Міністерство освіти і науки України

Одеський національний політехнічний університет

Інститут комп’ютерних систем

Кафедра системного програмного забезпечення

ЗВІТ

з виробничої практики

Спеціальність 6.050103 – Програмна інженерія

На тему: «Клієнт-серверна програма для обчислення перспективного прибутку від інвестиційної діяльності»

Виконав студент групи АС-123 \_\_\_\_\_\_\_\_\_\_\_\_\_ П.С. Буслюк

(підпис) (ПІБ)

Керівник практики

від підприємства \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_ О.М. Паулін

(посада) (підпис) (ПІБ)

Керівник практики

від університету \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(посада) (підпис) (ПІБ)

МП

Практика захищена з оцінкою \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Член комісії по захисту \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(посада) (підпис) (ПІБ)

Член комісії по захисту \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(посада) (підпис) (ПІБ)

Одеса

2016

**Реферат**

Курсова робота містить 32 сторінки, в т.ч. 26 рисунків, 11 таблиць та3посилань.

У практичній роботі розроблено архітектурний проект для програмного комплексу розрахунку перспективного прибутку від інвестиційної діяльності.

Метою роботи є автоматизація процесу обчислень необхідних даних та створення єдиної системи відновлення даних із сторонніх джерел.

Методи розробки базуються на технологіях: java, h2 (БД на стороні сервера), xml (файли на стороні клієнта), soketIO (система передачі даних), javaMail (api передачі даних на email адресу).

Ключові слова: java, обчислення перспективного прибутку, h2, передача даних на email java

Зміст

[Вступ 4](#_Toc423196038)

1 СПЕЦИФІКАЦІЯ ВИМОГ ДО СИСТЕМИ………………………………...5

[1.1 Функціональні вимоги до системи 5](#_Toc423196039)

[1.1.1 Опис предметної області і аналогів системи 5](#_Toc423196040)

[1.1.2 Варіанти використання системи 7](#_Toc423196041)

1.2 Нефункціональні вимоги…………………………………………………10

[2 План виконання проекту 12](#_Toc423196042)

[2.1 Оцінка тривалості розробки 12](#_Toc423196043)

[2.2 Оцінка ризиків](#_Toc423196044) 16

[3 Проектування системи 18](#_Toc423196046)

[3.1 Проектування алгоритмів 18](#_Toc423196048)

[3.2 Проектування інтерфейсу користувача 19](#_Toc423196049)

[4 Програмна реалізація](#_Toc423196050) 20

[4.1 Програмні компоненти](#_Toc423196051) 20

[Висновки 21](#_Toc423196055)

[Список використаних джерел 22](#_Toc423196056)

ДОДАТОК А ПРОГРАМНИЙ КОД………………………………………….23

# Вступ

На сьогоднішній день автоматизація діяльності стає невід'ємною частиною практично будь-якої людини і підприємства. Управління різними процесами за допомогою комп'ютера дозволяє домогтися більш високої продуктивності праці і заощадити масу часу. Високоякісна автоматизація технологічних процесів значно полегшує роботу в цілому.

Для розробки програмного засобу необхідно вирішити завдання синтезу та аналізу. До завдань аналізу відносяться:

* аналіз предметної області;
* аналіз аналогів програмних системи;
* вибір засобів проектування бази даних;
* вибір засобів проектування програмного засобу.

До завдань синтезу відносяться:

* проектування і створення бази даних;
* розробка архітектури програмного засобу;
* реалізація алгоритмів програмного засобу;
* реалізація цілісності і безпеки даних;
* налагодження і тестування програмного засобу;
* розробка супровідної документації.

Розроблений програмний засіб повинен дозволити автоматизувати роботу по розрахунку та автоматичному відновленню даних.

1. СПЕЦИФІКАЦІЯ ВИМОГ ДО СИСТЕМИ
   1. Функціональні вимоги до системи
      1. Опис предметної області і аналогів системи

Інвестиційний проект - обгрунтування економічної доцільності, обсягу і термінів здійснення капітальних вкладень, у тому числі необхідна проектно-кошторисна документація, розроблена відповідно до законодавства України та затверджена в установленому порядку стандартів (нормами і правилами).

Інвестиційні проекти класифікуються за різними ознаками.

Залежно від сфери реалізації інвестиційні проекти бувають:

* організаційними;
* технологічними;
* економічними;
* соціальними;
* змішаними.

Залежно від масштабу реалізації виділяють такі інвестиційні проекти:

* монопроект - простий проект;
* мультипроекта - проект включає ряд монопроектов;
* мегапроект - проект, що включає монопроекти і мультипроекти.
* Залежно від предметної області інвестиційні проекти діляться:
* на інноваційні;
* навчально-освітні;
* змішані.

Тож, на основі цих даних можна сказати, що розроблюваний проект відноситься до змішаного мультипроекту, предметна область якого є «»

До альтернативних систем можна віднести наступні проекти:

* <http://avtoforex.ru/onlayn-kalkulyator-investora.html>

Увесь збір даних зведений до 4 полів, що недостатньо для функціонального розрахунку необхідних параметрів.

* <http://axd.semestr.ru/econ/vvp.php>

Програмний засіб орієнтований на розрахунок в більшості на підприємства.

* <http://www.glazavezde.ru/kalkulyator-dlya-rascheta-npv-irr.html>

ПЗ у якому є реінвестиції, але обрати зворотню функцію неможливо.

* Калькулятор

Ніякої автоматизації. Повністю необхідно перераховувати введені данні у разі неправильного вводу.

* Excel

Лише в даному табличному редакторі є функція збереження, але немає функції відновлення даних у разі виникнення непередбаченої ситуації із ОС.

В жодному з цих проектів немає збереження введених даних, окрім таблиць Excel, розділення на певні вкладення в інвестиційний проект та необхідну функціональність на розрахунок даних.

**1.1.2 Варіанти використання системи**

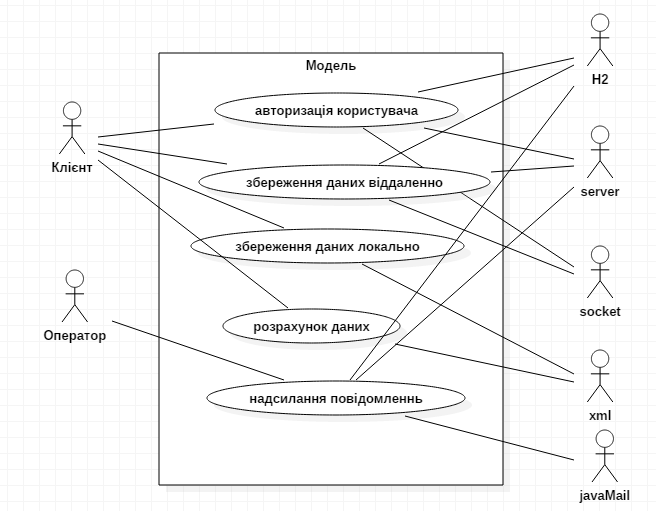


Рисунок 1.1 ­– Діаграма варіантів використання

Таблиця 1.1 – Сценарії використання

|  |  |  |
| --- | --- | --- |
| Варіант використання №1:  авторизація користувача | Передумова | К заповнив усі необхідні поля профілю |

Продовження таблиці 1.1

|  |  |  |
| --- | --- | --- |
|  | Післяумова | Сервер дав згоду на користування системою |
| Суб'єкт | Користувач(К), Сервер(С), Система Користувача (СК) |
| Основний потік | 1. К входить до програми та обирає профіль. СК пропонує збереження даних профілю. 2. СК відправляє дані авторизації на С. 3. С шукає в БД дані К та відсилає відповідь. |
| Альтернативний потік | 1а.1 К не згоджується із збереженням даних. СК не зберігає дані. Вихід з программи.  1б.1 К вже заповнив дані. Перехід до 2 пункту.  2а.1 С не знайшла К в БД. С створює К в БД та надає доступ.  3а.1 |
| Варіант використання №2:  Збереження даних локально | Передумова | К авторизований у СК; К вказав персональні дані. |
| Післяумова | Дані збережені |
| Суб'єкт | Користувач(К), Система Клієнта(СК) |

* 1. Продовження таблиці 1.1

|  |  |  |
| --- | --- | --- |
|  | Основний потік | 1. К обирає збереження даних. СК перевіряє дані та зберігає їх у XML файл |
| Альтернативний потік | 1а.1 Дані не коректні. СК видає повідомлення щодо невірних даних |
| Варіант використання №3:  Збереження даних віддалено | Передумова | К авторизований у СК; у К є доступ до Інтернет мережі; К вказав у персональні дані. |
| Післяумова | Обліковий запис збережений. |
| Суб'єкт | Користувач(К), Система Клієнта(СК), Сервер (С) |
| Основний потік | 1. К обирає збереження даних віддалено. СК перевіряє дані та відсилає їх на С. 2. С отримує їх та зберігає в БД |
| Альтернативний потік | 1а.1 К вказав невірні дані. СК пропонує ввести дані ще раз. |
| Варіант використання №4:розрахунок даних | Передумова | К авторизований у СК; |

Продовження таблиці 1.1

|  |  |  |
| --- | --- | --- |
|  | Післяумова | Дані розраховані |
| Суб'єкт | Користувач(К), Система Клієнта(СК) |
| Основний потік | 1. К обирає компанії для розрахунку. СК перевіряє дані, розраховує їх та видає результат. |
| Альтернативний потік | 2а.1 Дані невірні. СК видає повідомлення щодо перевірки введених даних |
| Варіант використання №5:  Надіслання повідомлення | Передумова | Є доступ до БД та мережі інтеренет. |
| Післяумова | Повідомлення відіслані вказаним адресатам. |
| Користувачі | Сервер(С), БД, Оператор(О), Клієнт(К) |
| Основний потік | 1. О обирає К для оповіщення та вводить сааме повідомлення. С встановлює кожного К email адресу та надсилає повідомлення. |
| Альтернативний потік | 1а.1 Невірна адреса. С пропускає К. |

* 1. Нефункціональні вимоги
* Здатність до взаємодії

Клієнт може взаємодіяти із клієнтською частиною ПЗ, робити розрахунки та отримувати нове ПЗ за згодою. Сервер веде збір даних клієнтів та зберігає дані в БД для подальшого інформування користувачів.

* Захищеність

Доступ до елементів керування інформаційної технології розрахунку можливий лише за сплати відповідних коштів, після чого оператор заносить клієнта по ідентифікаційному номеру до бази даних активних користувачів системи, тож повинно бути реалізована захищеність від злому ПЗ.

* Часові характеристики

Час на обробку розрахунків повинен становити від 1-3 секунди.

Час щодо відправлення запиту від клієнта до серверу, та навпаки, повинен відбуватись не більше 10 секунд.

Час на збереження даних користувача повинен становити близько 1-3 секунди.

* Використання ресурсів

Мінімальні ресурси системи для нормального функціонування програмного забезпечення:

ОС: Windows ХР;

Частота процесора: 1.2 ГГц;

RAM: 1 GB;

ROM: 1 GB;

Наявність підключення до мережі Інтернет із швидкістю не менше ніж: 1 Мб/с

* Стійкість до відмов

У випадку неможливості підключення до мережі Інтернет система має працювати в штатному режимі, лише якщо оплата відповідних послуг була проведена.

* Здатність до відновлення працездатності при відмовах

У випадках коли користувач навмисно пошкодить деякі файли, система повинна буде відновити дані із резервного сховища.

* Адаптованість

Система має бути сумісною із ОС Windows ХР та вище.

1. План виконання проекту
   1. Оцінка тривалості розробки

Типи акторів в системі:

Клієнт – кінцевий користувач.

Оператор ­­– кінцевий користувач.

soketIO – система взаємодії по протоколу TCP / IP.

H2 – БД, взаємодіюча по протоколу TCP / IP.

javaMail ­– бібліотека, взаємодіюча по протоколу TCP / IP.

Таблиця 1.1 – Визначення типів акторів

|  |  |  |  |
| --- | --- | --- | --- |
| Тип актора | Вага | Примітка | Актори |
| Простий | 1 | Представляє іншу систему, взаємодіючу по зумовленій API (REST, SOAP, dll) | 0 |
| Середній | 2 | Представляє іншу систему, що взаємодіє по протоколу типу TCP / IP | 3 |
| Складний | 3 | Кінцевий користувач, що взаємодіє через GUI або Web page | 2 |
| ∑ UAW = | | | 12 |

Для того визначити місткість варіантів використання по їхньому вагомому коефіцієнту згрупуємо варіанти використання по типу та зробимо обчислення. Результати роботи представлено в таблиці 1.2.

Результати технічних факторів та оцінки зовнішніх факторів знаходяться в таблицях 1.3-1.4 відповідно.

Таблиця 1.2 – Визначення типів варіантів використання за кількістю транзакцій

|  |  |  |  |
| --- | --- | --- | --- |
| Тип варіанту використання | Вага | Кількість транзакцій | Кількість варіантів використання |
| Простий | 5 | До 4 | 2 |
| Середній | 10 | Від 4 до 7 | 3 |
| Складний | 15 | Від 7 | 1 |
| Розрахунок Unadjusted Use Case Waiting (UUCW) | | | 55 |

UCP = UAW + UUCW = 12+ 55 = 67.

Таблиця 1.3 - Технічні фактори

| Фактор | Опис | Вага | Пояснення | Оцінка |
| --- | --- | --- | --- | --- |
| T1 | Розподіленість системи | 2 | Інформує про потребу системи в розподілених обчисленнях | 1 |
| T2 | Час відгуку | 1 | Визначає ефективність системи з точки зору часу відгуку, потоку робіт тощо | 4 |
| T3 | Ефективність кінцевого користувача | 1 | Визначає ефективність користувача з точки зору його (її) сприйняття | 5 |
| T4 | Складність обробки | 1 | Визначає, чи будуть застосовуватись складні алгоритми для обробки даних | 5 |
| T5 | Фокус на повторному використанні коду | 1 | Визначає, чи будуть елементи коду системи використовуватись знову | 2 |
| T6 | Простота інсталяції | 0,5 | Визначає метод інсталяції та простоту інсталяції для кінцевого користувача, чи буде потреба в спеціалісті для інсталяції системи | 3 |
| T7 | Простота використання | 0,5 | Визначає узгодженість інтерфейсу користувача з його потребами | 4 |
| T8 | Портативність | 2 | Визначає, чи має застосування працювати в різних середовищах | 3 |
| T9 | Простота змін | 1 | Визначає, чи будуватиметься система в такий спосіб, щоб спростити її модифікації в майбутньому | 3 |
| T10 | Паралельні обчислення | 1 | Інформує, чи будуть мати в системі місце паралельні обчислення | 0 |
| T11 | Засоби захисту | 1 | Визначає чи потребуватиме система спеціальні засоби захисту даних чи системи | 2 |
| T12 | Доступ до третьої сторони | 1 | Визначає ступень використання системи зовнішніми системами або акторами | 2 |
| T13 | Потреби в спеціальному навчанні | 1 | Визначає, чи потрібно організувати тренінги для користувачів | 1 |
| Technical Complexity Factor (TCF) | | | | 0,955 |

TCF = 0.6+(0.01\*SUM(Ki\*Vi)) = 0,955

Таблиця 1.4 – Оцінка зовнішніх факторів

|  |  |  |  |
| --- | --- | --- | --- |
| Фактор | Описание | Коэф-т(Ki) | Значение (Vi) |
| T1 | Familiarity With The Project Process | 1,5 | 4 |
| T2 | Application Experience | 0,5 | 5 |
| T3 | OO Programming Experience | 1 | 5 |
| T4 | Lead Analyst Capability | 0,5 | 4 |
| T5 | Motivation | 1 | 5 |
| T6 | Stable Requirements | 2 | 5 |
| T7 | Part Time Staff | -1 | 3 |
| T8 | Difficult Programming Language | -1 | 2 |
|  | Environment Factor (EF) |  | 0,635 |

EF = 1.4+(-0.03\*SUM(Ki\*Vi)) = 0,635

Скоректовані UCP значення обчислюються за формулою:

AUCP = UCP \* TCF \* EF = 67 \* 0,955 \* 0,635 = 40.63

Більшість значень T1-T8 перевищують 3, тому одному UCP відповідає 36 робочих годин.

Hours of Effort per Use Case Point = 36 \* 40.63 = 1462,68 годин.

На основі цих даних можна зробити висновок, що для того щоб розробити систему необхідно 1462,68 години.

* 1. Оцінка ризиків

0

3

5

7

 1

4

 6

 8

 2

1.1

1.2

7.1

7.2

5.1

5.2

3.1

3.2

2.2

2.1

4.2

4.1

6.2

6.1

8.2

8.1

Рисунок 3.1 – Діаграма Ішикави для розроблюваної системи

На Рисунку 3.1 показана діаграма Ішикави, завдяки якій ми наочно зображуємо ризики, їхні характеристики, та проблему яку вирішуємо.

Пояснення до діаграми:

0 – Проектування та розробка архітектури ПО.

1 – Взаємодія клієнтської частини та сховища даних.

1.1 – Встановлення та налаштування БД.

1.2 – Підключення модулів взаємодії з БД.

2 – Взаємодія серверної частини та сховища даних.

2.1 – Встановлення та налаштування БД.

2.2 – Підключення модулів взаємодії з БД.

3 – Налаштування клієнт-серверної взаємодії.

3.1 – Встановлення Інтернет з’єднання.

3.2 – Взаємодія модулів передачі даних.

4 – Розрахунок даних.

4.1 – Заповнення та відтворення збережених даних.

4.2 – Розрахунок та перевірка даних.

5 – Оновлення ПЗ.

5.1 – Встановлення Інтернет з’єднання.

5.2 – Перевірка поточної версії ПЗ та завантаження нової версії.

6 – Надання ключа доступу клієнту.

6.1 – Відсилання даних щодо оплати на серверну частину.

6.2 – Перевірка оплати ПЗ адміністратором та відсилання ключа доступу клієнту.

7 – Встановлення ресурсомісткості ПЗ у відповідних «рамках».

7.1 – Перевірка переданих/прийнятих елементів на обмеження.

7.2 – Перевірка ресурсомісткості розраховуємих даних на обмеження.

8 – Робота ПЗ на новому оточенні.

8.1 – Наявність модулів взаємодії з поточним оточенням.

8.2 – Наявність необхідних програм для функціонування ПЗ.

1. Проектування системи
   1. Проектування алгоритмів

Основний алгоритм роботи програми заключається в розрахунку даних.

Тож алгоритм для розрахунку для однієї компанії представлено нижче:

Depo +=( Persent / 100 ) \* Depo – minus + plus - Depo ;

Name, Depo, Persent, Period, Reinvesting, minus, plus

Begin

End

I=0; i < period; i++

Reinvestng(i)

A[b] =( Persent / 100 ) \* Depo – minus + plus - Depo ;

+

-

I=0;i<a;I++

Depo +=a[i]

Для обраних компаній формула для розрахунку наступна:

* 1. Проектування інтерфейсу користувача

При проектуванні інтерфейсу розроблюваної системою були використані принципи єдиного оформлення.

Для ефективного використання простору екрану клієнтського пристрою використаний адаптивний підхід до побудови інтерфейсу (адаптивний або чуйний дизайн). Адаптивний дизайн припускає, що елементи інтерфейсу адаптуються до розміру екрана користувача і можуть виглядати по-різному на дисплеях з різним розширенням.

1. Програмна реалізація

* 1. Програмні компоненти

Серед програмних компонентів першим можна виділити java як основну мову програмування. Java – [об'єктно-орієнтована](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [мова програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), випущена компанією [Sun Microsystems](https://uk.wikipedia.org/wiki/Sun_Microsystems) у [1995](https://uk.wikipedia.org/wiki/1995) році як основний компонент [платформи Java](https://uk.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B0_Java). У офіційній реалізації Java-програми [компілюються](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%96%D0%BB%D1%8F%D1%82%D0%BE%D1%80) у [байт-код](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B9%D1%82-%D0%BA%D0%BE%D0%B4), який при виконанні інтерпретується [віртуальною машиною](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D1%80%D1%82%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD%D0%B0_Java) для конкретної платформи.

Oracle надає компілятор Java та [віртуальну машину Java](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D1%80%D1%82%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD%D0%B0_Java), які задовольняють специфікації [Java Community Process](https://uk.wikipedia.org/wiki/Java_Community_Process), під ліцензією [GNU General Public License](https://uk.wikipedia.org/wiki/GNU_General_Public_License).

Мова значно запозичила синтаксис із [C](https://uk.wikipedia.org/wiki/C_(%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) і [C++](https://uk.wikipedia.org/wiki/C%2B%2B). Зокрема, взято за основу об'єктну модель С++, проте її модифіковано. Усунуто можливість появи деяких конфліктних ситуацій, що могли виникнути через помилки програміста та полегшено сам процес розробки об'єктно-орієнтованих програм. Ряд дій, які в С/C++ повинні здійснювати програмісти, доручено віртуальній машині. Передусім Java розроблялась як платформо-незалежна мова, тому вона має менше низькорівневих можливостей для роботи з апаратним забезпеченням. За необхідності таких дій java дозволяє викликати підпрограми, написані іншими мовами програмування.

Також для розроблення було взято за основу cередовище IDE NetBeans для Java 8. За допомогою редакторів, аналізаторів коду і перетворювачів, ви можете швидко і плавно модернізувати додатки.

Використовуються також javaMail бібліотека для передачі повідомлень на email адреси користувачів.

# Висновки

На практичній роботі я в повній мірі засвоїв роботу на об’єктно-орієнтованій мові програмування java. Спроектував та розробив програмний засіб для автоматизованого розрахунку перспективного прибутку від інвестиційної діяльності. Оновив свої знання із моделювання ПЗ та проектування БД.

# Список використаних джерел

1. Buschmann F. Pattern-Oriented Software Architecture. A System of Patterns. / Buschmann F., Meunier R., Rohnert H., Sommerlad P., Stal M. – John Wiley & Sons Ltd., 1996. – 467 p.
2. Gamma E. Design Patterns: Elements of Reusable Object-Oriented Software / Gamma E., Helm R., Johnson R., Vlissides J. – Addison-Wesley., 1994. – 353 p.
3. Caulfield, C. A Systematic Survey of Games Used for Software Engineering Education/ C. Caulfield, J. Xia, D. Veal, S. Maj // Journal of Modern Applied Science. – 2011. – V. 5(6). – Pp. 28-43.
4. Gondree, M. Valuing Security by Getting [d0x3d!]: Experiences with a Network Security Board Game / M. Gondree, Z. Peterson // The Procs. of the 6th Workshop on Cyber Security Experimentation and Test. – 2013.
5. Fernandes, J.M. PlayScrum - A Card Game to Learn the Scrum Agile Method / J.M. Fernandes , S.M. Sousa // The Procs. of Second International Conference “Games and Virtual Worlds for Serious Applications (VS-GAMES)”. – 2010. – Pp. 52-59.
6. Zapata, C. M. Requirements Game: Teaching Software Project Management / C.M. Zapata, G. Awad-Aubad // CLEI Electronic Journal. – 2007. – V. 10(1). – Pp. 3-14.
7. Denning, T. Control-Alt-Hack: the design and evaluation of a card game for computer security awareness and education / T. Denning, A. Lerner, A. Shostack, T. Kohno // The Procs. of the 2013 ACM SIGSAC conference on Computer & communications security. – Pp. 915-928.

# ДОДАТОК А Програмний код

**SERVER**

**User.java**

package server;

public class user {

private String name,secondname,tel,email;

private boolean online;

protected int item\_number;

user(boolean online,String payed, String name, String secondname, String tel, String email) {

this.online = online;this.name = name;

this.secondname = secondname;this.tel = tel;this.email = email;

}

user() {

this.online=false;

this.name=this.secondname=this.tel=this.email="";

}

Object [] get\_obj(){

return new Object []{this.online,this.name,this.secondname,this.tel,

this.email};

}

protected void set\_online(boolean str){this.online=str;}

protected String get\_name() {return this.name;}

protected String get\_secondname() {return this.secondname;}

protected String get\_email(){return this.email;}

protected boolean get\_online() {return this.online;}

protected void set\_item\_number(int item\_number){this.item\_number=item\_number;}

protected void set\_name(String name){this.name=name;}

protected void set\_secondname(String s){secondname=s;}

protected void set\_login(String username,String secondname,

String telephone,String email) {

this.name=username;

this.secondname=secondname;

this.email=email;

this.tel = telephone;

}

}

**Server.java**

package server;

import java.io.\*;

import java.net.\*;

import java.text.SimpleDateFormat;

import java.util.\*;

public class server {

private int uniqueId;

public ArrayList<ClientThread> al;

private static serverForm sg;

private static SimpleDateFormat sdf;

private static boolean keepGoing;

private static int port;

private static listOfUsers lu;

private ClientThread ct;

private ServerSocket serverSocket;

public server(int port, serverForm sg) {

this.sg = sg;

this.port = port;

sdf = new SimpleDateFormat("HH:mm:ss");

al = new ArrayList<>();

}

void set\_list\_user(listOfUsers lu) {

this.lu = lu;

}

public static class ObjectMessenger{

protected final static int LOGIN=0,QUIT=1,UPDATE=3,ACCESS=4,FILESNAME=5,FILES=6;

protected static Socket socket;

protected ObjectOutputStream oos;

protected ObjectInputStream ois;

public ObjectMessenger(Socket socket){

this.socket = socket;

}

protected void sendObject(String object) throws Exception{

oos = new ObjectOutputStream(socket.getOutputStream());

oos.writeObject(object);

oos.flush();

}

protected String receiveObject() throws Exception{

ois = new ObjectInputStream(socket.getInputStream());

return (String) ois.readObject();

}

protected void typeObject(String str,ClientThread ct) throws IOException, Exception{

String [] mes = str.split(";");

System.out.println("3");

switch(Integer.parseInt(mes[0])){

case LOGIN:{

System.out.println("9");

ct.set\_login(mes[1],mes[2],mes[3],mes[4]); //имя,фамилия,телефон,email,mac

if(lu.check\_user\_in\_list(ct.get\_email())){

sendObject(ACCESS+";"+true);

display("Клиент "+ct.get\_name()+" "+ct.get\_secondname()+" подключился.");

}else{

sendObject(ACCESS+";"+false);

//ct.close();

display("Клиент "+ct.get\_name()+" "+ct.get\_secondname()+" не получил доступа.");}

};break;

case QUIT:{

display("Клиент "+ct.get\_name()+" "+ct.get\_secondname()+" отключился.");

ct.close();

};break;

default:display("Неверное сообщение: "+ str);break;

}

}

}

protected void start() throws Exception {

if(serverSocket != null) stop();

keepGoing = true;

try{

serverSocket = new ServerSocket(port);

while(keepGoing){

display("Ждем клиентов на порту "+port+".");

Socket socket = serverSocket.accept();// accept connection

if(!keepGoing) break;

ClientThread t = new ClientThread(socket);

al.add(t);// save it in the ArrayList

t.start();

}

stop();

}

catch (IOException e) {

display("Сервер остановлен.");

}

}

protected void stop() throws IOException {

keepGoing = false;

serverSocket.close();

for(int i = 0; i < al.size(); ++i) {

ClientThread tc = al.get(i);

tc.close();

}

}

private static void display(String msg) {

sg.appendEvent(sdf.format(new Date()) + " " + msg + "\n");

}

synchronized void remove(int id) {

for(int i = 0; i < al.size(); ++i) {

ct = al.get(i);

if(ct.id == id) {al.remove(i);return;}

}

}

class ClientThread extends Thread {

protected Socket socket;

private int id;

protected user user;

private ObjectMessenger om;

ClientThread(Socket socket) throws Exception {

id = ++uniqueId;

user = new user();

this.socket = socket;

try{

om = new ObjectMessenger(socket);

om.typeObject(om.receiveObject(),this);

}

catch (IOException e) {

display("Ошибка записи/чтения в/из поток(-а): " + e);

}

catch (ClassNotFoundException e) {}

}

@Override

public void run() {

boolean keepGoing = true;

while(keepGoing) {

try {

om.typeObject(om.receiveObject(),this);

}

catch (IOException e) {

user.set\_online(false);

display(get\_name() + " отключился или произошел сбой.");

break;

}

catch (Exception ex) {}

}

try {remove(id);close();} catch (IOException ex) {}

}

String get\_name(){return user.get\_name();}

String get\_secondname(){return user.get\_secondname();}

String get\_email(){return user.get\_email();}

private void close() throws IOException {

if(socket != null) socket.close();

}

private void set\_login(String username,String secondname,

String telephone,String email) {

user.set\_login(username,secondname,telephone,email);

}

}

}

listOfUsers.java

package server;

import java.io.IOException;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import javax.swing.JTable;

import javax.swing.table.DefaultTableModel;

import javax.xml.parsers.ParserConfigurationException;

import org.xml.sax.SAXException;

public class listOfUsers extends javax.swing.JFrame {

private ArrayList<user> buf\_u=new ArrayList<>();

private final database db;

public listOfUsers(database db1) throws ParserConfigurationException, SAXException, IOException, SQLException {

initComponents();

this.db = db1;

getAllUsers();

refresh\_list();

}

final void getAllUsers() throws SQLException{

ResultSet rs = db.getAllUsers();

while(rs.next()){

user c = new user(false,rs.getString("payed"), rs.getString("name"),

rs.getString("secondname"),rs.getString("tel"),rs.getString("email"));

buf\_u.add(c);

}

}

protected final void refresh\_list(){

clear\_table(tcl);

buf\_u.stream().forEach((item) -> {

((DefaultTableModel)tcl.getModel()).addRow(item.get\_obj());

});

}

protected boolean check\_user\_in\_list(String email){

for(int i=0;i<buf\_u.size();i++){

System.out.println("14");

if(buf\_u.get(i).get\_email().equals(email)){

System.out.println("15");

if(buf\_u.get(i).get\_online()) return false;

buf\_u.get(i).set\_online(true);

buf\_u.get(i).set\_item\_number(i);

break;

}

}

System.out.println("12");

return true;

}

protected void clear\_table(JTable table){

DefaultTableModel dtm = (DefaultTableModel) table.getModel();

dtm.setRowCount(0);

}

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jScrollPane1 = new javax.swing.JScrollPane();

tcl = new javax.swing.JTable();

choiseMade = new javax.swing.JComboBox<>();

made = new javax.swing.JButton();

setMinimumSize(new java.awt.Dimension(600, 350));

tcl.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Online", "Name", "Secondname", "Telephone", "Email"

}

));

jScrollPane1.setViewportView(tcl);

choiseMade.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "Нет действий", "Обновить список", "Выслать ключ", "Сохранить", "Выделить всех" }));

choiseMade.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

choiseMadeActionPerformed(evt);

}

});

made.setText("Выбрать");

made.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

madeActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(choiseMade, javax.swing.GroupLayout.PREFERRED\_SIZE, 340, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(made, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT\_SIZE, 569, Short.MAX\_VALUE)

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(choiseMade, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(made))

.addGap(0, 0, 0)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 313, javax.swing.GroupLayout.PREFERRED\_SIZE))

);

}// </editor-fold>

private void madeActionPerformed(java.awt.event.ActionEvent evt) {

switch(choiseMade.getSelectedItem().toString()){

case "Обновить список":{

refresh\_list();

};break;

case "Выслать ключ":{}; break;

case "Сохранить":{};break;

case "Выделить всех":{};break;

default:break;

}

}

private void choiseMadeActionPerformed(java.awt.event.ActionEvent evt) {

}

private javax.swing.JComboBox<String> choiseMade;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JButton made;

private javax.swing.JTable tcl;

}

**Database.java**

package server;

import java.sql.\*;

import java.sql.Connection;

import org.h2.jdbcx.JdbcConnectionPool;

public class database {

Connection conn;

database() throws ClassNotFoundException, SQLException,

InstantiationException, IllegalAccessException {

JdbcConnectionPool cp = JdbcConnectionPool

.create("jdbc:h2:~/myDB", "root", "1111");

conn = cp.getConnection();

}

void closeConnection() throws SQLException{

conn.close();

}

ResultSet getAllUpdates() throws SQLException{

Statement stmt = conn.createStatement();

return stmt.executeQuery("Select \* from updates;");

}

ResultSet getAllUsers() throws SQLException{

Statement stmt = conn.createStatement();

return stmt.executeQuery("Select \* from users;");

}

ResultSet getUserByImail(String email) throws SQLException{

Statement stmt = conn.createStatement();

return stmt.executeQuery("Select \* from users where email='"+email+"';");

}

ResultSet getUserByTel(String tel) throws SQLException{

Statement stmt = conn.createStatement();

return stmt.executeQuery("Select \* from users where tel='"+tel+"';");

}

void insertUser(user u) throws SQLException{

//TODO

Statement stmt = conn.createStatement();

//return stmt.executeQuery("Select \* from users where tel='"++"';");

}

}