлення на планування

Керування інформацією про ресурси – включає в себе можливість керування такими ресурсами системи як робітники та деталі: їх редагування, видалення або додавання.

Перелік функцій:

* Видалення інформації про деталі
* Додавання інформації про деталі
* Наймання робітників
* Звільнення робітників

### 4.1.3 Мінімальний склад технічних засобів

Вимоги до технічного забезпечення серверу:

* Процесор – 1 Ггц, 1 ядро ЦП або краще
* Оперативна пам’ять не менш ніж 256 Мб
* Не менше ніж 5 ГБ ПЗУ
* 10 Мбіт/сек доступу до мережі Інтернет

Вимоги до технічного забезпечення серверу бази даних:

* Процесор – 1 Ггц, 1 ядро ЦП або краще
* Оперативна пам’ять не менш ніж 256 Мб
* Не менше ніж 5 ГБ ПЗУ
* 10 Мбіт/сек доступу до мережі Інтернет

### 4.1.4 Мінімальний склад програмних засобів

Вимоги до програмного забезпечення серверу:

* Java EE 7;
* Tomcat 7 або вище;

Вимоги до програмного забезпечення серверу бази даних:

* СУБД MySQL

Вимоги до програмного забезпечення клієнта:

* Браузер IE 10 або вище, оновлені Firefox, Chrome, Opera, Safari або будь-який інший браузер.

### 4.1.5 Вимоги до персоналу (планувальника, менеджера ресурсів)

Планувальник, менеджер ресурсів і користувачі повинні мати досвід роботи з персональним комп’ютером на рівні кваліфікованого користувача і вільно виконувати базові операції ОС. Також необхідно мати мінімальні навики користування браузером.

## 4.2 Структура системи

### 4.2.1 Відомості про структуру сервісу

Система «СКПВТ» складається з таких сторінок:

* Авторизація;
* Перегляд замовлень для користувача;
* Переглад замовлень для планувальника;
* Створення нового замовлення;
* Перегляд детальної інформації конкретного замовлення;
* Прегляд існуючих робітників та деталей;

### 4.2.2 Відомості про складові частини програми

* Авторизація:

Сторінка, за допомогою якого користувач входить в сервіс. Для цього треба ввести свій логін і пароль у відповідні текстові поля.

* Перегляд замовлень користувача:

Сторінка, на якій користувач може переглянути усі його замовлення та їх статуси, відмінити за необхідністю замовлення.

* Переглад замовлень для планувальника:

Сторінка, на якій планувальник може переглянути усі існуючі замовлення та їх статуси, відправити нові замовлення на планування, відмінити замовлення, відправити користувачу лист з проханням зкорегувати замовлення.

* Створення нового замовлення:

Сторінка, на якій користувач може створювати нові замовлення;

* Перегляд детальної інформації конкретного замовлення:

Сторінка, що містить інформацію про конкретне замовлення: його склад, статус, строк виконання;

* Прегляд існуючих робітників та деталей:

Сторінка, що містить інформацію про працівників та деталі, що наявні на підприємстві, та за необхідністю надає можливість редагувати та додавати нову інформацію.

### 4.2.3 Відомості про зв'язки між складовими частинами програми

Переходи між частинами системи здійснюються за допомогою відповідних кнопок та гіперпосилань.

### 4.2.4 Відомості про зв'язки з іншими програмами

В ході своєї роботи система використовує JavaPlanner для роботи з календарем: відображенням інформації про сплановані замовлення на календарі для наочності.

## 4.3 Налаштування програми

### 4.3.1 Налаштування сервера

Сервіс потребує мінімальні налаштування. Перше – це запуск самого сервера. Та друге – запуск самої системи, який виконується, як запуск звичайного Java застосування.

### 4.3.2 Налаштування на стороні користувача.

Зі сторони користувача виконувати ніяких налаштувань не треба. Йому для користування достатньо встановленого браузера.

## 4.4 Перевірка програми

Для тестування системи «СКПВТ» було застосоване модульне тестування. Менеджером разом з головним програмістом було вирішено протестувати модуль створення розкладу.

### 4.4.1 Постановка задачі тестування

Є таблиці access\_lists, fests, band\_lists, documents, bandsта fests. Таблиця access\_lists містить записи про додаткові права та можливості зареєстрованих користувачів. Таблиця festsмістить записи про існуючі концерти. Таблиця band\_lists містить записи про участь зареєстрованих користувачів у існуючих (зареєстрованих) гуртах. Таблиця documentsмістить записи про додані відео файли гуртів.

Є метод із сигнатурою:

* boolean can (User user, symbol access, Object object);
* де user – об’єкт користувача, щодо якого перевіряються права;
* access – символ, що представляє право для перевірки;
* object – об’єкт, стосовно якого перевіряється права користувача.

Необхідно отримати відповідь чи користувач має передане право на даний об’єкт. Мова йтиме про додаткові (нетривіальні) права користувачів. Тривіальним правом є такі як переглядати незареєстрованим користувачем сторінку гурту і таке інше.

### 4.4.2 Створення коду програми

В якості засобів розробки обрано мову програмування Ruby, середовище – RubyMine, фреймворк тестування – RoRminitests.

Функція can розглядатиме шість різних випадків:

* user = nil. Передано незалогіненого користувача – в нього немає додаткових прав. Зпрограмної точки зору – result =false;
* user != nil &&object.class == Fest and access == :manage.Користувач залогінений і перевірятиме права щодо керування концертом. Відповідь залежатиме від того, чи має користувач право на керування організатором цього концерту;
* user != nil && object.class = Band && (access == :upload || access == :write\_blog). Користувач залогінений і перевіряє права на завантаження додаткових документів (відео) та написання блогу. Відповідь залежатиме чи даний користувач є членом гурту чи його організатором;
* user != nil &&object.class == Document && access == :delete. Користувач є залогіненим і перевіряє права на видалення документу гурту. Відповідь залежатиме від того чи є даний користувач організатором гурту;
* user != nil && object.class == Blog. Користувач залогінений і перевіряє права щодо блогу. Відповідь залежить від того чи є даний користувач учасником чи організатором даного гурту;
* user != nil && access == :manage && !(object.class == Fest || object.class == Blog). Користувач залогінений і перевіряє права на керування об’єктом, що не є блогом чи концертом. Відповідь залежить від наявності запису про можливість керування у таблиці access\_lists;
* access != :manage && !(object.class = Band access == :upload) && !(object.class == Document && access == :delete) &&object.class != Blog. Користувач залогінений і перевіряє іншу комбінацію прав та об’єкта, що не потрапила в попередні обмеження;

Таким чином, повний текст коду даної функції матиме наступний вигляд:

|  |
| --- |
| defself.can? user, access, object  unlessuser.nil?  if object.class.to\_s == Fest.to\_s && (access == :manage || access == :write\_blog)  returnwhere(object\_type: Organizer.to\_s, user\_id: user.id).joins("joinfestsas f onaccess\_lists.object\_id = f.organizer\_id").exists?  elsif object.class.to\_s == Band.to\_s && :upload == access  f = BandList.where(user\_id: user.id, band\_id: object.id).exists?  f = f || where(object\_type: Band.to\_s, object\_id: object.id, user\_id: user.id, access: :manage )  return f  elsif object.class.to\_s == Document.to\_s && :delete == access  returnwhere(object\_type: Band.to\_s, user\_id: user.id, access: :manage).  joins("joindocumentsas d on d.band\_id = access\_lists.object\_id andd.id = #{object.id}").exists?  elsif object.class.to\_s == Blog.to\_s  f = where(object\_id: object.id, object\_type: Blog.to\_s, user\_id: user.id, access: access).exists?  f = f || where(object\_id: object.band.id, object\_type: Band.to\_s, user\_id: user.id, access: "manage").exists?  return f  end  ifaccess == :manage  returnwhere(object\_type: object.class, object\_id: object.id, user\_id: user.id, access: access).exists?  else  returnwhere("object\_type = \"#{object.class}\" andobject\_id = #{object.id} anduser\_id = #{user.id} and (access = \"#{access}\" oraccess = \"manage\")").exists?  end  else  returnfalse  end  end |

### 4.4.3 Тестування

Необхідно протестувати код створеної функції can.

Для цього побудуємо потоковий граф коду цієї функції. Він представлений на рисунку 4.1.



Рисунок 4.1 – Структурна схема потокового граву функції тестування

де fun1 – результат виклику функції

*where(object\_type: Organizer.to\_s, user\_id: user.id).joins("joinfestsas f onaccess\_lists.object\_id = f.organizer\_id").exists?*

fun 2 – результат виклику функції

*BandList.where(user\_id: user.id, band\_id: object.id, result: 1).exists?||where(object\_type: Band.to\_s, object\_id: object.id, user\_id: user.id, access: :manage ).exists?*

fun3 – результат виклику функції

*where(object\_type: Band.to\_s, user\_id: user.id, access: :manage).*

*joins("joindocumentsas d on d.band\_id = access\_lists.object\_id andd.id = #{object.id}").exists*?

fun4 – результат виклику функції

*where(object\_id: object.id, object\_type: Blog.to\_s, user\_id: user.id, access: access).exists?||where(object\_id: object.band.id, object\_type: Band.to\_s, user\_id: user.id, access: "manage").exists?*

fun5 – результат виклику функції

*where(object\_type: object.class.to\_s, object\_id: object.id, user\_id: user.id, access: access).exists?*

fun 6 – результат виклику функції

*where("object\_type = \"#{object.class.to\_s}\" and object\_id = #{object.id} and user\_id = #{user.id} and (access = \"#{access}\" or access = \"manage\")").exists?*

Визначимо цикломатичну складність потокового графу трьома способами:

1.  регіонів.
2.  дуг  вузлів .
3.  предикатних вузлів .

Отже, потрібно підготувати тестові варіанти вхідних даних для 7 незалежних шляхів у цьому потоковому графі. Легко переконатися, що такі тестові дані відповідатимуть 7 випадкам, які були описані у п.2.

Щоб приступити до створення груп тестів, необхідно створити спеціальні файли – fixtures – якібудутьгенеруватитимчасовубазуданихінаповнюватимутьїїтестовимизначеннями, наякихіпрацюватимутьтести. Наведемо лістинг кожного файла нижче.

access\_lists.yml

one:

object\_type: <%= Band.to\_s %>

object\_id: 1

access: :manage

user: user3

two:

object\_type: <%= Organizer.to\_s %>

object\_id: 1

access: :manage

user: user3

band\_list.yml

bandList1:

user: user1

band\_id: 1

result: 1

bands.yml

band1:

id: 1

name: "test\_band 1"

#

band2:

id: 2

name: "test\_band 2"

blogs.yml

blog1:

header: "blog 1"

band\_id: 1

#

blog2:

header: "blog2"

documents.yml

document1:

band\_id: 1

name: doc1

fests.yml

fest1:

name: "fest1"

organizer\_id: 1

#

fest2:

name: "fest 2"

organizers.yml

org1:

name: "org1"

id: 1

#

org2:

name: "org2"

users.yml

user1:

email: "tu11@u.u"

encrypted\_password: $2a$10$vFdMDebe4mVT0zG4EFEueOfBkyaSsCKpnisetUmwXBD.j5ztMztey

name: "user 1 in band 1"

user2:

email: "tu12@u.u"

encrypted\_password: $2a$10$vFdMDebe4mVT0zG4EFEueOfBkyaSsCKpnisetUmwXBD.j5ztMztey

name: "user 2 in band 1"

user3:

email: "tu13@u.u"

name: "user 3 in org 1"

encrypted\_password: $2a$10$vFdMDebe4mVT0zG4EFEueOfBkyaSsCKpnisetUmwXBD.j5ztMztey

user4:

name: "user 4 in org 2"

email: "tu23@u.u"

encrypted\_password: $2a$10$vFdMDebe4mVT0zG4EFEueOfBkyaSsCKpnisetUmwXBD.j5ztMztey

Таким чином, створимо 7 груп тестів, які відповідають 7 різним шляхам у потоковому графі. Кожна з цих груп повинна містити принаймні по одному тесту з відомим набором вхідних та очікуваних вихідних даних. Занесемо інформацію про ці тести у таблицю (таблиця 4.1).

Таблиця 4.1 – Варіанти тестів для різних шляхів у потоковому графі

| **Група тестів** | **Тест** | **Вхідні дані  – параметри метода** can?(user, access, object) | **Очікуваний результат**  **(значення** true **або**false **– результат виклику** AccessList.can?**)** |
| --- | --- | --- | --- |
| usercanwriteblog | Example1 | (user1, :write\_blog, band1) | true |
| usercanmanageorg | Example1 | (user3, :manage, band1) | true |
| usercanmanagefest | Example1 | (user3, :manage, fest1) | true |
| usercandeletedoc | OnlyExample | (user3, :delete, document1) | true |
| usercanmanageblog | Example1 | (user3, :manage, blog1) | true |
| usercanupdateband | Example1 | (user3,:update,band1) | true |
| nilcandoanything | Example1 | (nil, :update,band1) | true |

Реалізуємо ці тести в окремому файлі вихідного коду, використовуючи фреймворкminitest:

|  |
| --- |
| require'test\_helper'  classAccessListTest<ActiveSupport::TestCase  test "usercanwriteblog" do  assert\_equal true, AccessList.can?(users(:user1), :write\_blog, bands(:band1))  end  test "usercanmanageorg" do  assert\_equal true, AccessList.can?(users(:user3), :manage, bands(:band1))  end  test "usercanmanagefest" do  assert\_equal true, AccessList.can?(users(:user3), :manage, fests(:fest1))  end  test "usercandeletedoc" do  assert\_equal true, AccessList.can?(users(:user3), :delete, documents(:document1))  end  test "usercanmanageblog" do  assert\_equal true, AccessList.can?(users(:user3), :manage, blogs(:blog1))  end  test "usercanupdateband" do  assert\_equal true, AccessList.can?(users(:user3), :update, bands(:band1))  end  test "nilcandoanything" do  assert\_equal false, AccessList.can?(nil, :update, bands(:band1))  end  end |

Результат проходження створених тестів представлено на рисунку 4.2.

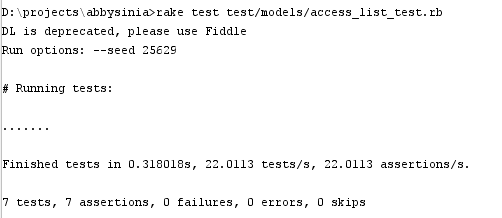


Рисунок 4.2 – Результат виконання тестів з таблиці 1 за допомогою фреймворкуminitests

### 4.4.5 Повний код програми

|  |
| --- |
| AccessList.rb |
| classAccessList<ActiveRecord::Base  belongs\_to :user  defself.can? user, access, object  unlessuser.nil?  if object.class.to\_s == Fest.to\_s &&access == :manage  returnwhere(object\_type: Organizer.to\_s, user\_id: user.id).joins("joinfestsas f onaccess\_lists.object\_id = f.organizer\_id").exists?  elsif object.class.to\_s == Band.to\_s && (:upload == access || access == :write\_blog)  f = BandList.where(user\_id: user.id, band\_id: object.id, result: 1).exists?  f = f || where(object\_type: Band.to\_s, object\_id: object.id, user\_id: user.id, access: :manage ).exists?  return f  elsif object.class.to\_s == Document.to\_s && :delete == access  returnwhere(object\_type: Band.to\_s, user\_id: user.id, access: :manage).  joins("joindocumentsas d on d.band\_id = access\_lists.object\_id andd.id = #{object.id}").exists?  elsif object.class.to\_s == Blog.to\_s  f = where(object\_id: object.id, object\_type: Blog.to\_s, user\_id: user.id, access: access).exists?  f = f || where(object\_id: object.band.id, object\_type: Band.to\_s, user\_id: user.id, access: "manage").exists?  return f  end  ifaccess == :manage  returnwhere(object\_type: object.class.to\_s, object\_id: object.id, user\_id: user.id, access: access).exists?  else  returnwhere("object\_type = \"#{object.class.to\_s}\" andobject\_id = #{object.id} anduser\_id = #{user.id} and (access = \"#{access}\" oraccess = \"manage\")").exists?  end  else  returnfalse  end  end  end |

\_\_\_\_\_\_\_\_\_\_\_\_Інструкція користувача\_\_\_\_\_\_\_\_\_\_\_\_

(найменування програми (документа)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(вид носія даних)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(обсяг програми (документа), арк., Кб)

*студента групи ІС-24 ІІІ курсу*

*Вереня О.І.*

*студента групи ІС-24 ІІІ курсу*

*Кмець М.І.*

*студент групи ІС-24 ІІІ курсу*

*Сидоров М.О.*

*студент групи ІС-24 ІІІ курсу*

*Тимчук А.О.*

2015

ЗАТВЕРДЖУЮ:

Керівник курсового проекту

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_*ас. Сперкач М.О.\_\_\_\_*

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 20\_\_р.

АРКУШ ЗАТВЕРДЖЕННЯ

\_\_\_\_\_\_\_\_\_\_\_\_Інструкція користувача\_\_\_\_\_\_\_\_\_\_\_\_

(найменування програми (документа)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(вид носія даних)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(обсяг програми (документа), арк., Кб)

*студента групи ІС-24 ІІІ курсу*

*Вереня О.І.*

*студента групи ІС-24 ІІІ курсу*

*Кмець М.І.*

*студент групи ІС-24 ІІІ курсу*

*Сидоров М.О.*

*студент групи ІС-24 ІІІ курсу*

*Тимчук А.О.*

2015

# **5 ІНСТРУКЦІЯ КОРИСТУВАЧА**

## 5.1 Вступ

### 5.1.1 Область застосування

Система «СКПВТ» розроблена для використання у сфері виробництва. А саме для підприємств, які прагнуть автоматизувати систему планування виготовлення компьютерів.

### 5.1.2 Короткий опис можливостей

Система «СКПВТ» передбачає такі можливості:

* Створення нових замовлень;
* Створення календарного плану для виготовлення товарів;
* Менеджмент нових замовлень;
* Менеджмент підприємства, шляхом заміни робітників та деталей;

### 5.1.3 Рівень підготовки користувача

Користувачі повинні мати досвід роботи з персональним комп’ютером та брузером на рівні звичайного користувача.

### 5.1.4 Перелік експлуатаційної документації

Перелік експлуатаційних документів, з якими необхідно ознайомитися:

* Інструкція користувача;
* Інструкція адміністратора.

## 5.2 Призначення і умови застосування

### 5.2.1 Види діяльності, функції

### Система «СКПВТ» призначена як для великих організацій, так і для звичайних користувачів, які хочуть зібрати собі комп’ютер, який відповідає усім їхнім потребам та можливостям. Цей ресурс дозволяє зареєстрованим користувачам створювати нові замовлення на збір копютерів (одного чи багатьох) та задавати строки на виконання цих замовлень. Також ця система дозволяє отримати своєчасну та швидку інформацію стосовно статусу нового замовлення, можливість чи неможливість його виконання, а також дату закінчення виконання замовлення. Кожен користувач, який створив нове замовлення може відмінити його, якщо воно ще не перейшло в стан виконуваного. У разі відмови у виконанні або успішному закінченню замовлення, користувач отримує своєчасне повідомлення на електронну пошту, зазначену при реєстрації.

### 5.2.2 Програмні та апаратні вимоги до системи

Вимоги до технічного забезпечення серверу:

* Процесор – 1 Ггц, 1 ядро ЦП або краще
* Оперативна пам’ять не менш ніж 256 Мб
* Не менше ніж 5 ГБ ПЗУ
* 10 Мбіт/сек доступу до мережі Інтернет

Вимоги до технічного забезпечення серверу бази даних:

* Процесор – 1 Ггц, 1 ядро ЦП або краще
* Оперативна пам’ять не менш ніж 256 Мб
* Не менше ніж 5 ГБ ПЗУ
* 10 Мбіт/сек доступу до мережі Інтернет

Вимоги до програмного забезпечення серверу:

* Java EE 7;
* Tomcat 7 або вище;

Вимоги до програмного забезпечення серверу бази даних:

* СУБД MySQL

Вимоги до програмного забезпечення клієнта:

* Браузер IE 10 або вище, оновлені Firefox, Chrome, Opera, Safari або будь-який інший браузер.

## 5.3 Опис операцій

Інтерфейс системи «СКПВТ» є логічним та простим. Тому проблем з користуванням ресурсом виникнути не повинно.

Тому, проаналізувавши інтерфейс, було прийнято рішення у керівництві користувача описати і продемонструвати процес верифікації користувача. Робота з системою починається з вікна яке ми продемонстрували на рисунку 5.1.

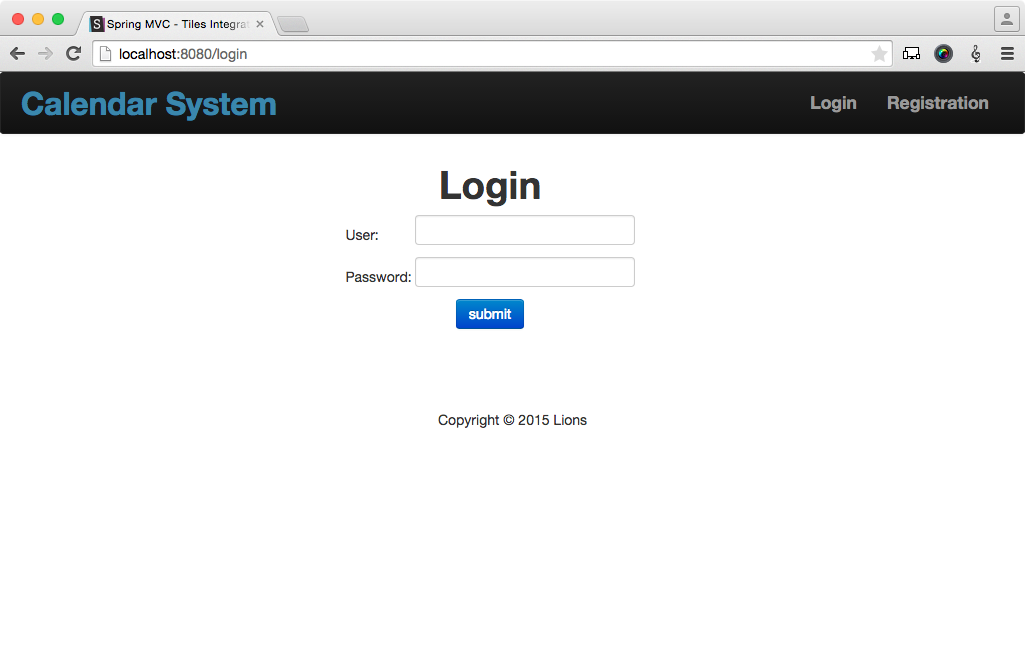


Рисунок 5.1 – Екранна форма реэстрації у «СКПВТ»

Зараєстрований користувач може виконати вхід в систему. Користувач, який ще не зареєструвався може це зробити перешовши до форми реєстрації.

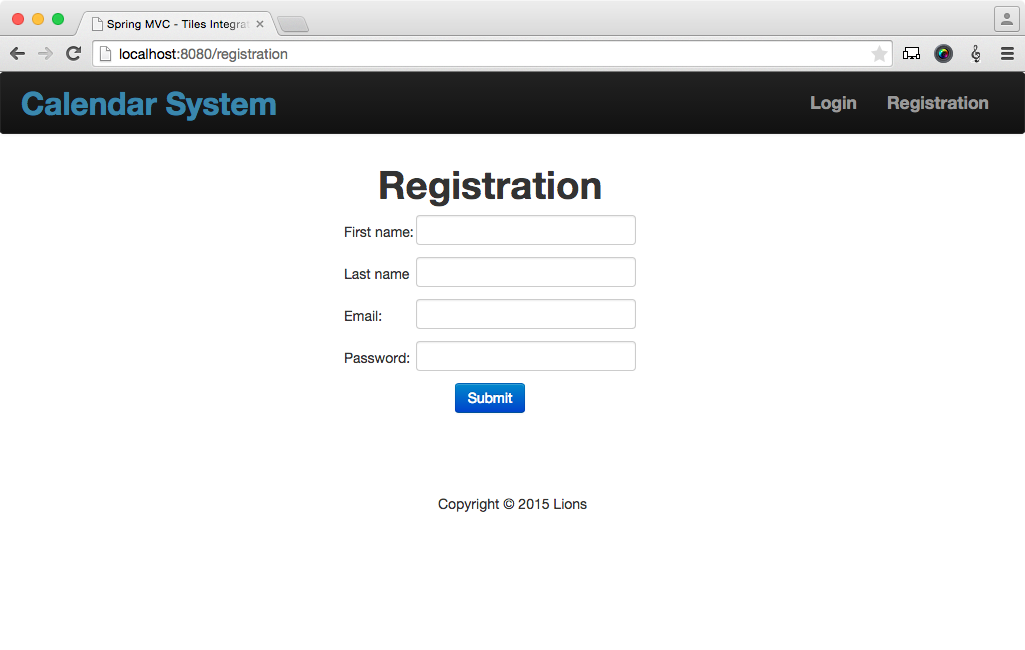


Рисунок 5.2 – Екранна копія форми реєстрації.

Після того як користувач залогінився він опиняється на сторінці списку своїх замовлень. За допомогою цієї сторінки користувач може скасувати своє замовлення або отримати змогу переглянути деталі свого замовлення.

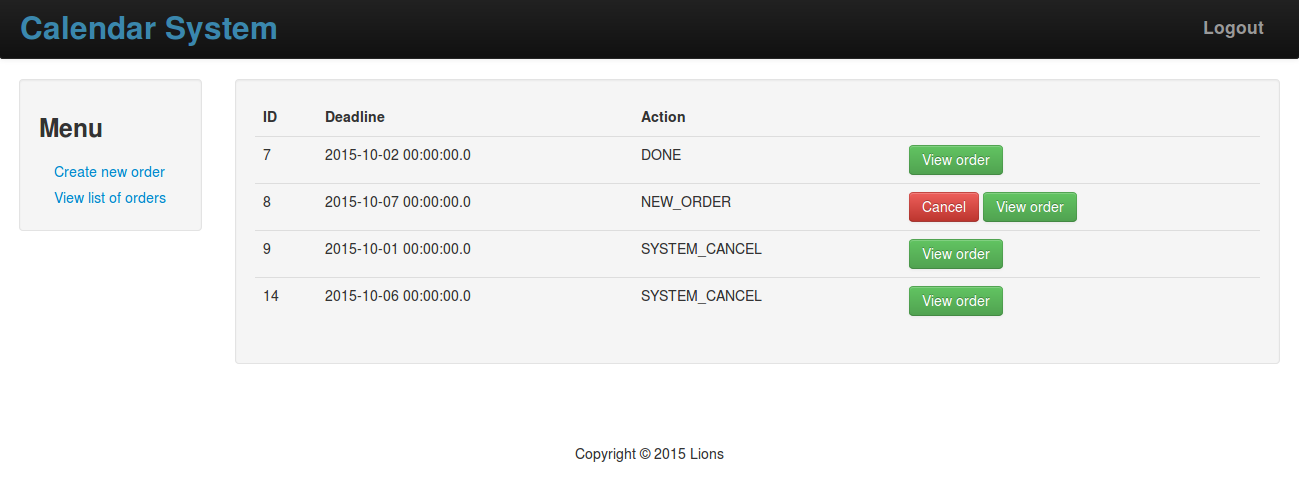


Рисунок 5.3 – Сторінка перегляду списку замовлень

Для того щоб переглянути детальну інформацію про замовлення необхіжно натиснути на кнопку “View order”.

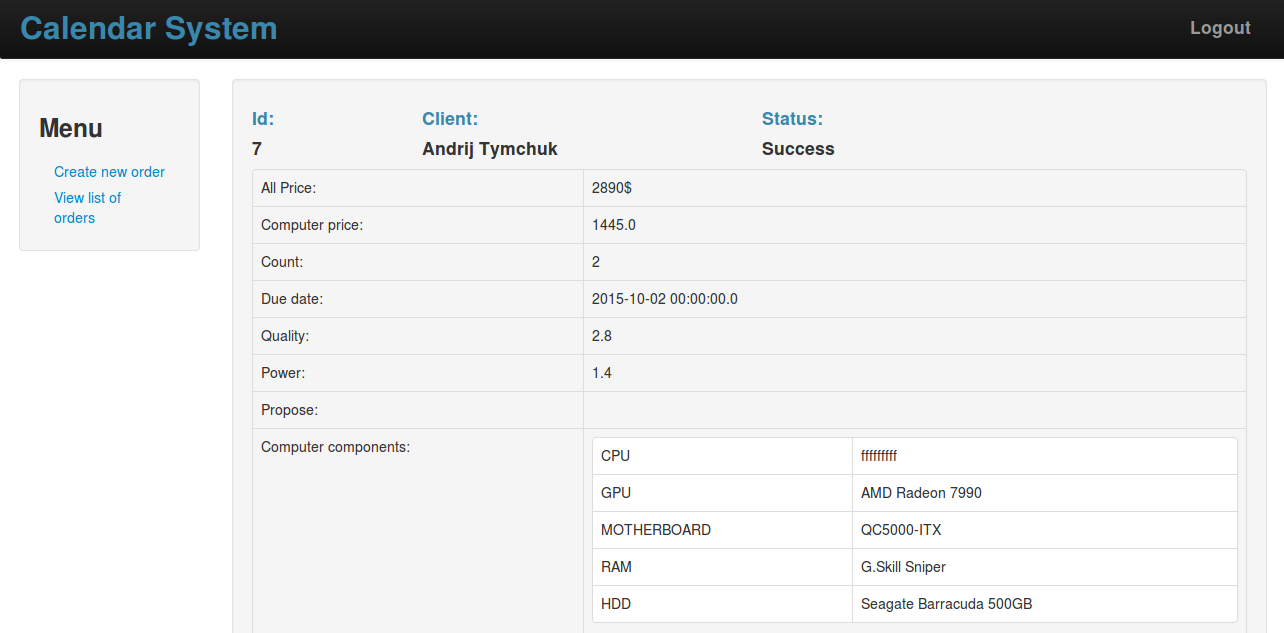


Рисунок 5.4 – Екранна копія сторінки перегляду інформації про замовлення

Для створення нового замовлення необхідно перейти за посиланням яке знаходиться у боковому меню та заповнити форму.

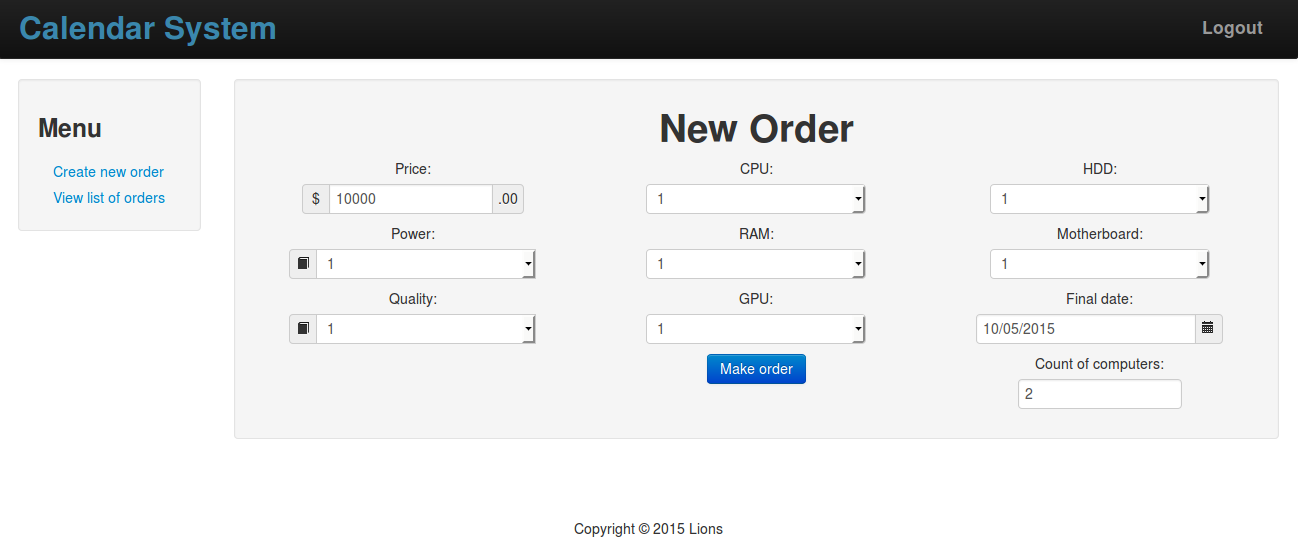


Рисунок 5.5 – Екранна копія процесу подачі заявки

Після натиснення кнопки «Надіслати», ми чекаємо на відповідь від планувальника, а заявка зявляється у списку заявок. Це продемонстровано на рисунку 5.6.

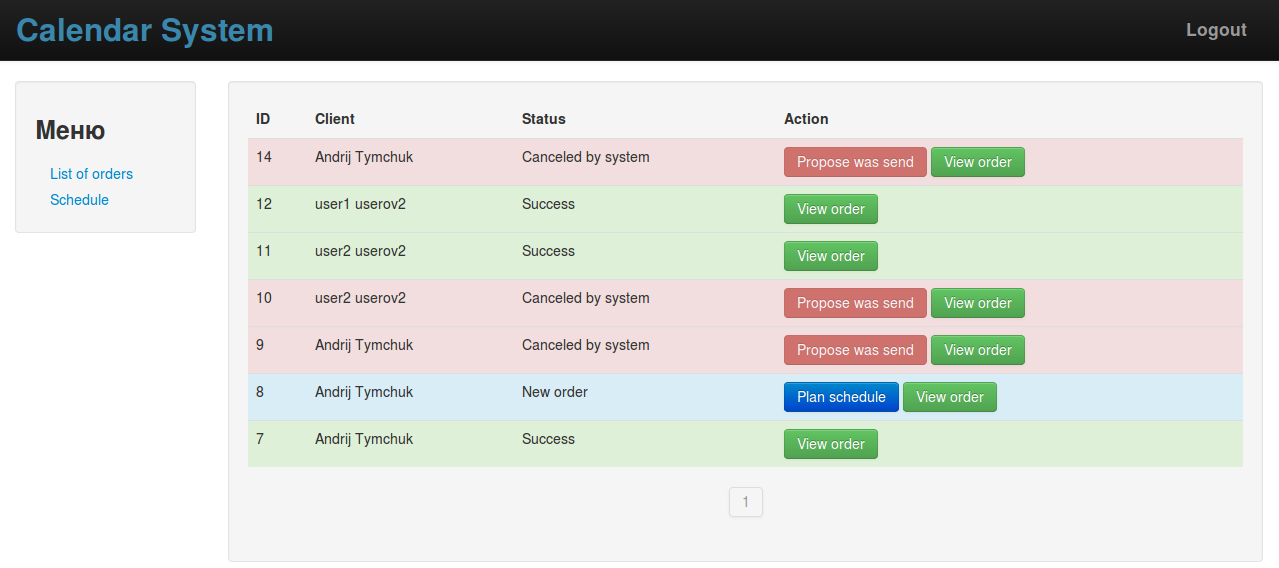


Рисунок 5.6 – Список замовлень планувальника

Коли планувальник спланує замовлення, то його статус у системі змінитьсяпро що користувач буде попереджений. Після виконання замовлення замовнику також прийде повідомлення на його електронну адресу яку він вказував при реєстрації.

# 6 ВИСНОВКИ

## 6.1 Результати роботи

По результатах проведених робіт було створено онлайн-систему для підприємств. Реалізовано майже повною мірою запланований функціонал. Тобто, повною мірою функціонують сторінки користувачів, планувальника та менеджера ресурсів. Коректно працюють алгоритми, що оптимізують планування нових замовлень та репланування старих. Коректно виконується надсилання повідомлень на електронну пошту, робота з інтерфейсами користувача, планувальника, менеджера ресурсів; додавання та скасування замовлень, робота з замовленнями. Також коректно проходить валідація користувача.

Основні проблеми з якими зіштовхнулася команда:

* мала кількість членів команди, на одну людину лягали завдання і менеджера, і тестувальника, і дизайнера;
* виникали проблеми при роботі з правами, а саме з виділенням унікальних прав на певний ресурс;
* проблеми були і з крос табличним пошуком по таблицях.

Загалом же команда задоволена результатом, тому що було отримано дуже велику кількість корисних навиків. Зокрема розробка на Java EE, робота в команді, розробка документації по проекту та ін.

Отже, роботу команди можна оцінювати позитивно, а проект досить якісним.

## 6.2 Виконана функціональність відповідно до ТЗ

Із задач, зазначених в ТЗ, були виконані наступні:

1. Задача створення замовлення та отримання результату:

* Отримання від замовника критеріїв по створенню комп’ютерів (кількість, термін, якість, ціна, потужність);
* Відправлення замовлення до планувальника;
* Отримання результату планування;

1. Задача складання календарного плану:

* Перевірка можливості виконання замовлення;
* Перегляд альтернативних варіантів замовлення;
* Додавання в календарний план нового замовлення;
* Видалення існуючих замовлень;
* Перегляд календарного плану;

1. Задача керування інформацією про ресурси:

* Зміна, видалення, додавання інформації про наявних майстрів;
* Зміна, видалення, додавання інформації про деталі;

1. Задача аутентифікації в системі:

* Реєстрація замовників, планувальників та замовників ресурсів;
* Вхід в систему з правами відповідно до вищенаведених ролей;

## 6.3 Невиконана функціональність відповідно до ТЗ

Задачі, які не було виконано в рамках розрахункової роботи:

* Друк детальної інформації про замовлення;

## 6.4 Перспективи розвитку програми

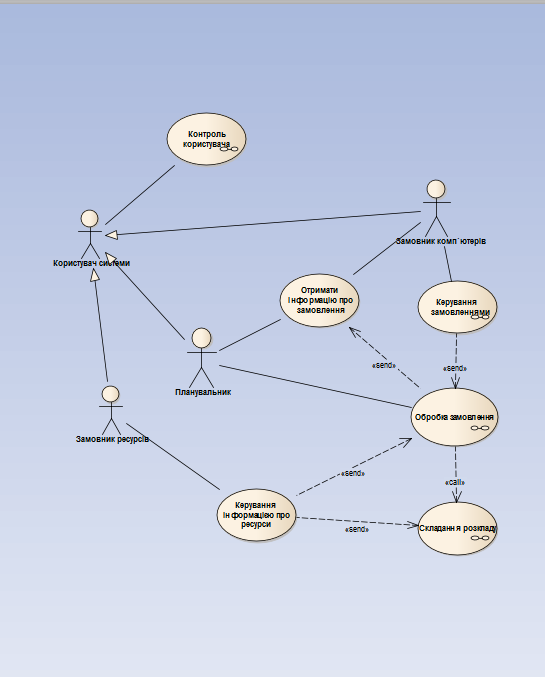
В майбутньому планується реалізувати ширший функціонал, розробити яскравіший дизайн, зробити можливість більш гнучкого керування процесами виробництва.

# 7 ПЕРЕЛІК ПОСИЛАНЬ

1. [Вгору](http://uk.wikipedia.org/wiki/Java#cite_ref-2) J. Gosling, B. Joy, G. Steele, G. Brachda. [The Java Language Specification, 2nd Edition](http://www.webcitation.org/6GYqdEjc4). 2005
2. [Шмелев В.В.](https://ru.wikipedia.org/wiki/%D0%A8%D0%BC%D0%B5%D0%BB%D1%91%D0%B2,_%D0%92%D0%B8%D0%BA%D1%82%D0%BE%D1%80_%D0%92%D0%B0%D1%81%D0%B8%D0%BB%D1%8C%D0%B5%D0%B2%D0%B8%D1%87) Метод упорядочения в задачах календарного планирования. Препринт.— М.: ВНИИСИ, 1983. — с.5—16.
3. Конвей Р.В., Максвелл В.Л., Миллер Л.В. Теория расписаний.—М.: «Наука», 1975., стр. 144—146.
4. S.M. Johnson, Optimal two- and three-stage production schedules with setup times included, Naval Res. Log. Quart. I(1954)61-68.
5. Танаев В.С., Гордон В.С., Шафранский Я.М. Теория расписаний. Одностадийные системы.- М.: Наука, 1984.
6. Poli, R., Langdon, W. B., McPhee, N. F. (2008). A Field Guide to Genetic Programming. Lulu.com, freely available from the internet. [ISBN](http://uk.wikipedia.org/wiki/International_Standard_Book_Number) [978-1-4092-0073-4](http://uk.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B5%D1%86%D1%96%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0:%D0%94%D0%B6%D0%B5%D1%80%D0%B5%D0%BB%D0%B0_%D0%BA%D0%BD%D0%B8%D0%B3/978-1-4092-0073-4).
7. [Субботін С.О., Олійник А.О., Олійник О.О. Неітеративні, еволюційні та мультиагентні методи синтезу нечіткологічних і нейромережних моделей: Монографія / Під заг. ред. С.О. Субботіна. – Запоріжжя: ЗНТУ, 2009. – 375 с.](http://www.csit.narod.ru/subject/mag_SShI/mono.pdf)
8. Кларенс Хо, Роб Харроп. [Spring 3 для профессионалов](http://www.williamspublishing.com/Books/978-5-8459-1803-1.html) = Pro Spring 3. — М.: [«Вильямс»](https://ru.wikipedia.org/w/index.php?title=%D0%92%D0%B8%D0%BB%D1%8C%D1%8F%D0%BC%D1%81_(%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE)&action=edit&redlink=1), 2012. — 880 с. — [ISBN 978-5-8459-1803-1](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9785845918031).

# ДОДАТОК А.1

## Use Case діаграма

Рисунок А..1 – діаграма Use Case

# ДОДАТОК А.3

## Діаграма DFD

## 

Рисунок А.3.1 – Діаграма DFD

# ДОДАТОК А.4

## Діаграма компонентів

Рисунок А.4.1 – Діаграма компонентів