Одеський національний політехнічний університет

Інститут комп’ютерних систем

Кафедра системного програмного забезпечення

**Пояснювальна записка**

до дипломної роботи бакалавра

на тему «Клієнт-серверна програма розрахунку перспективного прибутку від інвестиційної діяльності»

Виконав: студент 4 курсу, групи АС-123

напряму підготовки

6.050103 – Програмна інженерія

Буслюк П. С.

Керівник: Паулін О. М.

Рецензент: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Одеса – 2016

Одеський національний політехнічний університет

Інститут комп’ютерних систем

Кафедра системного програмного забезпечення

Освітньо-кваліфікаційний рівень бакалавр

Напрям підготовки 6.050103 – Програмна інженерія

**ЗАТВЕРДЖУЮ**

Завідувач кафедри СПЗ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (В.В. Любченко)

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_\_ року

ЗАВДАННЯ

НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

|  |
| --- |
| Буслюку Петру Сергійовичу |
| (прізвище, ім’я, по-батькові) |

1. Тема роботи: \_Клієнт-серверна програма розрахунку перспективного прибутку від інвестиційної діяльності

Керівник роботи: \_\_Паулін Олег Миколайович, доктор технічних наук, професор\_ (прізвище, ім’я, по-батькові, науковий ступінь, вчене звання)

затверджені наказом ректора ОНПУ від «27» квітня 2016 року № 246-в

2. Строк подання студентом роботи \_\_\_\_\_\_\_\_\_\_\_

3. Вихідні дані до роботи у відповідності до технічного завдання на розробку

4. Зміст розрахунково-пояснювальної записки (перелік всіх питань, які необхідно розробити)

Вступ. Функціональні вимоги до системи. План виконання проекту. Проектування системи. Програмна реалізація. Тестування системи. Висновки. Список використаних джерел.\_Лістинги програми

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

У відповідності зі слайдами електронної презентації

6. Консультанти розділів проекту

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
| завдання видав | завдання прийняв |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

7. Дата видачі завдання \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАЛЕНДАРНИЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| №  з/п | Назва етапів дипломної  Роботи | Строк виконання етапів роботи | Примітка |
| 1 | Специфікація вимог | 04.01.16 – 10.01.16 | Виконано |
| 2 | Планування проекту | 11.01.16 – 22.01.16 | Виконано |
| 3 | Проектування проекту | 22.01.16 – 08.02.16 | Виконано |
| 4 | Реалізація проекту | 08.02.16 – 06.04.16 | Виконано |
| 5 | Тестування програми | 06.04.16 – 25.04.16 | Виконано |
| 6 | Введення в експлуатацію | 25.04.16 – 12.06.16 | Виконано |

Студент: \_\_\_\_\_\_\_\_\_\_\_\_ Буслюк П.С.

(підпис) (прізвище та ініціали)

Керівник роботи: \_\_\_\_\_\_\_\_\_\_\_\_ Паулін О.М.

(підпис ) (прізвище та ініціали)

Реферат

Пояснювальна записка до дипломної роботи: 94 с. (42с. без додатків), 17 рис., 10 табл., 10 джерел, 2 додатки.

Метою роботи є підвищення ефективності обчислення даних перспективного прибутку, щодо інвестованих коштів в різні сфери інвестованої діяльності, спрощення створення звітів щодо інвестованих компаній та створення системи віддаленого збереження та відновлення даних клієнта.

Методи розробки базуються на технології Java, із використанням компонентів Swing для GUI, H2 базами даних які підтримують SQL-подібні запити, Java Mail бібліотеки для інформування клієнтів на email адреси щодо нових цікавих проектів, Itext бібліотеки для створення звітів у pdf-форматі, SoketIO для передачі та відновлення даних та JDOM-бібліотеки для відновлення даних із xml-файлу локально.

Як результат роботи виконана реалізація клієнт-серверної програми для розрахунку перспективного прибутку від інвестиційної діяльності .

Ключові слова: клієнт-серверна GUI програма, JDOM, java xml, sql, h2 java, pdf, java mail, socket io java, itext, swing java .

ABSTRACT

The aim is to improve the efficiency of data calculations prospective profits on invested funds invested in various spheres of activity, simplify reporting on the investees and creating a system of remote conservation and restoration of customer data.

The methods of making are based on the technology Java, using Swing components for GUI, H2 databases that support SQL-like queries, Java Mail library to inform clients on the email address on new projects, Itext library for creating reports in pdf-format, SoketIO for transmission and data recovery and JDOM-library to recover data from xml-file locally.

As a result of the implementation done client-server application to calculate prospective profit from the investment.

Keywords: client-server GUI program, JDOM, java xml, sql, h2 java, pdf, java mail, socket io java, itext, swing java.

Зміст

ВСТУП 7

1 ВИМОГИ ДО СИСТЕМИ 9

1.1 Функціональні вимоги до системи 9

1.1.1 Опис предметної області і аналогів системи 11

1.1.2 Варіанти використання системи 13

1.2 Нефункціональні вимоги до системи 17

2 ПЛАН ВИКОНАННЯ ПРОЕКТУ 19

2.1 Оцінка тривалості розробки 19

2.2 Оцінка ризиків 24

2.3 Розробка плану робіт проекту 27

3 ПРОЕКТУВАННЯ СИСТЕМИ 29

3.1 Концептуальне проектування 29

3.2 Проектування алгоритмів 30

3.3 Проектування інтерфейсу користувача 33

4 ПРОГРАМНА РЕАЛІЗАЦІЯ 34

4.1 Опис програмних технологій 34

4.2 Опис програмних бібліотек 36

4.3 Особливості створення структур даних та програмних модулів 36

5 ТЕСТУВАННЯ СИСТЕМИ 38

ВИСНОВКИ 41

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 42

Додаток А. ІНСТРУКЦІЯ З ВСТАНОВЛЕННЯ ТА ВИКОРИСТАННЯ 43

Додаток Б. ЛІСТИНГ ПРОГРАМИ 48

**Вступ**

Основними проблемами у сучасному світі для інвесторів є пошук нових систем для інвестування та правильно розподілений бюджет (менеджмент коштів), вистроювання планів щодо інвестування та правильне розуміння ризиків щодо інвестованих інструментів.

Загалом, якщо мова іде про інвестування в шахрайські проекти (хайпи/фінансові піраміди) [9], то як ви розумієте повинен бути максимальний прибуток в найкоротший час, а ризики при цьому можуть складати до 100 відсотків від суми інвестування. Це зокрема, як на мій погляд, най ризиковані проекти, однак є багато прикладів, коли люди, які взнали про це першими, стали дуже впливовими.

Якщо розглядається фондова біржа [5], то тут є зокрема декілька факторів щоб залишитись із своїми коштами та примножити їх. Це і математичне очікування, коли у виграшній операції ви маєте потрійну перевагу ніж у збитковій, тож на дистанції ви будете у профіті, і розподілення депозиту балансу щодо вашого інвестиційного портфелю на рівні не більше 10 відсотків від усіх коштів, і максимальне просідання балансу депозиту на рівні в 1 відсоток у збитковій угоді тощо. Однак, можуть бути фактори, які не можна спрогнозувати, наприклад банкротство компанії, неуважність при створенні нового ордеру, навіть різка зміна напряму тренду може призвести до повного банкротства, коли брокер не може закрити ваш ордер по необхідній ціні, і автоматично ваш ордер відхиляється, або ж просто відсутність мережі Інтернет може суттєво наблизити депозит до нуля тощо [4].

Тобто над усіма цими факторами рано чи пізно задумується кожний інвестор, а для цього треба бути готовим. Дана ж дипломна робота розроблена для того, щоб уникнути марної роботи із розрахунку прибутку, який можливо отримає клієнт в разі, коли все буде добре, якщо інвестує свої кошти в будь-який проект, тобто підвищити ефективність розрахунку перспективного прибутку; щоб надати клієнту важливу інформацію щодо нових, мало знайомих але перспективних інвестиційних фондів, а це являється покращенням інформування клієнтів; щоб клієнт ввівши один раз дані міг з легкістю оновити їх, або відтворити заново, тобто автоматизувати процес збереження та відновлення даних із віддаленого джерела; підвищити швидкість створення звітів. Це і являється ціллю даного проекту.

Для реалізації вказаної цілі в роботі було створено програмне середовище, за допомогою якого, навіть без спеціального навчання, будь-яка мало знайома з комп’ютером людина зможе з легкістю обчислити та зберегти дані щодо інвестованих фінансових фондів, відновити дані локально або віддалено та отримати новини не тільки щодо нових пропозицій із інвестування, але й оновити свої знання щодо новин бізнесу, інвестицій, фінансів через систему інформування на email адресу тощо.

Зокрема ця система може застосовуватись лише в окремих колах людей, для яких час – це гроші, а новини – це ідеї для втілення нових можливостей.

Також завдяки цій реалізації люди зможуть вирішити принаймні три найважливіших проблеми сучасності, про які я розповів вище – керувати своїми фінансами, продумувати плани щодо інвестування та отримувати інформацію про нові системи для примноження свого капіталу.

**1 ВИМОГИ ДО СИСТЕМИ**

* 1. **Функціональні вимоги до системи**

У функціональних вимогах можуть зустрічатись не зовсім відомі усім позначення, тож для того, щоб прояснити ситуацію можна навести приклад словника термінів (тезаріусу) для наступних слів:

* Архітектура клієнт-сервер – передбачає взаємодію та обмін даними між вузлами. Вона передбачає такі основні компоненти:

1. набір [серверів](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80), які надають інформацію або інші послуги програмам, які звертаються до них;
2. набір [клієнтів](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), які використовують сервіси, що надаються серверами;
3. [мережа](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D1%83%D0%BD%D1%96%D0%BA%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0), яка забезпечує взаємодію між клієнтами та серверами.

* Авторизація – керування рівнями та засобами [доступу](https://uk.wikipedia.org/w/index.php?title=%D0%94%D0%BE%D1%81%D1%82%D1%83%D0%BF&action=edit&redlink=1) до певного захищеного ресурсу, як в фізичному розумінні (доступ до кімнати готелю за карткою), так і в галузі цифрових технологій (наприклад, автоматизована система контролю доступу) та ресурсів системи залежно від [ідентифікатора](https://uk.wikipedia.org/wiki/%D0%86%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D1%96%D0%BA%D0%B0%D1%82%D0%BE%D1%80) і [пароля](https://uk.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%BE%D0%BB%D1%8C) користувача або надання певних повноважень (особі, програмі) на виконання деяких дій у системі обробки даних.
* База даних ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) database) – це сукупність [даних](https://uk.wikipedia.org/wiki/%D0%94%D0%B0%D0%BD%D1%96_(%D0%BE%D0%B1%D1%87%D0%B8%D1%81%D0%BB%D1%8E%D0%B2%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D1%82%D0%B5%D1%85%D0%BD%D1%96%D0%BA%D0%B0)), організованих відповідно до концепції, яка описує характеристику цих даних і взаємозв'язки між їх елементами; ця сукупність підтримує щонайменше одну з областей застосування. В загальному випадку база даних містить схеми, таблиці, подання, збережені процедури та інші об'єкти. Дані у базі організовують відповідно до моделі організації [даних](https://uk.wikipedia.org/wiki/%D0%94%D0%B0%D0%BD%D1%96). Таким чином, сучасна база даних, крім саме даних, містить їх опис та може містити засоби для їх обробки.
* XML – розширювана мова розмітки ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) Extensible Markup Language) запропонований консорціумом World Wide Web ([W3C](https://uk.wikipedia.org/wiki/W3C)) стандарт побудови [мов розмітки](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D1%80%D0%BE%D0%B7%D0%BC%D1%96%D1%82%D0%BA%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) ієрархічно структурованих [даних](https://uk.wikipedia.org/wiki/%D0%94%D0%B0%D0%BD%D1%96) для обміну між різними [застосунками](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA), зокрема, через [Інтернет](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82). Є спрощеною підмножиною мови розмітки [SGML](https://uk.wikipedia.org/wiki/SGML). XML-документ складається із текстових знаків, і придатний до читання людиною.
* Інтернет (від [англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) Internet), міжмережжя – всесвітня система взаємосполучених [комп'ютерних мереж](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D1%83%D0%BD%D1%96%D0%BA%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0), що базуються на [комплекті Інтернет-протоколів](https://uk.wikipedia.org/wiki/TCP/IP). Інтернет також називають мережею мереж. Інтернет складається з мільйонів локальних і глобальних приватних, публічних, академічних, ділових і урядових мереж, пов'язаних між собою з використанням різноманітних дротових,[оптичних](https://uk.wikipedia.org/wiki/%D0%9E%D0%BF%D1%82%D0%BE%D0%B2%D0%BE%D0%BB%D0%BE%D0%BA%D0%BD%D0%BE) і [бездротових технологій](https://uk.wikipedia.org/wiki/%D0%91%D0%B5%D0%B7%D0%B4%D1%80%D0%BE%D1%82%D0%BE%D0%B2%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0). Інтернет становить фізичну основу для розміщення величезної кількості інформаційних ресурсів і послуг, таких як взаємопов'язані гіпертекстові документи [Всесвітньої павутини](https://uk.wikipedia.org/wiki/%D0%92%D1%81%D0%B5%D1%81%D0%B2%D1%96%D1%82%D0%BD%D1%8F_%D0%BF%D0%B0%D0%B2%D1%83%D1%82%D0%B8%D0%BD%D0%B0) (World Wide Web – WWW) та [електронна пошта](https://uk.wikipedia.org/wiki/%D0%95%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0_%D0%BF%D0%BE%D1%88%D1%82%D0%B0).
* Електронна пошта ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) e-mail, або email, скорочення від electronic mail) – популярний [сервіс](https://uk.wikipedia.org/wiki/%D0%95%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%B9_%D1%81%D0%B5%D1%80%D0%B2%D1%96%D1%81) в [інтернеті](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82), що робить можливим обмін даними будь-якого змісту ([текстові документи](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D0%B8%D0%B9_%D1%84%D0%B0%D0%B9%D0%BB), [аудіо](https://uk.wikipedia.org/wiki/%D0%90%D1%83%D0%B4%D1%96%D0%BE%D1%84%D0%B0%D0%B9%D0%BB)-[відеофайли](https://uk.wikipedia.org/w/index.php?title=%D0%92%D1%96%D0%B4%D0%B5%D0%BE%D1%84%D0%B0%D0%B9%D0%BB&action=edit&redlink=1), [архіви](https://uk.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D1%96%D0%B2_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), [програми](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%B0)).
* Інвестиція, капітальні вкладення (від [лат.](https://uk.wikipedia.org/wiki/%D0%9B%D0%B0%D1%82%D0%B8%D0%BD%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) invest, вкладення коштів) – [господарська операція](https://uk.wikipedia.org/wiki/%D0%93%D0%BE%D1%81%D0%BF%D0%BE%D0%B4%D0%B0%D1%80%D1%81%D1%8C%D0%BA%D0%B0_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D1%8F), яка передбачає придбання основних фондів, нематеріальних активів, корпоративних прав та [цінних паперів](https://uk.wikipedia.org/wiki/%D0%A6%D1%96%D0%BD%D0%BD%D1%96_%D0%BF%D0%B0%D0%BF%D0%B5%D1%80%D0%B8) в обмін на [кошти](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D1%88%D1%82%D0%B8) або [майно](https://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D0%B9%D0%BD%D0%BE). Інвестиції поділяються на капітальні, фінансові та реінвестиції [5].
* Інвестиційний менеджмент – процес управління інвестиційною діяльністю.
* Оцінка інвестицій – [оцінка](https://uk.wikipedia.org/wiki/%D0%9E%D1%86%D1%96%D0%BD%D0%BA%D0%B0) можливих майбутніх [витрат](https://uk.wikipedia.org/wiki/%D0%92%D0%B8%D1%82%D1%80%D0%B0%D1%82%D0%B8) і [доходів](https://uk.wikipedia.org/wiki/%D0%94%D0%BE%D1%85%D0%BE%D0%B4), що можуть виникнути в результаті інвестицій у даний [проект](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B5%D0%BA%D1%82) протягом очікуваного терміну його дії. Оцінка інвестицій включає оцінку ризиків і чутливості проекту, тобто того, в якому ступені можливі помилки в прогнозах можуть уплинути на очікувані результати за даним проекту. Така оцінка допомагає прийняти рішення про те, чи варто вкладати ресурси в даний проект.

Тож, ознайомившись із даними термінами, наведемо приклад багаторівневої ієрархії системи функціональних вимог, зображеної на рис. 1.1 де показано WBS структуру даних функціональних вимог.

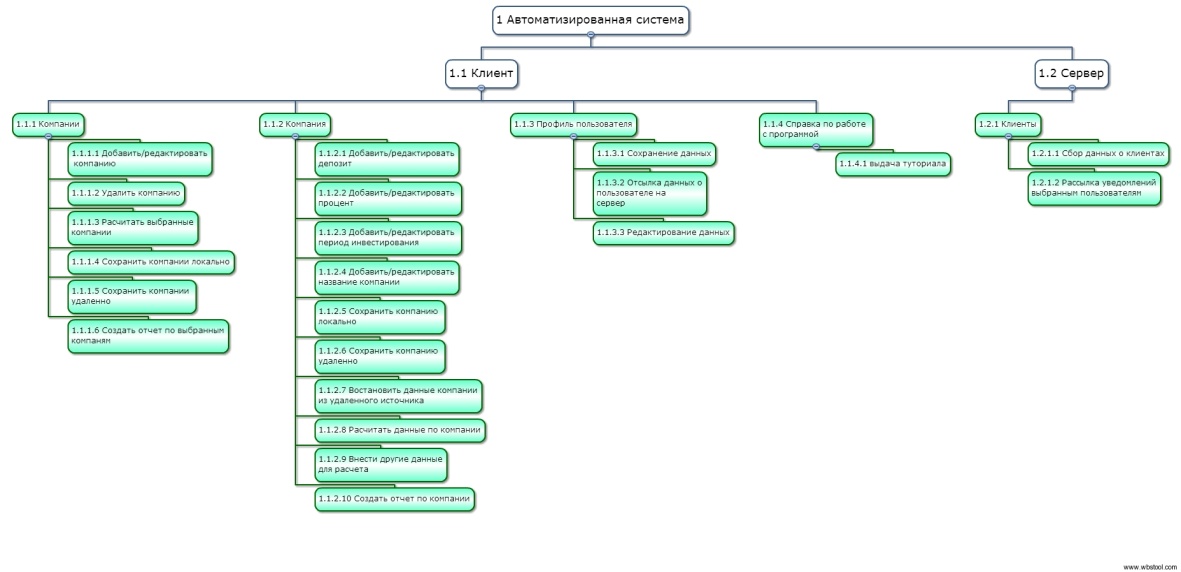


Рисунок 1.1 – Багаторівнева ієрархія функціональних вимог

Можна виключити такі вимоги, адже вони зберігаються у вимогах «Компаній»:

1. 1.1.2.5 – зберегти компанію локально;
2. 1.1.2.6 – зберегти компанію віддалено;
3. 1.1.2.7 – відновити данні з віддаленого джерела.

Визначення критеріїв за типом MoSCoW: усі вимоги мають тип M, однак можна додати ще пару вимог типу W – додати оновлення програми та ввести оплату за ПЗ, де тип М – повинні це зробити, W – не буде достатньо часу на розробку, але в майбутньому можливо буде розроблено.

* + 1. **Опис предметної області і аналогів системи.** Перш ніж приступати до розробки треба проаналізувати вже існуючі продукти на наявність функціональності яка в них присутня, і якщо необхідна функціональність відсутня, то має сенс на розробку власного продукту.

Серед знайдених в просторах Інтернету знайшлось лише декілька прикладів схожого функціоналу. Тож, можна виявити більш-менш підходящі до нашої теми дипломної роботи вже існуючі програмні продукти. Лише одна із наведеного нижче списку система являється для користування на комп’ютері (desktop), інші ж являються онлайн програмами, які розраховують дані на стороні клієнта в браузері.

Існуючі програмні продукти:

1. <http://invite-invest.ru/calculator-investitsii.html> (invest-invest.ru);
2. <http://soft.delovar.info/> (delovar.info);
3. [http://www.denega.ru/portcalc162.zip](http://mmgp.ru/redirect/away.php?url=http%3A%2F%2Fwww.denega.ru%2Fportcalc162.zip) (denega.ru);
4. [http://www.firmasoft.ru/hide/finans/clear.htm (firmasoft.ru](http://www.firmasoft.ru/hide/finans/clear.htm%20-%20firmasoft.ru)).

Можна побачити, що даних програмних продуктів не так вже й багато, сюди можна було також включити Microsoft Excel, але в ньому необхідно знати основи програмування, а це не дуже підходить під сучасні умови, коли більшість людей неспроможні навіть видалити вірус чи опановувати нові, незнайомі для них, технології, втрачаючи на це купу часу та нервів.

Тож, наведемо список проблем які зустрічаються в даних програмних продуктах та проаналізуємо, чи варто проектувати та розробляти наше програмне забезпечення (табл. 1.1).

Таблиця 1.1 - Таблиця рішень проблеми:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Продукти\  Рішення проблеми | Збереження даних віддалено | Розрахунок більше 1 компанії | Можливість довкладення/знімання суми | Відновлення даних | Інформаційна розсилка |
| investinvest.ru | **–** | **–** | + | **–** | **–** |
| delovar.info | **–** | **–** | + | **–** | **–** |
| denega.ru | **–** | **–** | **–** | + | **–** |
| firmasoft.ru | – | – | – | **–** | **–** |
| InvestProfit | + | + | + | + | + |

Як можна побачити, лише в декількох із них присутні лише часткові функції які необхідні для повноцінного функціоналу, із цього можна зробити висновок, що систему необхідно спроектувати та розробити із встановленими для цього функціональними вимогами.

* + 1. **Варіанти використання.** Проектування системи починається із початкового осмислення важливих функціональних елементів, які будуть в подальшому присутні в системі.

Тож, для цього існують так звані Use Case діаграми (діаграми варіантів використання/діаграми прецедентів), за допомогою яких на мові UML (уніфікованій мові моделювання) зображуються актори та взаємозв’язки між прецедентами (основними функціями системи).

Зокрема на діаграмах варіантів використання зображують систему яку розроблюють у вигляді прямокутника, де зверху пишуть назву системи, акторів чи сервісів (у вигляді людини), безпосередньо прецедентів та взаємозв’язків між акторами чи сервісами та прецедентами.

Структуру змодельованої системи можна переглянути нижче на рис. 1.2 – діаграмі прецедентів для системи розрахунку перспективного прибутку.

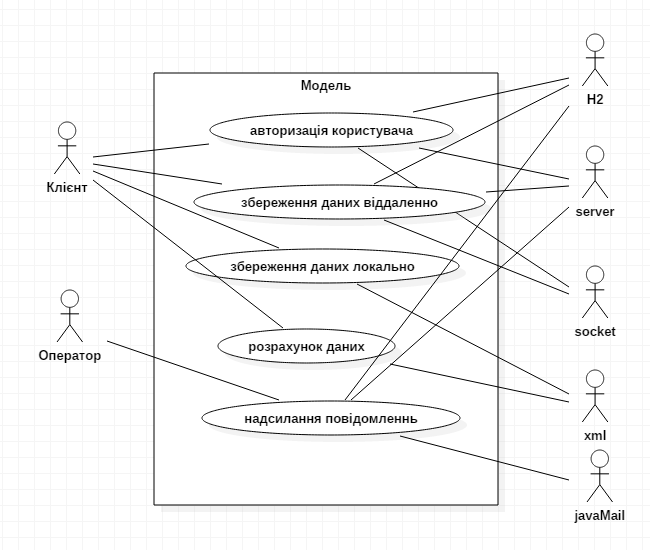


Рисунок 1.2 – Діаграма прецедентів системи

Згідно рис. 1.2 можна виділити такі типи акторів та сервіси в системі:

1. Клієнт – кінцевий користувач;
2. Оператор ­­– кінцевий користувач;
3. SocketIO – система взаємодії по протоколу TCP / IP;
4. H2 – БД, взаємодіюча по протоколу TCP / IP;
5. JavaMail ­– бібліотека, взаємодіюча по протоколу TCP / IP;
6. Xml-сховище локальних даних.

Для того щоб зробити детальніший план для розробки, необхідно описати яка послідовність дій буде в кожному прецеденті, тож наведемо це в табл. 1.2 – із сценаріями використання для кожного із них.

Таблиця 1.2 – Сценарії використання

|  |  |  |
| --- | --- | --- |
| Варіант використання №\_\_ : «Назва прецеденту» | Дії та суб’єкти | Опис дій |
| Варіант використання №1:  «Авторизація користувача» | Передумова | К заповнив усі необхідні поля профілю |
| Післяумова | Сервер дав згоду на користування системою |
| Суб'єкт | Користувач(К), Сервер(С), Система Користувача (СК) |
| Основний потік | 1. К входить до програми та заповнює профіль. СК пропонує збереження даних профілю. 2. СК відправляє дані авторизації на С. 3. С шукає в БД дані К та відсилає відповідь. |

Продовження таблиці 1.2

|  |  |  |
| --- | --- | --- |
| Варіант використання №\_\_ : «Назва прецеденту» | Дії та суб’єкти | Опис дій |
|  | Альтернативний потік | 1а.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. |
| 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) |
| Основний потік | 1. К обирає збереження даних локально. 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. |

Продовження таблиці 1.2

|  |  |  |
| --- | --- | --- |
| Варіант використання №\_\_ : «Назва прецеденту» | Дії та суб’єкти | Опис дій |
|  | Альтернативний потік | 1а.1 К вказав невірні дані. СК пропонує ввести дані ще раз. |
| Варіант використання №4:  «Розрахунок даних» | Передумова | К авторизований у СК; Дані заповнені; Дані вірні; |
| Післяумова | Дані розраховані |
| Суб'єкт | Користувач(К), Система Клієнта(СК) |
| Основний потік | 1. К обирає компанії для розрахунку. СК розраховує дані та видає результат. |
| Альтернативний потік |  |
| Варіант використання №5:  «Надіслання повідомлення» | Передумова | Є доступ до БД та мережі Iнтернет. |
| Післяумова | Повідомлення відіслані вказаним адресатам. |
| Користувачі | Сервер(С), БД, Оператор(О), Клієнт(К) |
| Основний потік | 1. О обирає К для оповіщення, вводить саме повідомлення та назву заголовку. С встановлює email адресу кожного К та надсилає повідомлення. |
| Альтернативний потік | 1а.1 Невірна адреса. С пропускає К та відсилає на нашу емейл адресу звіт із неправильним надходженням повідомлення. |

Таким чином, в табл. 1.2 описані прецеденти системи.

* 1. **Нефункціональні вимоги до системи**

Нефункціональні вимоги характеризують властивості реалізації, які потрібно використовувати при безпосередній реалізації системи. Тож, можна описати спершу зовнішні інтерфейси.

Зовнішні інтерфейси системи: інтерфейс повинен бути інтуїтивно зрозумілим, із однаковим оформленням на усіх формах, тож було розроблено однотипний інтерфейс для усіх форм. Також використовуються компоненти меню та таблиць для надання повноцінної інформації, розбитої на певні блоки із мінімальною затратою місця на формах для кращого зовнішнього вигляду.

Одночасно із цим, були внесені зміни в наступні стилі оформлення: шрифт – Segoe UI, назви кнопок та елементів на формах – російською мовою, задній фон: сірий (RGB(240,240,240)).

За управління відповідають стандартні елементи swing бібліотеки (для графічного зображення елементів на моніторі), яка надається разом із jdk (набором компонентів для розробки). Також налаштовані гарячі клавіші:

1. F1 – довідка;
2. Ctrl+R – формування звіту;
3. Ctrl+A – вибір усіх компаній;
4. Alt+C – розрахунок даних обраних компаній;
5. Ctrl+N – створення нової компанії;
6. Ctrl+D – видалення обраної компанії;
7. Ctrl+Alt+S – збереження компаній віддалено;
8. Ctrl+Alt+R – відновлення даних з серверу;
9. Alt+N – редагування назви компанії;
10. Alt+D – редагування депозиту;
11. Alt+P – редагування відсотку;
12. Alt+I – редагування тривалості інвестування.

Відображення повідомлень відбувається в стандартних MessageBox операційної системи.

При користуванні системою необхідно, щоб версія операційної системи була від Windows – XP SP1 та вище.

Також, для повноцінної роботи програми необхідна обов’язкова наявність JDK версії 8u77 та вище.

Інтерфейси передачі даних між клієнтом та сервером: Wifi або Ethernet.

Зокрема можна виділити основні нефункціональні вимоги, а саме:

Продуктивність:

* максимальний час відгуку програми на запит до 10 секунд;
* максимальна кількість обслуговуваних клієнтів одночасно до 200.

Надійність:

* мінімальний час безвідмовної роботи 10 хвилини;
* також передбачено відновлення даних віддалено після збоїв у системі.

Технології розробки:

* ОС: Windows 7, Windows 10;
* СУБД: H2;
* програмні технології: desktop;
* структури даних: xml, files, db.

Інструменти розробки:

* IDE(платформа для розробки): NetBeans;
* збирач програмних модулів: JDK версія 8u77;
* VCS (система контролю версій): Git;
* бібліотеки модульного тестування: jUnit4;

1. **ПЛАН ВИКОНАННЯ ПРОЕКТУ**
   1. **Оцінка тривалості розробки**

Серед методологій оцінок тривалості проекту є SLOC (кількість строк коду) не підходяща за декількома факторами та FP (функціональні точки), за допомогою яких ми спроектували нашу систему [8].

UCP метод оцінки об’єму виконуваних робіт входить до методології функціональних точок, тож, виділимо наступні показники для оцінки тривалості нашого продукту. В табл. 2.1 приведено визначення типів акторів і підрахована кількісна вага акторів (UAW).

Таблиця 2.1 – Визначення типів акторів

|  |  |  |  |
| --- | --- | --- | --- |
| Тип актора | Вага(*Кi*) | Примітка | Актори(*Vi*) |
| Простий | 1 | Представляє іншу систему, взаємодіючу по зумовленій API (REST, SOAP, dll). | 1 |
| Середній | 2 | Представляє іншу систему, що взаємодіє по протоколу типу TCP / IP. | 3 |
| Складний | 3 | Кінцевий користувач, що взаємодіє через GUI або Webpage | 2 |
| UAW = | | | 13 |

Кількісна вага акторів (UAW) визначається за формулою 2.1:

, (2.1)

де *Кi* – вага актора,

*Vi* – кількість акторів.

Далі необхідно визначити кількісну складність варіантів використання (UUCW) за формулою 2.2 із даних, представлених в табл. 2.2, де визначено вагу акторів та кількість варіантів використання (див. рис. 1.2).

Таблиця 2.2 – Визначення типів варіантів використання за кількістю транзакцій

|  |  |  |  |
| --- | --- | --- | --- |
| Тип варіанту використання | Вага(VAi) | Кількість транзакцій | Кількість варіантів використання(KAi) |
| Простий | 5 | До 4 | 2 |
| Середній | 10 | Від 4 до 7 | 2 |
| Складний | 15 | Від 7 | 1 |
| Розрахунок кількісної складності варіантів використання (UUCW) | | | 45 |

Розрахована UUCW за формулою 2.2 складає 45.

, (2.2)

де KAi – кількість варіантів використання,

VAi – вага кожного прецеденту.

Наступним кроком для визначення UCP значення являється сумування отриманих раніше значень UUCW та UAW. Тож, UCP становить 58 одиниць.

Однак, необхідно встановити ще часову характеристику значень варіантів використання (Hours Of Effort Per Use Case Point). А для цього необхідно визначити спочатку технічні фактори системи. Представимо дані в табл. 2.3, де приведемо технічні фактори системи.

Таблиця 2.3 – Технічні фактори

| Фактор | Опис | Вага(VSi) | Пояснення | Оцінка(KSi) |
| --- | --- | --- | --- | --- |
| T1 | Розподіленість системи | 2 | Інформує про потребу системи в розподілених обчисленнях | 1 |

Продовження таблиці 2.3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Фактор | Опис | Вага(VSi) | Пояснення | Оцінка(KSi) |
| 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 |

Продовження таблиці 2.3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Фактор | Опис | Вага(VSi) | Пояснення | Оцінка(KSi) |
| 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 |
| Технічна факторна складність (TCF) | | | | 0,955 |

Технічна факторна складність являється одним із основних елементів оцінки тривалості проекту, тому вона розраховується за формулою 2.3 і складає 0,955. Це означає, система не дуже легка в розробці.

, (2.3)

де *KSi* – оцінка фактору,

*VSi* – вага фактору.

Після визначення технічних факторів обчислимо зовнішні фактори за формулою 2.4, дані для якої представлені в табл. 2.4.

Таблиця 2.4 – Оцінка зовнішніх факторів

|  |  |  |  |
| --- | --- | --- | --- |
| Фактор | Опис | Коефіцієнт(KEi) | Значення (VEi) |
| T1 | Знайомство з процесом  Розробки | 1,5 | 4 |
| T2 | Досвід подібних проектів | 0,5 | 5 |
| T3 | Досвід об’єктно-орієнтованої розробки | 1 | 5 |
| T4 | Досвідченість провідного  Аналітика | 0,5 | 4 |
| T5 | Мотивація | 1 | 5 |
| T6 | Стабільність вимог | 2 | 5 |
| T7 | Часткова зайнятість  Працівників | -1 | 3 |
| T8 | Складність мови програмування | -1 | 2 |
| Зовнішній фактор (EF) | | | 0,635 |

Сама ж формула для розрахунку зовнішніх факторів представлена нижче:

, (2.4)

де *KEi* – оцінка зовнішнього фактору,

*VEi* – вага зовнішнього фактору.

Зовнішні фактори показують, наскільки сприятливі умови для розробки системи, в нашому випадку зовнішній фактор EF складає 0,635.

Далі необхідно визначити скоректовану UCP (AUCP), для цього необхідно перемножити між собою раніше знайдені значення UCP, TCF та EF. Тож, AUCP, яка показує кількісну вагу із урахуванням різних факторів складає 35,17.

Зараз визначимо загальну тривалість проекту за HOEPUCP, спершу встановивши, що більшість значень T1-T8 перевищують 3, тому одному АUCP відповідає 35 робочих годин, тож помноживши раніше знайдене значення скоректованої UCP на кількість робочих годин ми отримаємо 1230,95 робочих годин.

Однак, для більш наглядної оцінки тривалості операцій можна скористатись діаграмою Ганта, розробленою Генрі Л. Гантом в 1990 році, яка ілюструє план/графік з тривалості будь-якого проекту. Тривалість проекту зображено на рис. 2.1.

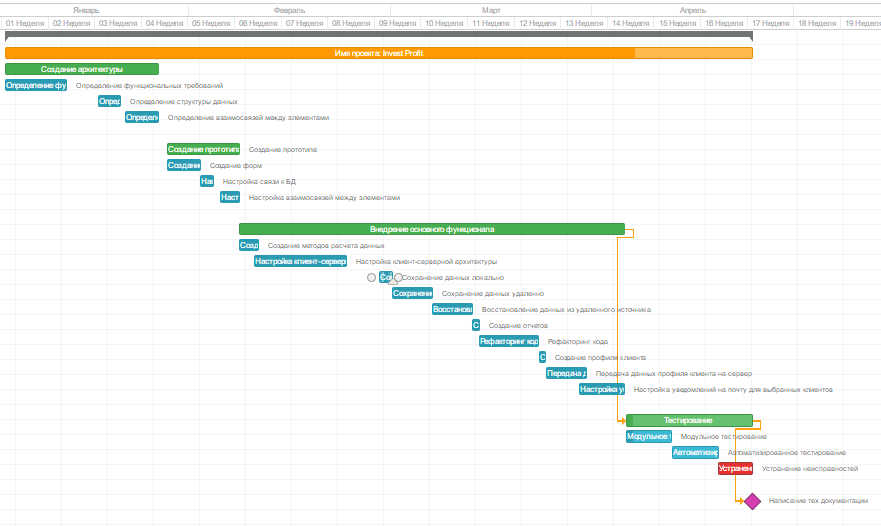


Рисунок 2.1 – Діаграма Ганта

Із рис.2.1 випливає, що на розробку та впровадження системи необхідно приблизно 16 тижнів роботи по 8-10 годин на добу, що складає приблизно 807 годин роботи без урахування написання технічної документації. Із урахуванням документації до проекту знадобиться приблизно 830 годин.

* 1. **Оцінка ризиків**

Серед ризиків системи можна виділити наступні, представлені в табл. 2.5.

Таблиця 2.5 – Ризики проекту

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Ризик | Ймовірність | Вплив | Ранг |
| 1 | Нестача часу на розробку та повне тестування системи | 3 | 3 | 9 |
| 2 | Недостатня кваліфікація розробки даною мовою програмування | 2 | 3 | 6 |
| 3 | Неможливість реалізації повністю взаємодіючої між усіма елементами системи | 2 | 3 | 6 |
| 4 | Невірне спроектована система | 2 | 3 | 6 |
| 5 | Невірно поставлена початкова задача | 2 | 2 | 4 |
| 6 | Довгий час на реалізацію декотрої задачі та під задачі | 1 | 3 | 3 |
| 7 | Невірно обрана бібліотека розробки для задачі або під задачі | 1 | 3 | 3 |
| 8 | Відсутність певних елементів для розробки | 1 | 2 | 2 |
| 9 | Виявлення повного аналогу системи | 1 | 2 | 2 |
| 10 | Інші сторонні фактори | 2 | 1 | 2 |

Серед них, як правило, можна визначити декілька суттєвих, зокрема це номери 1-5 включно, тож для цих ризиків складемо картки опису та наведемо їх в табл. 2.6.

Таблиця 2.6 – Картки описання ризиків системи

|  |  |
| --- | --- |
| Номер: R-1 | Категорія: Організаційна |
| Причина: Нестача часу на розробку та повне тестування | Умови: терміни суттєво занижені із очікуваними для розробки |
| Наслідки: Неспроможність реалізувати необхідні функціональні блоки та реалізувати усю систему вцілому | Вплив: підвищення строків для реалізації, або ж реалізація неповної функціональності |

Продовження таблиці 2.6

|  |  |
| --- | --- |
|  |  |
| Ймовірність: 3 | Ступінь впливу: 3 |
| Близькість: Дуже скоро | Ранг: 9 |
| Стратегія: | |
| Номер: R-2 | Категорія: Організаційна |
| Причина: Недостатня кваліфікація розробки даною мовою програмування | Умови: Необізнаність даної мови програмування, недавня її реалізація |
| Наслідки: Необхідність писати власні бібліотеки тощо | Вплив: збільшення строків на реалізацію |
| Ймовірність: 2 | Ступінь впливу: 3 |
| Близькість: Скоро | Ранг: 6 |
| Стратегія: Відмова від мови програмування, пошук аналогів мови чи ознайомлення із особливостями мови на етапі розробки | |
| Номер: R-3 | Категорія: Технічна |
| Причина: Неможливість реалізації повністю взаємодіючої між усіма елементами системи | Умови: Взаємовязок програмних компонентів не функціонує |
| Наслідки: Неможливо впровадити програмний комплект до релізу | Вплив: на роботу програми |
| Ймовірність: 2 | Ступінь впливу: 3 |
| Близькість: Дуже скоро | Ранг: 6 |
| Стратегія: написання програмного комплексу за загальними паттернами, із використанням знанням усього функціоналу | |

Продовження таблиці 2.6

|  |  |
| --- | --- |
| Номер: R-4 | Категорія: Організаційна |
| Причина: Невірне спроектована система | Умови: із самого початку дана невірна специфікація на розробку |
| Наслідки: Неможливість повноцінної роботи ПЗ | Вплив: вірне функціонування програми неможливе |
| Ймовірність: 2 | Ступінь впливу: 3 |
| Близькість: Дуже скоро | Ранг: 6 |
| Стратегія: Добре спроектувати програмний комплекс, вточнити усі деталі заздалегідь | |
| Номер: R-5 | Категорія: Організаційна |
| Причина: Невірно поставлена початкова задача | Умови: Дані для задачі були невірно оформлені у спецификації |
| Наслідки: Функціонування певних модулів порушена | Вплив: вірна робота системи |
| Ймовірність: 2 | Ступінь впливу: 2 |
| Близькість: Скоро | Ранг: 4 |
| Стратегія: Вточнити усі деталі на етапі проектування системи | |

Тож, із цих 5-ти ризиків, загалом, при розробці ПЗ, дуже сильно, як на мій погляд, будуть впливати саме брак часу на реалізацію та повна взаємодія елементів системи.

* 1. **Розробка плану робіт проекту**

Ієрархічна структура робіт (ІСР) (англ. Work Breakdown Structure, WBS; іноді Структура декомпозиції робіт, СДР) – це ієрархічне розбиття всієї роботи, яку необхідно виконати для досягнення цілей проекту, на більш дрібні операції і дії до такого рівня, на якому способи виконання цих дій цілком зрозумілі і відповідні роботи можуть бути оцінені і сплановані. Вона включає також визначення проміжних результатів всіх складових структури робіт.

Структура декомпозицій робіт представлена на рис 2.2.

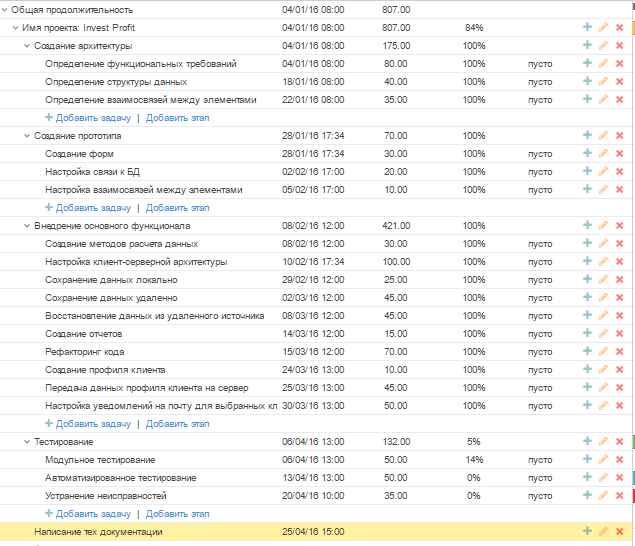


Рисунок 2.2 – WBS структура проекту

Тож, зважаючи на те, що було описано раніше, про тривалість проекту, загальна тривалість проекту в межах допустимих норм, і найбільш довші етапи – це розробка безпосередньо самого функціоналу, розроблення архітектури та тестування.

1. **ПРОЕКТУВАННЯ СИСТЕМИ**
   1. **Концептуальне проектування**

В концептуальному проектуванні безпосередньо необхідно розробити логіку системи, необхідно продумати усі елементи системи та взаємозв’язок між ними[7].

Ґрунтуючись на досвіді, можу лише сказати, що розробляти правильну логіку може лише людина, яка має певний досвід у створенні подібних систем, включно із особливостями мови програмування та інших факторів, які існують.

Загалом, щоб показати логіку системи використовують UML Діаграми концептуальних класів, наприклад для нашої системи ця діаграма зображена на рис. 3.1.

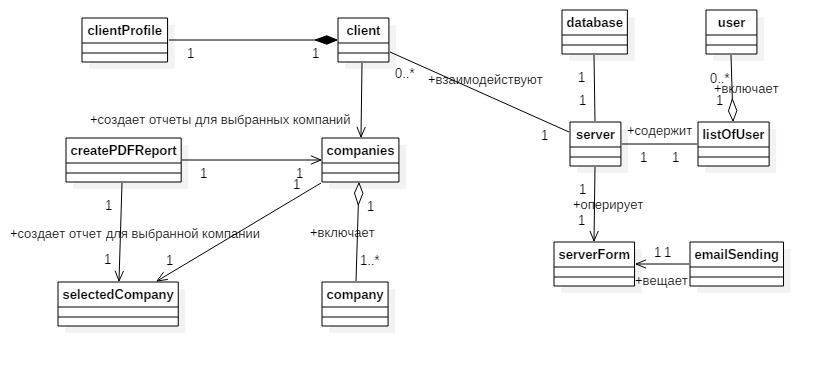


Рисунок 3.1 – Діаграма концептуальних класів

Із рис. 3.1 можна явно виявити, що структура розбивається на 2 пакети – клієнтську та серверну частину, в якій присутні необхідні класи як для однієї так і для іншої частини.

Такі важливі елементи як сервер чи клієнт будуть відповідати за логіку клієнт-серверної архітектури, а якщо мова іде про розрахунок даних для кожної компанії дані будуть розраховуватись на клієнті у класі «Company»

* 1. **Проектування алгоритмів**

Проектування алгоритмів роботи програми починається з того, що визначаються головні методи та атрибути класів із концептуального проектування. Для того, щоб зобразити нагляд ніше взаємодію цих методів та показати які атрибути є в класах існує діаграма програмних класів. Для нашої системи вона зображена на рис. 3.2.

## C:\Diplom\report\диаграма програмных класов.jpg

Рисунок 3.2 – Діаграма програмних класів

Тож, із структурою проекту ми визначились, можна переходити до другого етапу проектування алгоритмів[3]. На прикладі двох найважливіших блок-схем[6] можна зобразити загальний тип виконуваних операцій в системі.

Перша блок-схема алгоритму розрахунку даних зображена на рис. 3.3.



Рисунок 3.3 – Схема алгоритму розрахунку даних

Друга блок-схема характеризує правильність введення даних користувача. Ці дані в майбутньому будуть використовуватись для інформування і підтримання зв’язку із користувачем за допомогою email рис. 3.4.



Рисунок 3.4 – Схема алгоритму перевірки валідності даних користувача

Для того, щоб охарактеризувати правильність роботи алгоритму з точки зору ефективності необхідно оцінити складність даних алгоритмів[10].

Оцінка може бути оцінена як часом (часова складність) так і об’ємом яка задіяна через кількість операцій (ємкісна складність).

Оцінка складності кожного елементарного блоку для різних блоків характеризується такими величинами:

* оператори введення/виведення , проста умова – О (1);
* складна умова – O (m), m – кількість полів.

Оцінка алгоритмічної складності циклів при лінійному часі – О (n), де n – кількість операторів .

Загальна оцінка алгоритмічної складності по першому та другому алгоритму буде квадратичною, так як в функціях є по одному циклу, по якому розраховуються певні оператори, після виконання необхідних умов.

Оцінка алгоритмічної складності доступу до даних:

В даних алгоритмах не використовуються зчитування даних із xml-файлів та БД, бо це зроблено раніше і дані вже завантажені в оперативну пам'ять комп’ютера.

* 1. **Проектування інтерфейсу користувача**

Загалом, інтерфейс користувача був описаний при розробці не функціональних вимог в розділі 1.2. Там приведені приклади, яким саме чином будуть викликатись основні елементи та загальна структура інтерфейсів.

Також, в програмі використовується золотий перетин на усіх формах для поліпшення сприймання користувачем даних.

1. **ПРОГРАМНА РЕАЛІЗАЦІЯ**
   1. **Опис програмних технологій**

Опис програмних технологій включає в себе опис усіх ПЗ, які використовувались або потрібні для програмної реалізації проекту[1].

Тож, зокрема, використовувались такі програмні технології:

NetBeans IDE – [вільне](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%BB%D1%8C%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) [інтегроване середовище розробки](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D1%81%D0%B5%D1%80%D0%B5%D0%B4%D0%BE%D0%B2%D0%B8%D1%89%D0%B5_%D1%80%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B8) (IDE) для [мов 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) [Java](https://uk.wikipedia.org/wiki/Java), [JavaFX](https://uk.wikipedia.org/wiki/JavaFX), [C](https://uk.wikipedia.org/wiki/%D0%A1_(%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))/[C++](https://uk.wikipedia.org/wiki/C%2B%2B), [PHP](https://uk.wikipedia.org/wiki/PHP), [JavaScript](https://uk.wikipedia.org/wiki/JavaScript), [HTML5](https://uk.wikipedia.org/wiki/HTML5), [Python](https://uk.wikipedia.org/wiki/Python), [Groovy](https://uk.wikipedia.org/wiki/Groovy). Середовище може бути встановлене і для підтримки окремих мов, і у повній конфігурації. Середовище розробки NetBeans за замовчуванням підтримує розробку для платформ [J2SE](https://uk.wikipedia.org/w/index.php?title=J2SE&action=edit&redlink=1) і [J2EE](https://uk.wikipedia.org/wiki/J2EE).

Поширюється у open source code під ліцензіями [GPLv2](https://uk.wikipedia.org/wiki/GPLv2) і [CDDL](https://uk.wikipedia.org/w/index.php?title=CDDL&action=edit&redlink=1). Проект NetBeans IDE підтримувався і спонсорувався фірмою [Sun Microsystems](https://uk.wikipedia.org/wiki/Sun_Microsystems) і після придбання Sun – [Oracle](https://uk.wikipedia.org/wiki/Oracle_Corporation), проте розробка NetBeans ведеться незалежно співтовариством розробників (NetBeans Community) і компанією NetBeans.Org.

NetBeans IDE доступна для платформ [Microsoft Windows](https://uk.wikipedia.org/wiki/Microsoft_Windows), [GNU](https://uk.wikipedia.org/wiki/GNU)/[Linux](https://uk.wikipedia.org/wiki/Linux), [FreeBSD](https://uk.wikipedia.org/wiki/FreeBSD) і [Solaris](https://uk.wikipedia.org/wiki/Solaris_(%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0)). Для інших платформ доступна можливість зібрати NetBeans самостійно із open source code.

За якістю і можливостям останні версії NetBeans IDE змагається з найкращим інтегрованими середовищами розробки для мови Java, підтримуючи [рефакторинг](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%BD%D0%B3), [профілювання](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%84%D1%96%D0%BB%D1%8E%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_(%D0%BA%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D1%96_%D0%BD%D0%B0%D1%83%D0%BA%D0%B8)), виділення синтаксичних конструкцій кольором, автодоповнення мовних конструкцій на льоту, шаблони коду тощо.

H2 – відкрита кросплатформна СУБД, повністю написана на мові Java.

Незважаючи на малий розмір (трохи більше 1 МБ) H2 підтримує такі можливості:

* два режиму роботи (клієнт-сервер, вбудований);
* два режиму зберігання даних (файлова система, пам'ять);
* підтримка планів виконання запитів;
* підтримка кластеризації і реплікації;
* шифрування даних;
* зовнішні (пов'язані) таблиці;
* драйвер odbc;
* повнотекстовий пошук;
* визначення доменів;
* мультіверсійний конкурентний доступ;
* підтримка послідовностей;
* підтримка ключових слів limit і offset в запитах;
* тимчасові таблиці;
* обчислювані стовпці;
* призначені для користувача агрегатні функції;
* призначені для користувача процедури стиснення clob / blob об'єктів
* робота з csv файлами на читання і запис;
* браузерна консоль управління;
* запуск як сервіс windows.

Що ще не реалізовано:

* віконні функції (реалізована тільки функція row\_number() та over ());
* багатопоточна обробка запитів;
* повна підтримка стандарту sql 2003.

Застосовується H2, наприклад, в проекті Grails як базова СУБД для розробки.

Java Сокети ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) Socket – заглиблення, гніздо, роз’їм) – назва [програмного інтерфейсу](https://uk.wikipedia.org/wiki/API) для забезпечення обміну даними між [процесами](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)). Процеси при такому обміні можуть виконуватися як на одній ЕОМ, так і на різних ЕОМ, пов'язаних між собою [мережею](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0). Сокет – абстрактний об'єкт, що представляє кінцеву точку з'єднання.

Загалом це і є ті програмні технології, які використовувались при розробці даного програмного продукту.

* 1. **Опис програмних бібліотек**

JavaMail – це Java API, призначене для отримання і відправки електронної пошти з використанням протоколів SMTP, POP3 та IMAP. JavaMail є частиною платформи Java EE, але також доступний в якості додаткового пакета для використання в додатках Java SE.

Існує також альтернативна реалізація JavaMail з відкритим вихідним кодом – GNU JavaMail – яка реалізує тільки специфікацію JavaMail версії 1.3; це єдина вільна реалізація, що підтримує протокол NNTP, дозволяючи читати і відсилати статті в новинні групи.

До складу JavaMail не входить поштовий сервер, проте для подібних завдань можуть використовуватися вільні Apache James і Java Email Server (POP3 і SMTP), або бібліотека SubEthaSMTP для створення SMTP сервера.

JDOM це вільна Java-реалізація DOM для XML, створена з урахуванням особливостей мови і платформи Java. JDOM інтегрується з Document Object Model (DOM) і Simple API for XML (SAX), підтримує XPath і XSLT. У JDOM використовуються зовнішні парсери для генерації документів.

**4.3 Особливості створення структур даних та програмних модулів**

В основі створення програмних модулів лежить ООП підхід. Java відноситься до мов об'єктно-орієнтованого типу, тому, повністю підходить для написання програмних модулів.

Об'єктно-орієнтоване програмування (ООП) – одна з [парадигм](https://uk.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%B8%D0%B3%D0%BC%D0%B8_%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) [програмування](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), яка розглядає програму як множину «об'єктів», що взаємодіють між собою. Основу ООП складають три основні концепції: [інкапсуляція](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D0%BA%D0%B0%D0%BF%D1%81%D1%83%D0%BB%D1%8F%D1%86%D1%96%D1%8F), [успадкування](https://uk.wikipedia.org/wiki/%D0%A3%D1%81%D0%BF%D0%B0%D0%B4%D0%BA%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_(%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)) та [поліморфізм](https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D1%96%D0%BC%D0%BE%D1%80%D1%84%D1%96%D0%B7%D0%BC_(%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)). Одною з переваг ООП є краща [модульність](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D1%8C_(%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)) програмного забезпечення (тисячу функцій процедурної мови, в ООП можна замінити кількома десятками класів із своїми методами).

Для збереження і відновлення даних на стороні клієнту використовуються xml-файли.

Структура даних по компаніям (для розрахунку даних) повинна бути така:

<companies>

<company>

<name>Назва компанії</name>

<depo>Початковий депозит</depo>

<persent>Середній відсоток інвестування</persent>

<period>Період інвестування</period>

<intervals>

<i>

<plus>Сумма довкладень за i-й період</plus>

<minus>Сумма знімань коштів за i-й період </minus>

<reinvesting>Перенесення поточної сумми у наступний період інвестування</reinvesting>

</i>

</intervals>

</company>

</companies>

Усі маніпуляції виконуються із даним файлом через JDOM парсер.

Дані на стороні серверу зберігаються в БД. Доступ для елементів комірок доступний через активне підключення до БД із виконанням sql-подібних запитів.

Приклад вибірки даних усіх користувачів із БД:

ResultSet result = st.executeQuery("SELECT \* FROM USER");  
while(result.next()){  
 // Виконуються необхідні маніпуляції

}

1. **ТЕСТУВАННЯ СИСТЕМИ**

Для тестування існують методи чорного, білого та сірого ящика. Метод чорного ящика зображений на табл. 5.1, де представлено еквівалентне розбиття.

Таблиця 5.1 - Еквівалентне розбиття:

|  |  |  |
| --- | --- | --- |
| Вхідна умова | Правильний клас еквівалентності | Неправильний клас еквівалентності |
| Depo = 100; persent = 7; period = 10; reinvesting = true; | 191,72 – вірне округлення | 191,7151 – невірні дані без округлення чисел |
| Depo = 100; persent = 7; period = 10; reinvesting = true; plus[0]=100 | 420,97 – плюс додається до 0-го періоду | 380,56 – коли переноситься на наступну дату при реінвестуванні |
| Depo = 100; persent = 7; period = 10; reinvesting = true; plus[0]=100; minus[0]=100; | 191,72 – вірні дані зняття та до вкладення сум | 208,59 – невірна сума, бо дані до вкладені на початок, а зняті в кінці місяця, а потім сума перенесена на наступний період |

Покриття умов:

Для цього методу протестуємо валідність email користувача для того, щоб він коректно заповнив дані, бо якщо email одразу некоректний, користувач не зможе бути інформованим задля надання йому доступу до нових цікавих пропозицій.

Тож, в нас є наступний код для тестування, а саме його невелика частина із JUnit тесту[2] для визначення валідності email адреси:

@Test

public void testCheckEmail() {

System.out.println("checkEmail");

String em = "diploma@buslyuk.petr";

clientProfile instance = new clientProfile();

boolean expResult = true;

boolean result = instance.checkEmail(em);

assertEquals(expResult, result);

}

Змінна em в даному випадку слугує для вхідних даних.

При даній email адресі «[diploma@buslyuk.petr](mailto:diploma@buslyuk.petr)» усе вірно, і задовольняє паттерну тестування, тож маємо результат зображений на рис. 5.1 .

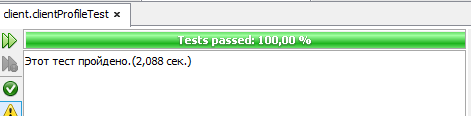


Рис. 5.1 – Пройдений вірно тест

Задовільні тести також показують наступні адреси: «petr@yandex.ru», «petr-27@yandex.com», «petr.27@yandex.com», «petr111@buslyuk.com», «petr.100@buslyuk.com.ua», «petr@gmail.com.com», «petr+27@gmail.com», «petr-27@yandex-test.com».

Інші тести, як наприклад «petr», «petr@.buslyuk.ua», «petrbuslyuk@.com», «petr .buslyuk.com» і т.д. мають невірний формат адреси, тож результат буде таким (див. рис. 5.2).

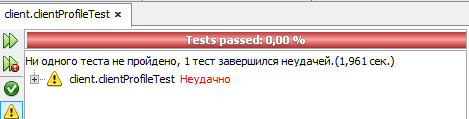


Рисунок 5.2 – Невірно пройдений JUnit тест

Для тестування функціональних частин зобразимо результати на табл. 5.2.

Таблиця 5.2 - Функціональне тестування

|  |  |  |  |
| --- | --- | --- | --- |
| № прецеденту | Дія | Очікуваний результат | Результат тестування успішний(passed)/ невдачний(failed) |
| 1 | Сервер не знайшов клієнта в БД. | Сервер створив нового клієнта та надіслав доступ. | Passed |
| 2 | Клієнт обирає збереження даних компаній. | Система Клієнта перевіряє дані та зберігає їх у XML файл | Passed |
| 3 | Сервер отримує дані та зберігає в БД. | Сервер зберіг дані. | Passed |
| 3 | Вказано невірні дані. | Система Клієнта пропонує ввести дані ще раз. | Passed |
| 4 | Розрахунок обраних компаній. | Система Клієнта розраховує дані та видає результат. | Passed |
| 5 | Оповіщення клієнтів на обрані емейл адреси. | Клієнти оповіщені. | Passed |

Тож, система, загалом була протестована із результатами наведеними вище. При тестуванні декількох частин використовувалось JUnit тестування.

**ВИСНОВКИ**

Добре спроектована система в повній мірі допомагає розробити програмний продукт, скорочує час на усунення недоліків, які виникали би при розробці відокремлених програмних комплексів та допомагає добре зрозуміти вимоги, котрі необхідні для повного функціонування програмних засобів.

Саме завдяки добре спроектованій системі вдалось досягти підвищення ефективності в обчисленні даних, націлених на розрахунок перспективного прибутку від інвестованих коштів у різні сфери інвестиційної діяльності, при цьому підвищена швидкість створення звітів, покращена система інформування клієнтів шляхом відправлення новин на email адреси користувачів та автоматизована система віддаленого збереження/відновлення даних клієнтів.

Труднощі, які виникали при розробці даного програмного продукту були подолані у найкоротші терміни, тож функціональні вимоги для даної версії програмного забезпечення було виконано у повному обсязі. Функціональні вимоги з оновлення ПЗ та внесення благодійних коштів, можливо, увійдуть до подальших програмних реалізацій, бо вони не були в початковій специфікації.

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. Java. Новітні технології [Текст] / Е.О. Шавермін, О.О. Оплюй, В.С. Нагнибіда – М. : Космос, 2015. – 378 с.
2. Peterson, Р.М. JUnit4 interactive testing [Текст] / P.M. Peterson. – ST. : NewDigest, 2016. – 179 с.
3. Алгоритми [Текст] / В.П. Павлов, А.У. Арчібальд – М. : Роджер, 2015. – 180 с.
4. Лабутенко, А.В. Як стати успішним. Інвестиції [Текст] / А.В. Лабутенко. – М. : Консорціум, 2016. – 56 с.
5. Малайченко, Г.І. Інвестування в мережі Інтернет [Текст] / Г.І. Малайченко. – М. : Зірка, 2016. – 130 с.
6. Основи побудови блок-схем [Текст] / Л.Л. Саммерс, Д.Е. Вілямс – М. : ВиджестДайджІнтернешнл, 2015. – 147 с.
7. Паттерни Java [Текст] / Л.Ф. Махновська, Г.Є. Шапля, О. С. Мамцін – М. : Перемога, 2016. – 160 с.
8. Планування проекту ч.3 [Текст] / Д.Д. Локк, C.K. Інджел – М. : Hippo, 2015. – 270 с.
9. Поняківскій, П.С. Перспективи ринку форекс [Текст] / П.С. Поняківскій. – М. : Зірка, 2016. – 256 с.
10. Розробка сучасного ПЗ [Текст] / К.А. Висоцький, М.В. Викон – М. : Космос, 2016. – 279 с.

**Додаток А. ІНСТРУКЦІЯ З ВСТАНОВЛЕННЯ ТА ВИКОРИСТАННЯ**

Для роботи з програмою перед її інсталяцією необхідно встановити:

1) Jdk 7 версії або вище;

2) БД H2 та зробити бекап із файлу дампу, котрий знаходиться на github репозиторії – для серверної частини;

3) запустити jar-файл із програмою.

Початкова загрузка програми відбувається із того, що клієнту пропонується ввести дані щодо його подальшої роботи, а потім зберегти дані локально. Профіль клієнта знаходиться в пункті меню «Профіль». Після чого з’являється форма, зображена на рис. А.1.

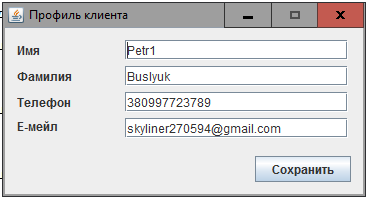


Рисунок А.1 – Форма клієнта «Профіль клієнта»

Клієнт повинен заповнити дані імені, фамілії, телефону та емейлу для інформування клієнта щодо інвестиційних новин серед нещодавно створених, або свої інвестиційних продуктів.

Наступним кроком необхідно додати нову компанію для розрахунку даних. Для цього обирається розділ меню «Компанії->Додати компанію», або натискається комбінацію клавіш Ctrl+N.

Далі необхідно ввести назву компанії, початковий депозит, середній відсоток та продовжність інвестування. Після цього створена компанія з’явиться в таблиці. На рис. А.2 зображено створені компанії, а виділеним рядком – щойно створена.

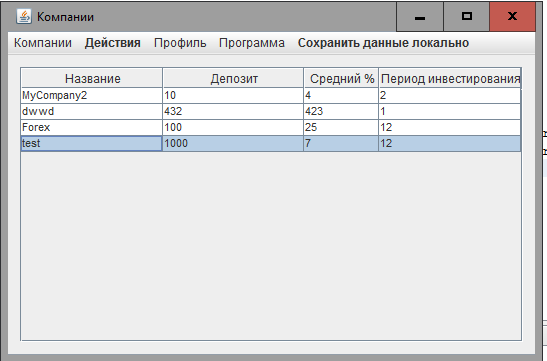


Рисунок А.2 – Форма «Компанії»

Перейдемо до щойно створеної компанії двійним натисненням лівої кнопки миші на неї.

Висвітиться вікно, показане на рис. А.3.

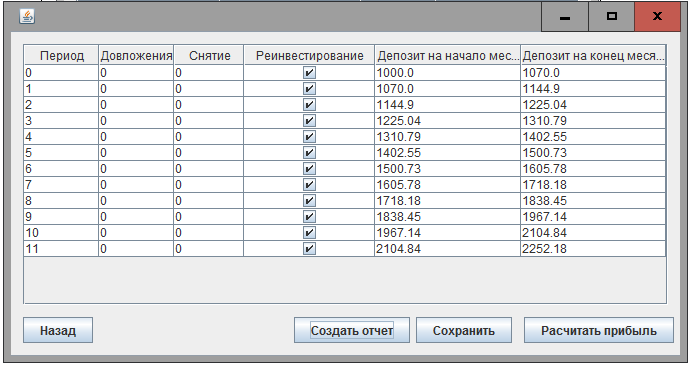


Рисунок А.3 – Форма обраної компанії

Тут ми можемо зберегти дані, створити звіт та розрахувати дані.

Розглянемо створення звіту. Звіт можна створити 2 шляхами – через вибір декількох компаній, і безпосередньо в обраній в 1 екземлярі компанії.

Для створення звіту необхідно буде ввести назву звіту, для того, щоб ви змогли в майбутньому його відшукати на комп’ютері. За умовчанням він зберігається в папці з програмою. Автоматично він зберігається в pdf форматі – рис. А.4.



Рисунок А.4 – Приклад тестового звіту

Приклад звіту всередині можна проглянути на рис. А.5.

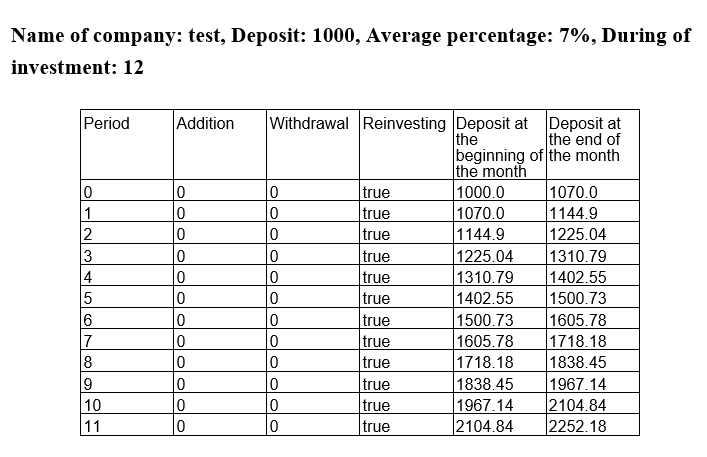


Рисунок А.5 – Приклад звіту в pdf-файлу

Інструкція оператора (сервер):

Коли оператор входить до серверної частини програми, то він попадає на головну форму, зображену на рис. А.6.

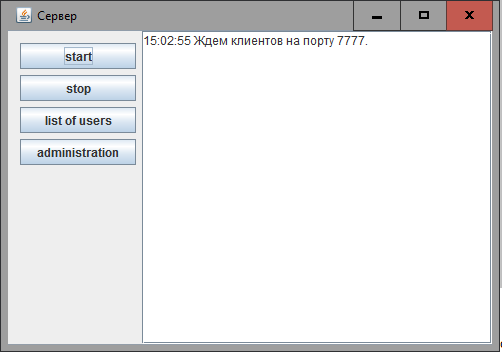


Рисунок А.6 – Головна форма серверу

Тут ми можемо бачити запуск та зупинення сервера, список клієнтів та адміністрування основних деталей.

При виборі списку клієнтів ми попадаємо на форму, де можемо проглянути список усіх користувачів, а також їх поточний статус, та персональні дані. Форма з клієнтами представлена на рис. А.7.

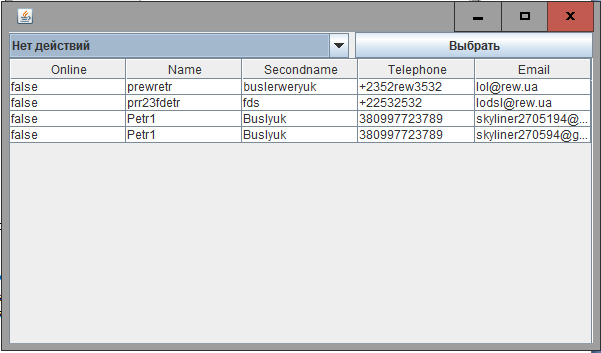


Рисунок А.7 – Форма клієнтів

Ми можемо відіслати заданим користувачам повідомлення на емейл адресу. Для цього необхідно обрати необхідних клієнтів на у випадаючому списку обрати «Вислати повідомлення» та натиснути на кнопку «Обрати».

Система запросить ввести Вас заголовок повідомлення та саме повідомлення. Після цього проінформує про відправлення.

Результат можна проглянути на рис. А.8.

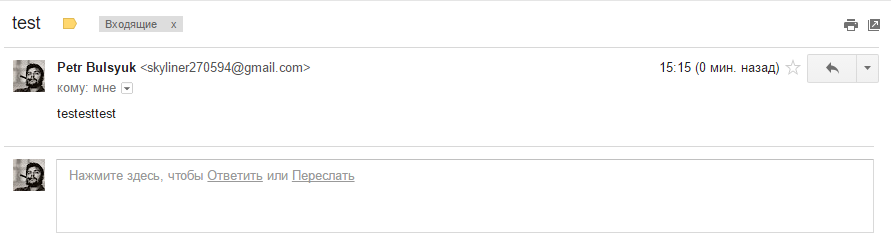


Рисунок А.8 – Поштова скринька із результатом відправлених даних на вказаного клієнта

Відправка даних також була протестована не тільки для поштової адреси на gmail.com але й на інших, наприклад прийшло повідомлення на skyliner270594@mail.ru із тими даними, які були віправлені.

**Додаток Б. ЛІСТИНГ ПРОГРАМИ**

Клас client

package client;

import java.net.\*;

import java.io.\*;

import java.util.logging.Level;

import java.util.logging.Logger;

public class client {

public static Socket socket;

private static clientProfile cg;

private static companies comp;

public static ObjectMessenger om;

private String server="localhost";

private static String email;

protected final static int LOGIN=0,QUIT=1,SAVE\_DATA=5,RESTORE\_DATA=6;

private final int port;

private static boolean access=true;

static boolean sendedOurData=false;

client(String server,int port,clientProfile cg, String e){

this.server = server;

this.port = port;

this.cg = cg;

this.email = e;

}

void set\_comp(companies comp){

this.comp = comp;

}

private void display(String msg) {cg.append(msg + "\n");}

//Client-server arc

public boolean start(){

try {

// try to connect to the server

try {socket = new Socket(server, port);}

catch(Exception ec) {return false;}

om = new ObjectMessenger(socket);

// creates the Thread to listen from the server

new ListenFromServer().start();

//send username data

if(check\_connect()){

sendData();

}else{

sendedOurData = false;

disconnect();return false;

}

}

catch(IOException ex) {}

return true;

}

protected void sendData(){

try {

om.sendObject(LOGIN,cg.get\_firstname()+";"+cg.get\_secondname()+";"

+cg.get\_telephone()+";"+cg.get\_email());

sendedOurData = true;

} catch (IOException ex) {

Logger.getLogger(client.class.getName()).log(Level.SEVERE, null, ex);

}

}

protected static boolean check\_connect(){return socket.isConnected();}

protected void close\_main\_window(){

if(socket != null){

try {

om.sendObject(QUIT,"");

} catch (IOException ex) {}

}

System.exit(0);

}

void sendDataToServer() {

new SendDataToServer().start();

}

void restoreDataFromServer() {

new ClientThreadSaveData().start();

}

public static class ObjectMessenger{

protected static Socket socket;

protected ObjectOutputStream oos;

protected ObjectInputStream ois;

protected ObjectMessenger (Socket socket) throws UnknownHostException, IOException{

this.socket = socket;

}

protected void sendObject (int type,Object object) throws IOException{

if(check\_connect()){

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

oos.writeObject(type+";"+object);

oos.flush();

}

}

protected String receiveObject() throws IOException, ClassNotFoundException{

if(check\_connect()){

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

return (String) ois.readObject();

}else return QUIT+";";

}

protected void setType(String receiveObject) throws IOException {

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

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

case QUIT:{

disconnect();

}

case RESTORE\_DATA:{

new ClientThreadSaveData().start();

}

break;

}

}

}

static class ClientThreadSaveData extends Thread{

public void run() {

try {

ServerSocket servSocket;

Socket fromClientSocket;

int cTosPortNumber = 1777;

String str1;

companiesToSent comp;

servSocket = new ServerSocket(cTosPortNumber);

om.sendObject(RESTORE\_DATA,email);

System.out.println("Waiting for a connection on " + cTosPortNumber);

fromClientSocket = servSocket.accept();

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

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

while ((comp = (companiesToSent) ois.readObject()) != null) {

companies.setCompanies(comp);

oos.writeObject("bye bye");

break;

}

oos.close();

ois.close();

fromClientSocket.close();

servSocket.close();

} catch (IOException | ClassNotFoundException ex) {

}

}

}

private static void disconnect() throws IOException {

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

cg.connectionFailed();

}

public class ListenFromServer extends Thread {

@Override

public void run(){

while(true){

try{

String msg = om.receiveObject();

om.setType(msg);

cg.append(msg);

}catch(IOException e){

try {

disconnect(); break;

} catch (IOException ex) {}

} catch (ClassNotFoundException ex) {}

}

}

}

public class SendDataToServer extends Thread {

@Override

public void run(){

try {

om.sendObject(SAVE\_DATA,email);

Socket socket1;

int portNumber = 1777;

String str = "";

socket1 = new Socket(InetAddress.getLocalHost(), portNumber);

ObjectInputStream ois = new ObjectInputStream(socket1.getInputStream());

ObjectOutputStream oos = new ObjectOutputStream(socket1.getOutputStream());

companiesToSent companiesSent = new companiesToSent();

companiesSent.setCompanies(companies.getCompaniesToObject());

oos.writeObject(companiesSent);

while ((str = (String) ois.readObject()) != null) {

System.out.println(str);

oos.writeObject("bye");

if (str.equals("bye"))

break;

}

ois.close();

oos.close();

socket1.close();

} catch (IOException | ClassNotFoundException ex) {

}

}

}

}

Клас clientProfile

package client;

import java.io.\*;

import java.util.Timer;

import java.util.TimerTask;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.JOptionPane;

import javax.swing.JTextField;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class clientProfile extends javax.swing.JFrame {

private boolean connected;

private client c;

private companies comp;

private File profile= new File(companies.get\_current\_dir()+"profile.txt");

private String firstname,secondname,telephone,email;

private boolean sendedOurData=false;

Timer timer;

Timer timer1;

public clientProfile() {

super("ClientProfile");

try {

initComponents();

if(profile.exists()){

BufferedReader in = null;

try {

in = new BufferedReader(new FileReader(profile));

String [] mas = in.readLine().split(";");

fn.setText(mas[0]);

sn.setText(mas[1]);

tn.setText(mas[2]);

em.setText(mas[3]);

fill\_client\_all(mas[0], mas[1], mas[2], mas[3]);

} catch (FileNotFoundException ex) {} catch (IOException ex) {} catch (Exception ex) {} finally {

try { in.close(); } catch (IOException ex) {}

}

} else check\_and\_send();

c = new client("localhost", 7777, this, email);

timer1 = new Timer();

timer1.schedule(new TimerTask(){

public void run(){

if(firstname.equals("firstname") || secondname.equals("secondname") ||

telephone.equals("01234567") || email.equals("example@mail.ru")){

companies.show("Заполните профиль! После этого нажмите кнопку \"Сохранить\".");

}

}

},0L, 10000L);

timer = new Timer();

c.start();

timer.schedule( new TimerTask(){

public void run(){

if(c.socket==null){

c.start();

}

if(!c.sendedOurData && c.socket!=null){

try{

c.sendData();

} catch (Exception ex) {companies.log(ex);}

}

}

}, 0L ,1000L);

} catch (Exception ex) {}

}

protected void fill\_client\_all(String firstname,String secondname,

String telephone,String email) throws Exception{

this.firstname=firstname;

this.secondname=secondname;

this.telephone = telephone;

this.email = email;

}

protected boolean check(JTextField tf,String tfn){

if(tf.getText().equals("")){

JOptionPane.showMessageDialog(rootPane, "Заполните поле "+tfn);

return false;

}else{return true;}

}

protected boolean checkEmail(String em){

return Pattern.compile("^[\_A-Za-z0-9-\\+]+(\\.[\_A-Za-z0-9-]+)\*@" +

"[A-Za-z0-9-]+(\\.[A-Za-z0-9]+)\*(\\.[A-Za-z]{2,})$").matcher(em).matches();

}

protected void check\_and\_send(){

if(!checkEmail(em.getText())){

companies.show("Неправильный емейл");

return ;

}

if(check(fn,"имени") && check(sn,"фамилии") && check(tn,"телефона") && check(em,"емейла")){

try {

connected=true;

String str=fn.getText()+";"+sn.getText()+";"+tn.getText()+";"+em.getText();

profile.createNewFile();

write(profile,str);

} catch (IOException ex) {}

}else connected=false;

}

protected static void write(File file, String text) {

try {

if(!file.exists()){file.createNewFile();}

PrintWriter out = new PrintWriter(file.getAbsoluteFile());

try {out.print(text);

} finally {out.close();}

} catch(IOException e){throw new RuntimeException(e);}

}

protected String get\_firstname() {return this.firstname;}

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

protected String get\_telephone() {return this.telephone;}

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

void append(String string) {JOptionPane.showMessageDialog(null, string);}

void connectionFailed(){connected = false; append("Соединение с сервером закрыто.");}

void close\_main\_window() throws IOException {c.close\_main\_window();}

@SuppressWarnings("unchecked")

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

private void initComponents() {

save\_client = new javax.swing.JButton();

fn = new javax.swing.JTextField();

sn = new javax.swing.JTextField();

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

tn = new javax.swing.JTextField();

jLabel7 = new javax.swing.JLabel();

em = new javax.swing.JTextField();

setTitle("Профиль клиента");

setResizable(false);

save\_client.setText("Сохранить");

save\_client.addActionListener(new java.awt.event.ActionListener() {

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

save\_clientActionPerformed(evt);

}

});

fn.setText("firstname");

sn.setText("secondname");

jLabel1.setText("Имя");

jLabel2.setText("Фамилия");

jLabel4.setText("Телефон");

tn.setText("01234567");

jLabel7.setText("Е-мейл");

em.setText("example@mail.ru");

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

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

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

.addGroup(layout.createSequentialGroup()

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

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

.addComponent(save\_client)

.addGroup(layout.createSequentialGroup()

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

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

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel1)

.addComponent(jLabel2))

.addComponent(jLabel4))

.addComponent(jLabel7))

.addGap(53, 53, 53)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(sn)

.addComponent(fn)

.addComponent(tn)

.addComponent(em, javax.swing.GroupLayout.DEFAULT\_SIZE, 223, Short.MAX\_VALUE))

.addGap(3, 3, 3)))

.addContainerGap())

);

layout.setVerticalGroup(

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

.addGroup(layout.createSequentialGroup()

.addContainerGap()

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

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

.addComponent(jLabel1))

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

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

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

.addComponent(jLabel2))

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

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

.addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED\_SIZE, 17, javax.swing.GroupLayout.PREFERRED\_SIZE)

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

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

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

.addComponent(jLabel7)

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

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 18, Short.MAX\_VALUE)

.addComponent(save\_client)

.addContainerGap())

);

pack();

setLocationRelativeTo(null);

}// </editor-fold>

private void save\_clientActionPerformed(java.awt.event.ActionEvent evt) {

try {

check\_and\_send();

fill\_client\_all(fn.getText(), sn.getText(), tn.getText(), em.getText());

this.setVisible(false);

} catch (Exception ex) {

Logger.getLogger(clientProfile.class.getName()).log(Level.SEVERE, null, ex);

}

}

// Variables declaration - do not modify

private javax.swing.JTextField em;

private javax.swing.JTextField fn;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel4;

private javax.swing.JLabel jLabel7;

private javax.swing.JButton save\_client;

private javax.swing.JTextField sn;

private javax.swing.JTextField tn;

// End of variables declaration

void set\_comp(companies comp) {

this.comp = comp;

}

void sendDataToServer() {

c.sendDataToServer();

}

void restoreData() {

c.restoreDataFromServer();

}

}

Клас companies

package client;

import java.awt.Window;

import javax.swing.JOptionPane;

import java.io.File;

import java.io.IOException;

import java.io.Serializable;

import java.math.BigDecimal;

import java.util.ArrayList;

import java.util.Iterator;

import java.util.Vector;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.table.DefaultTableModel;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.ParserConfigurationException;

import org.w3c.dom.Node;

import org.xml.sax.SAXException;

public class companies extends javax.swing.JFrame implements Serializable{

private static clientProfile c1;

private static ArrayList<company> companys=new ArrayList<>();

private static File file= new File(get\_current\_dir()+"companies.xml");

private final static String version = "1.0";

private static companies comp;

private static selectedCompany sc;

public companies(){

initComponents();

}

@SuppressWarnings("unchecked")

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

private void initComponents() {

jScrollPane1 = new javax.swing.JScrollPane();

ct = new javax.swing.JTable();

menu = new javax.swing.JMenuBar();

companies = new javax.swing.JMenu();

generate\_report\_to\_all = new javax.swing.JMenuItem();

select\_all\_companies = new javax.swing.JMenuItem();

calculate\_selected = new javax.swing.JMenuItem();

separator = new javax.swing.JPopupMenu.Separator();

add\_company = new javax.swing.JMenuItem();

delete\_company = new javax.swing.JMenuItem();

separator1 = new javax.swing.JPopupMenu.Separator();

saveDataToServer = new javax.swing.JMenuItem();

restoreDataFromServer = new javax.swing.JMenuItem();

jMenu1 = new javax.swing.JMenu();

calculate = new javax.swing.JMenuItem();

edit\_cname = new javax.swing.JMenuItem();

edit\_cdepo = new javax.swing.JMenuItem();

edit\_cpercent = new javax.swing.JMenuItem();

edit\_cperiod = new javax.swing.JMenuItem();

profile = new javax.swing.JMenu();

programm = new javax.swing.JMenu();

about\_programm = new javax.swing.JMenuItem();

save\_companies = new javax.swing.JMenu();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setTitle("Компании");

setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT\_CURSOR));

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowClosing(java.awt.event.WindowEvent evt) {

closeWindow(evt);

}

});

ct.setFont(new java.awt.Font("Arial", 0, 11)); // NOI18N

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

new Object [][] {

},

new String [] {

"Название", "Депозит", "Средний %", "Период инвестирования"

}

) {

boolean[] canEdit = new boolean [] {

false, false, false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

ct.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

ctMouseClicked(evt);

}

});

jScrollPane1.setViewportView(ct);

if (ct.getColumnModel().getColumnCount() > 0) {

ct.getColumnModel().getColumn(2).setMaxWidth(100);

}

companies.setText("Компании");

companies.setFont(new java.awt.Font("Arial", 0, 12)); // NOI18N

generate\_report\_to\_all.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_R, java.awt.event.InputEvent.CTRL\_MASK));

generate\_report\_to\_all.setText("Сформировать отчет");

generate\_report\_to\_all.addActionListener(new java.awt.event.ActionListener() {

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

generate\_report\_to\_allActionPerformed(evt);

}

});

companies.add(generate\_report\_to\_all);

select\_all\_companies.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_A, java.awt.event.InputEvent.CTRL\_MASK));

select\_all\_companies.setText("Выбрать все компании");

select\_all\_companies.addActionListener(new java.awt.event.ActionListener() {

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

select\_all\_companiesActionPerformed(evt);

}

});

companies.add(select\_all\_companies);

calculate\_selected.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_C, java.awt.event.InputEvent.ALT\_MASK));

calculate\_selected.setText("Расчитать выбранные");

calculate\_selected.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

calculate\_selectedMouseClicked(evt);

}

});

companies.add(calculate\_selected);

companies.add(separator);

add\_company.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_N, java.awt.event.InputEvent.CTRL\_MASK));

add\_company.setText("Добавить компанию");

add\_company.addActionListener(new java.awt.event.ActionListener() {

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

add\_companyActionPerformed(evt);

}

});

companies.add(add\_company);

delete\_company.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_D, java.awt.event.InputEvent.CTRL\_MASK));

delete\_company.setText("Удалить компанию");

delete\_company.addActionListener(new java.awt.event.ActionListener() {

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

delete\_companyActionPerformed(evt);

}

});

companies.add(delete\_company);

companies.add(separator1);

saveDataToServer.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_S, java.awt.event.InputEvent.ALT\_MASK | java.awt.event.InputEvent.CTRL\_MASK));

saveDataToServer.setText("Сохранить данные удаленно");

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

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

saveDataToServerActionPerformed(evt);

}

});

companies.add(saveDataToServer);

restoreDataFromServer.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_R, java.awt.event.InputEvent.ALT\_MASK | java.awt.event.InputEvent.CTRL\_MASK));

restoreDataFromServer.setText("Восстановить данные с сервера");

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

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

restoreDataFromServerActionPerformed(evt);

}

});

companies.add(restoreDataFromServer);

menu.add(companies);

jMenu1.setText("Действия");

calculate.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_C, java.awt.event.InputEvent.ALT\_MASK));

calculate.setText("Расчитать данные");

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

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

calculateActionPerformed(evt);

}

});

jMenu1.add(calculate);

edit\_cname.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_N, java.awt.event.InputEvent.ALT\_MASK));

edit\_cname.setText("Редактировать название");

edit\_cname.addActionListener(new java.awt.event.ActionListener() {

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

edit\_cnameActionPerformed(evt);

}

});

jMenu1.add(edit\_cname);

edit\_cdepo.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_D, java.awt.event.InputEvent.ALT\_MASK));

edit\_cdepo.setText("Редактировать депозит");

edit\_cdepo.addActionListener(new java.awt.event.ActionListener() {

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

edit\_cdepoActionPerformed(evt);

}

});

jMenu1.add(edit\_cdepo);

edit\_cpercent.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_P, java.awt.event.InputEvent.ALT\_MASK));

edit\_cpercent.setText("Редактировать процент");

edit\_cpercent.addActionListener(new java.awt.event.ActionListener() {

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

edit\_cpercentActionPerformed(evt);

}

});

jMenu1.add(edit\_cpercent);

edit\_cperiod.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_I, java.awt.event.InputEvent.ALT\_MASK));

edit\_cperiod.setText("Редактировать период инвестирования");

edit\_cperiod.addActionListener(new java.awt.event.ActionListener() {

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

edit\_cperiodActionPerformed(evt);

}

});

jMenu1.add(edit\_cperiod);

menu.add(jMenu1);

profile.setText("Профиль");

profile.setFont(new java.awt.Font("Arial", 0, 12)); // NOI18N

profile.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

profileMouseClicked(evt);

}

});

menu.add(profile);

programm.setText("Программа");

programm.setFont(new java.awt.Font("Arial", 0, 12)); // NOI18N

about\_programm.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_F1, 0));

about\_programm.setText("О программе");

about\_programm.addActionListener(new java.awt.event.ActionListener() {

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

about\_programmActionPerformed(evt);

}

});

programm.add(about\_programm);

menu.add(programm);

save\_companies.setText("Сохранить данные локально");

save\_companies.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

save\_companies(evt);

}

});

menu.add(save\_companies);

setJMenuBar(menu);

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

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

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

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT\_SIZE, 503, Short.MAX\_VALUE)

.addContainerGap())

);

layout.setVerticalGroup(

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

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addContainerGap()

.addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT\_SIZE, 275, Short.MAX\_VALUE)

.addContainerGap())

);

pack();

setLocationRelativeTo(null);

}// </editor-fold>

private void select\_all\_companiesActionPerformed(java.awt.event.ActionEvent evt) {

if(ct.getRowCount()==0) JOptionPane.showMessageDialog(null, "Добавьте компанию");

else ct.selectAll();

}

private void profileMouseClicked(java.awt.event.MouseEvent evt) {

c1.setVisible(true);

}

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

if(ct.getSelectedRowCount()==0) JOptionPane.showMessageDialog(null, "Выберите компании");

else{

int[] sr = ct.getSelectedRows();

String [] cs = calculate\_selected(sr);

JOptionPane.showMessageDialog(null, "Были выбраны следующие компании:\n"

+cs[0]+BigDecimal.valueOf(Double.parseDouble(cs[1])).setScale(2,BigDecimal.ROUND\_HALF\_DOWN).doubleValue());

}

}

private void add\_companyActionPerformed(java.awt.event.ActionEvent evt) {

DefaultTableModel model = (DefaultTableModel)ct.getModel();

String name = JOptionPane.showInputDialog("Введите название");

for (company compName: companys){

if (compName.get\_name().equals(name)){

show("Такая компания уже существует");

return ;

}

}

if (name.isEmpty()){

show("Вы не ввели название компании");

return ;

}

String depo = JOptionPane.showInputDialog("Введите депозит");

if (!isDouble(depo)){

show("Вы ввели некорректный депозит");

return ;

}

String persent = JOptionPane.showInputDialog("Введите средний процент");

if (!isDouble(persent)){

show("Вы ввели некорректный процент");

return ;

}

String period = JOptionPane.showInputDialog("Введите длительность инвестирования");

if (!isInteger(period)){

show("Вы ввели некорректный период");

return ;

}

company c = new company(name,depo,persent,period,

new ArrayList<>(),new ArrayList<>(),new ArrayList<>());

companys.add(c);

save\_companies();

try {

get\_companies();

} catch (ParserConfigurationException | SAXException | IOException ex) {

Logger.getLogger(companies.class.getName()).log(Level.SEVERE, null, ex);

}

show\_companies();

}

public static boolean isInteger(String s) {

boolean isValidInteger = false;

try{

Integer.parseInt(s);

isValidInteger = true;

}catch (NumberFormatException ex){}

return isValidInteger;

}

public static boolean isDouble(String s) {

boolean isValidBoolean = false;

try{

Double.parseDouble(s);

isValidBoolean = true;

}catch (NumberFormatException ex){}

return isValidBoolean;

}

private void delete\_companyActionPerformed(java.awt.event.ActionEvent evt) {

deleteCompanies();

}

private void closeWindow(java.awt.event.WindowEvent evt) {

try {c1.close\_main\_window();} catch (IOException ex) {}

}

private void about\_programmActionPerformed(java.awt.event.ActionEvent evt) {

JOptionPane.showMessageDialog(null, "Программа для расчета перспективной"

+ " прибыли\n от инвестиционной деятельности.");

}

private void ctMouseClicked(java.awt.event.MouseEvent evt) {

if (evt.getClickCount() == 2) {

sc.setVisible(true);

String s = ct.getValueAt(ct.getSelectedRow(), 0).toString();

showSelectedCompany(s);

}

}

private void save\_companies(java.awt.event.MouseEvent evt) {

save\_companies();

}

private void calculate\_selectedMouseClicked(java.awt.event.MouseEvent evt) {

int[] sr = ct.getSelectedRows();

calculate\_selected(sr);

}

private void edit\_cnameActionPerformed(java.awt.event.ActionEvent evt) {

switch (ct.getSelectedRowCount()) {

case 0:show("Выберите компанию"); break;

case 1:

String new\_cname = input("Изменение названия комании","Введите новое название компании");

companys.get(ct.getSelectedRow()).set\_name(new\_cname);

show\_companies();

save\_companies();

break;

default:show("Выбрано больше 1 компании");break;

}

}

private void edit\_cdepoActionPerformed(java.awt.event.ActionEvent evt) {

switch (ct.getSelectedRowCount()) {

case 0:show("Выберите компанию"); break;

case 1:

String new\_cdepo = input("Изменение депозита","Введите новый начальный депозит");

companys.get(ct.getSelectedRow()).set\_depo(new\_cdepo);

show\_companies();

save\_companies();

break;

default:show("Выбрано больше 1 компании");break;

}

}

private void edit\_cpercentActionPerformed(java.awt.event.ActionEvent evt) {

switch (ct.getSelectedRowCount()) {

case 0:show("Выберите компанию"); break;

case 1:

String new\_cpersent = input("Изменение процента","Введите средний процент");

companys.get(ct.getSelectedRow()).set\_persent(new\_cpersent);

save\_companies();

try {

get\_companies();

} catch (ParserConfigurationException | SAXException | IOException ex) {

log(ex);

}

break;

default:show("Выбрано больше 1 компании");break;

}

}

private void edit\_cperiodActionPerformed(java.awt.event.ActionEvent evt) {

switch (ct.getSelectedRowCount()) {

case 0:show("Выберите компанию"); break;

case 1:

String new\_cperiod = input("Изменение периода","Введите период инвестирования");

companys.get(ct.getSelectedRow()).set\_period(new\_cperiod);

show\_companies();

save\_companies();

break;

default:show("Выбрано больше 1 компании");break;

}

}

private void generate\_report\_to\_allActionPerformed(java.awt.event.ActionEvent evt) {

int[] sr = ct.getSelectedRows();

if(sr.length==0){

show("Вы не выбрали компаний");

}else{

ArrayList<company> CompanyList = new ArrayList<>();

for(int i=0;i<sr.length;i++){

String company\_name = ct.getValueAt(sr[i], 0).toString();

companys.stream().filter((Company) -> (Company.get\_name().equals(company\_name))).forEach((Company) -> {

CompanyList.add(Company);

});

}

createPDFReport.createReport(true, CompanyList, null);

CompanyList.clear();

}

}

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

// TODO add your handling code here:

c1.sendDataToServer();

}

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

// TODO add your handling code here:

c1.restoreData();

}

static void log(Object o){

System.out.println(o);

}

static String input(String title, String massage){

return JOptionPane.showInputDialog(null, massage, title);

}

static void show(String massage){

JOptionPane.showMessageDialog(null, massage);

}

static String [] calculate\_selected(int[] sr){

String companies\_table="";

double depo=0;

for(int i=0;i<sr.length;i++){

int period = Integer.parseInt(ct.getValueAt(sr[i], 3).toString());

String ct\_new =ct.getValueAt(sr[i], 0).toString();

showSelectedCompany(ct\_new);

depo += sc.calculate().get\_depolast(period-1);

companies\_table += ct\_new+"\n";

}

return new String [] {companies\_table,Double.toString(depo)};

}

protected static void deleteCompanies(){

if(ct.getSelectedRowCount()>0){

DefaultTableModel model = (DefaultTableModel)ct.getModel();

int[] row = ct.getSelectedRows();

for(int row1 : row){

companys.remove(row1);

}

save\_companies();

show\_companies();

}

}

protected static String get\_current\_dir(){

String path = companies.class.getProtectionDomain().getCodeSource().getLocation().getPath();

return path.substring(1,path.length());

}

private static void get\_companies() throws ParserConfigurationException, SAXException, IOException{

Node doc = DocumentBuilderFactory.newInstance()

.newDocumentBuilder().parse(file);

companys = new ArrayList<>();

String name,depo,persent,period;

ArrayList<String> plus,minus,reinvesting;

Node n = doc.getFirstChild();

if(n.getNodeName().equals("companies")){

while(n!=null){

Node m = n.getFirstChild();

while(m != null){

if(m.getNodeName().equals("company")) {

name=depo=persent=period="";

plus=new ArrayList<>();

minus=new ArrayList<>();

reinvesting=new ArrayList<>();

Node comp=m.getFirstChild();

while(comp!=null){

switch (comp.getNodeName()){

case "name":name=comp.getTextContent().trim();break;

case "depo":depo=comp.getTextContent().trim();break;

case "persent":persent=comp.getTextContent().trim();break;

case "period":period=comp.getTextContent().trim();break;

case "intervals":{

Node i=comp.getFirstChild();

while(i!=null){

if(i.getNodeName().equals("i")){

Node t = i.getFirstChild();

while(t!=null){

switch (t.getNodeName()){

case "plus":{

plus.add(t.getTextContent());

};break;

case "minus":{

minus.add(t.getTextContent());

};break;

case "reinvesting":{

reinvesting.add(t.getTextContent());

};break;

default:;break;

}

t = t.getNextSibling();

}

}

i = i.getNextSibling();

}

};break;

default:break;

}

comp=comp.getNextSibling();

}

company c = new company(name,depo,persent,period,plus,minus,reinvesting);

companys.add(c);

}

m = m.getNextSibling();

}

n = n.getNextSibling();

}

}

show\_companies();

}

static void save\_companies(){

StringBuilder str = new StringBuilder();

str.append("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n")

.append("<companies>\n");

companys.stream().forEach((c2)->{str.append(c2.get\_company\_to\_string());});

str.append("</companies>");

clientProfile.write(file,str.toString());

}

static void set\_updated\_company(company c1){

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

if(companys.get(i).get\_name().equals(c1.get\_name())){

companys.set(i, c1);

break;

}

}

save\_companies();

}

protected static void show\_companies(){

((DefaultTableModel)ct.getModel()).setNumRows(0);

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

((DefaultTableModel)ct.getModel()).addRow(

new Object[]{companys.get(i).get\_name(),

companys.get(i).get\_depo(),

companys.get(i).get\_persent(),

companys.get(i).get\_period()

});}

}

static void showSelectedCompany(String selectedCName){

companys.stream().forEach((company c)->{

if(c.get\_name().equals(selectedCName)){

sc.setSelectedCompany(c);

}

});

}

public static void main(String args[]) {

java.awt.EventQueue.invokeLater(() -> {

try {

comp = new companies();

comp.setVisible(true);

} catch (Exception ex) {

log(ex);

}

});

try{

sc = new selectedCompany();

sc.setVisible(false);

c1 = new clientProfile();

c1.set\_comp(comp);

c1.setVisible(false);

get\_companies();

int [] i=new int[companys.size()];int a=0;

for(company c:companys){i[a]=a++;}

calculate\_selected(i);

}catch(Exception ex){log(ex);}

}

protected static ArrayList<company> getCompaniesToObject(){

return companys;

}

protected static void setCompanies(companiesToSent v){

companys.clear();

Iterator i = v.getCampanyIterator();

while(i.hasNext()){

company c = (company) i.next();

companys.add(c);

}

show\_companies();

}

// Variables declaration - do not modify

private javax.swing.JMenuItem about\_programm;

private javax.swing.JMenuItem add\_company;

private javax.swing.JMenuItem calculate;

private javax.swing.JMenuItem calculate\_selected;

private javax.swing.JMenu companies;

private static javax.swing.JTable ct;

private javax.swing.JMenuItem delete\_company;

private javax.swing.JMenuItem edit\_cdepo;

private javax.swing.JMenuItem edit\_cname;

private javax.swing.JMenuItem edit\_cpercent;

private javax.swing.JMenuItem edit\_cperiod;

private javax.swing.JMenuItem generate\_report\_to\_all;

private javax.swing.JMenu jMenu1;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JMenuBar menu;

private javax.swing.JMenu profile;

private javax.swing.JMenu programm;

private javax.swing.JMenuItem restoreDataFromServer;

private javax.swing.JMenuItem saveDataToServer;

private javax.swing.JMenu save\_companies;

private javax.swing.JMenuItem select\_all\_companies;

private javax.swing.JPopupMenu.Separator separator;

private javax.swing.JPopupMenu.Separator separator1;

// End of variables declaration

}

Клас companiesToSent

package client;

import java.io.Serializable;

import java.util.ArrayList;

import java.util.Iterator;

import java.util.Vector;

public class companiesToSent extends Thread implements Serializable {

public Vector companiesToSend;

public companiesToSent() {

companiesToSend = new Vector();

}

public void setCompanies(ArrayList<company> companiesList){

companiesList.stream().forEach((c) -> {

companiesToSend.add(c);

});

}

public Iterator getCampanyIterator() {

return companiesToSend.iterator();

}

public void printSomething(){

Iterator i = getCampanyIterator();

while(i.hasNext()){

company c = (company) i.next();

System.out.println(c.get\_name());

}

}

}

Клас company

package client;

import java.io.Serializable;

import java.math.BigDecimal;

import java.util.ArrayList;

public class company implements Serializable{

private String name,depo,persent,period;

private final ArrayList<String> plus, minus, reinvesting;

private final ArrayList<Double> depofirst, depolast, resultbuffer;

public company(){

name="";depo="";period="";persent="";

plus = new ArrayList<>();

minus = new ArrayList<>();

reinvesting = new ArrayList<>();

depofirst = new ArrayList<>();

depolast = new ArrayList<>();

resultbuffer = new ArrayList<>();

}

public company(String \_name,String \_depo, String \_persent, String \_period,

ArrayList<String> \_plus,ArrayList<String> \_minus,ArrayList<String> \_reinvesting) {

name=\_name;depo=\_depo;period=\_period;persent=\_persent;

plus = \_plus;

minus = \_minus;

reinvesting = \_reinvesting;

depofirst = new ArrayList<>();

depolast = new ArrayList<>();

resultbuffer = new ArrayList<>();

}

public void set\_name(String \_name){name=\_name;}

public void set\_depo(String \_depo){depo=\_depo;}

public void set\_persent(String \_persent){persent=\_persent;}

public void set\_period(String \_period){period=\_period;}

public void set\_plus(String \_plus){plus.add(\_plus);}

public void set\_minus(String \_minus){minus.add(\_minus);}

public void set\_reinvesting(String \_reinvesting){reinvesting.add(\_reinvesting);}

public void set\_plus(int i,String \_plus){plus.set(i,\_plus);}

public void set\_minus(int i,String \_minus){minus.set(i,\_minus);}

public void set\_reinvesting(int i,String \_reinvesting){reinvesting.set(i,\_reinvesting);}

public void set\_depofirst(int i,double df){depofirst.set(i,df);}

public void set\_depolast(int i,double dl){depolast.set(i,dl);}

public void set\_resultbuffer(int i,double rb){resultbuffer.set(i,rb);}

public ArrayList get\_depolast(){return depolast;}

public double get\_depolast(int i){return depolast.get(i);}

public double get\_depofirst(int i){return depofirst.get(i);}

public double get\_resultbuffer(int i){return resultbuffer.get(i);}

public ArrayList get\_depofirst(){return depofirst;}

public String get\_name(){return name;}

public String get\_persent(){return persent;}

public String get\_period(){return period;}

public String get\_depo(){return depo;}

public String get\_plus(int i){return plus.get(i);}

public String get\_minus(int i){return minus.get(i);}

public boolean get\_reinvesting(int i){return Boolean.parseBoolean(reinvesting.get(i));}

public ArrayList get\_plus(){return plus;}

public ArrayList get\_minus(){return minus;}

public ArrayList get\_reinvesting(){return reinvesting;}

public String get\_company\_to\_string(){

log("here");

StringBuilder str = new StringBuilder();

str.append("<company>\n")

.append("\t<name>").append(name).append("</name>\n")

.append("\t<depo>").append(depo).append("</depo>\n")

.append("\t<persent>").append(persent).append("</persent>\n")

.append("\t<period>").append(period).append("</period>\n")

.append("\t<intervals>\n");

int period1 = Integer.parseInt(period);

for(int i=0; i < period1;i++){

str.append("\t<i>\n")

.append("\t <plus>");

str.append((plus.size()>i)?plus.get(i):"0");

str.append("</plus>\n");

str.append("\t <minus>");

str.append((minus.size()>i)?minus.get(i):"0");

str.append("</minus>\n");

str.append("\t <reinvesting>");

str.append((reinvesting.size()>i)?reinvesting.get(i):"false");

str.append("</reinvesting>\n");

str.append("\t</i>\n");

}

str.append("</intervals>\n")

.append("</company>\n");

return str.toString();

}

public Object [] get\_company\_to\_object(){

return new Object[]{name,depo,persent,period};

}

public String [] get\_company\_to\_array\_in\_report(){

int size = Integer.parseInt(get\_period()) \* 6;

String [] result = new String[size];

int i = 0,a = 0;

while(i!=size){

result[i]=Integer.toString(a);++i;

result[i]=get\_plus(a);++i;

result[i]=get\_minus(a);++i;

result[i]=Boolean.toString(get\_reinvesting(a));++i;

result[i]=Double.toString(get\_depofirst(a));++i;

result[i]=Double.toString(get\_depolast(a));++i;

++a;

}

return result;

}

company getCalculateCompany(){

double depo1 = Double.parseDouble(get\_depo());

double persent1 = Double.parseDouble(get\_persent());

int period1 = Integer.parseInt(get\_period());

ArrayList<String> plus1 = get\_plus();

ArrayList<String> minus1 = get\_minus();

ArrayList<String> reinvesting1 = get\_reinvesting();

double plus2,minus2;

depofirst.clear();

depolast.clear();

resultbuffer.clear();

for(int i=0;i<period1;i++){

depofirst.add(0.0);

depolast.add(0.0);

resultbuffer.add(0.0);

}

for(int i=0;i<period1;i++){

plus2 = (plus1.get(i).equals(""))? 0: Double.parseDouble(plus1.get(i));

minus2 = (minus1.get(i).equals(""))? 0:Double.parseDouble(minus1.get(i));

if(Boolean.parseBoolean(reinvesting1.get(i))){

set\_depofirst(i,(i==0)?depo1:

((Boolean.parseBoolean(reinvesting1.get(i-1)))?

BigDecimal.valueOf(get\_depolast(i-1)+ plus2).setScale(2,BigDecimal.ROUND\_HALF\_DOWN).doubleValue():

BigDecimal.valueOf(get\_depofirst(i-1)+ plus2).setScale(2,BigDecimal.ROUND\_HALF\_DOWN).doubleValue()));

set\_resultbuffer(i,get\_depofirst(i)\*(persent1/100) - minus2);

set\_depolast(i,BigDecimal.valueOf(get\_resultbuffer(i)+get\_depofirst(i)).setScale(2,BigDecimal.ROUND\_HALF\_DOWN).doubleValue());

}else{

set\_depofirst(i,(i==0)?depo1:

((Boolean.parseBoolean(reinvesting1.get(i-1)))?BigDecimal.valueOf(get\_depolast(i-1)+ plus2).setScale(2,BigDecimal.ROUND\_HALF\_DOWN).doubleValue():

BigDecimal.valueOf(get\_depofirst(i-1)+ plus2).setScale(2,BigDecimal.ROUND\_HALF\_DOWN).doubleValue()

));

set\_resultbuffer(i,get\_depofirst(i)\*(persent1/100) - minus2);

set\_depolast(i,BigDecimal.valueOf(get\_resultbuffer(i)+get\_depofirst(i)+((i==0)?0:get\_resultbuffer(i-1))).setScale(2,BigDecimal.ROUND\_HALF\_DOWN).doubleValue());

}

}

return this;

}

static void log(Object o){

System.out.println(o);

}

}

Клас createPDFReport

package client;

import java.io.FileOutputStream;

import com.itextpdf.text.BadElementException;

import com.itextpdf.text.Document;

import com.itextpdf.text.DocumentException;

import com.itextpdf.text.Element;

import com.itextpdf.text.Font;

import com.itextpdf.text.Paragraph;

import com.itextpdf.text.Phrase;

import com.itextpdf.text.pdf.PdfPCell;

import com.itextpdf.text.pdf.PdfPTable;

import com.itextpdf.text.pdf.PdfWriter;

import java.io.FileNotFoundException;

import java.util.ArrayList;

import java.util.logging.Level;

import java.util.logging.Logger;

public class createPDFReport {

static String namePDFReport;

private static String FILE;

private static final Font CATFONT = new Font(Font.FontFamily.TIMES\_ROMAN, 18, Font.BOLD);

private static final Font SUBFONT = new Font(Font.FontFamily.TIMES\_ROMAN, 16, Font.BOLD);

private static final Font SMALLBOLD = new Font(Font.FontFamily.TIMES\_ROMAN, 12, Font.BOLD);

private static final String [] listOfTitles = {"Period", "Addition", "Withdrawal", "Reinvesting", "Deposit at the beginning of the month", "Deposit at the end of the month"};

public static void createReport(boolean typeOfReport, ArrayList<company> Companies, company Company) {

try {

namePDFReport = companies.input("Введите название отчета", "Введите название отчета");

FILE = companies.get\_current\_dir()+namePDFReport+".pdf";

Document document = new Document();

PdfWriter.getInstance(document, new FileOutputStream(FILE));

document.open();

if(typeOfReport){

createFullReport(document,Companies);

}else{

createUnfullReport(document,Company);

}

document.close();

}catch(FileNotFoundException | DocumentException e) {}

}

public static void createFullReport(Document document,ArrayList<company> Companies){

Companies.stream().forEach((Company) -> {

createUnfullReport(document, Company);

});

}

public static void createUnfullReport(Document document,company Company){

try {

companies.log(Company.get\_name());

Paragraph paragraph0 = new Paragraph(" ",CATFONT);

document.add(paragraph0);

Paragraph paragraph1 = new Paragraph(

"Name of company: " + Company.get\_name() +

", Deposit: " + Company.get\_depo() +

", Average percentage: " + Company.get\_persent()+ "%" +

", During of investment: " + Company.get\_period()

, SUBFONT);

paragraph1.setAlignment(Element.ALIGN\_LEFT);

document.add(paragraph1);

Paragraph paragraph2 = new Paragraph(" ",CATFONT);

document.add(paragraph2);

String [] arrayOfFields = Company.get\_company\_to\_array\_in\_report();

companies.log(arrayOfFields.length);

document.add(createTable(listOfTitles,arrayOfFields));

} catch (DocumentException ex) {

Logger.getLogger(createPDFReport.class.getName()).log(Level.SEVERE, null, ex);

}

}

public static PdfPTable createTable(String [] headers, String [] fields) throws BadElementException, DocumentException {

PdfPTable table = new PdfPTable(headers.length);

PdfPCell c1;

for(String head : headers){

c1 = new PdfPCell(new Phrase(head));

c1.setHorizontalAlignment(Element.ALIGN\_LEFT);

table.addCell(c1);

}

for(String field : fields){

table.addCell(field);

}

return table;

}

public static void addEmptyLine(Paragraph paragraph, int number) {

for (int i = 0; i < number; i++) {

paragraph.add(new Paragraph(" "));

}

}

}

Клас selectedCompany

package client;

import javax.swing.table.DefaultTableModel;

public class selectedCompany extends javax.swing.JFrame {

company c;

public selectedCompany() {

initComponents();

}

@SuppressWarnings("unchecked")

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

private void initComponents() {

jScrollPane1 = new javax.swing.JScrollPane();

sct = new javax.swing.JTable();

jButton1 = new javax.swing.JButton();

jButton2 = new javax.swing.JButton();

jButton3 = new javax.swing.JButton();

jButton4 = new javax.swing.JButton();

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

new Object [][] {

},

new String [] {

"Период", "Довложения", "Снятие", "Реинвестирование", "Депозит на начало месяца", "Депозит на конец месяца"

}

) {

Class[] types = new Class [] {

java.lang.Object.class, java.lang.Object.class, java.lang.Object.class, java.lang.Boolean.class, java.lang.Object.class, java.lang.Object.class

};

boolean[] canEdit = new boolean [] {

false, true, true, true, false, false

};

public Class getColumnClass(int columnIndex) {

return types [columnIndex];

}

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

jScrollPane1.setViewportView(sct);

if (sct.getColumnModel().getColumnCount() > 0) {

sct.getColumnModel().getColumn(0).setMaxWidth(110);

sct.getColumnModel().getColumn(1).setMaxWidth(130);

sct.getColumnModel().getColumn(2).setMaxWidth(70);

sct.getColumnModel().getColumn(4).setMinWidth(90);

sct.getColumnModel().getColumn(5).setMinWidth(90);

}

jButton1.setText("Сохранить");

jButton1.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

jButton1MouseClicked(evt);

}

});

jButton2.setText("Назад");

jButton3.setText("Расчитать прибыль");

jButton3.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

jButton3MouseClicked(evt);

}

});

jButton4.setText("Создать отчет");

jButton4.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

jButton4MouseClicked(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()

.addContainerGap()

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

.addGroup(layout.createSequentialGroup()

.addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT\_SIZE, 645, Short.MAX\_VALUE)

.addContainerGap())

.addGroup(layout.createSequentialGroup()

.addComponent(jButton2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton4)

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

.addComponent(jButton1)

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

.addComponent(jButton3)

.addGap(6, 6, 6))))

);

layout.setVerticalGroup(

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

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT\_SIZE, 261, Short.MAX\_VALUE)

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

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

.addComponent(jButton1)

.addComponent(jButton3)

.addComponent(jButton2)

.addComponent(jButton4))

.addContainerGap())

);

pack();

}// </editor-fold>

private void jButton3MouseClicked(java.awt.event.MouseEvent evt) {

showSelectedCompany();

}

company calculate(){

return c.getCalculateCompany();

}

private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {

//Save company

if (!checkTableValues()){

return ;

}

for(int i=0;i<sct.getRowCount();i++){

c.set\_plus(i, sct.getValueAt(i, 1).toString());

c.set\_minus(i, sct.getValueAt(i, 2).toString());

c.set\_reinvesting(i, sct.getValueAt(i, 3).toString());

}

companies.set\_updated\_company(c);

}

private void jButton4MouseClicked(java.awt.event.MouseEvent evt) {

createPDFReport.createReport(false, null, c); // неполный отчет

}

void setSelectedCompany(company c) {

this.c = c;

showSelectedCompany();

}

protected void showSelectedCompany(){

c.getCalculateCompany();

((DefaultTableModel)sct.getModel()).setNumRows(0);

int sizeOfExisting = c.get\_reinvesting().size();

for(int i=0;i<Integer.parseInt(c.get\_period());i++){

if(i >= sizeOfExisting){

c.set\_plus("0");

c.set\_minus("0");

c.set\_reinvesting("false");

}

((DefaultTableModel)sct.getModel()).addRow(

new Object[]{Integer.toString(i),

c.get\_plus(i),

c.get\_minus(i),

c.get\_reinvesting(i),

c.get\_depofirst(i),

c.get\_depolast(i)

});

}

}

protected boolean checkTableValues(){

boolean checked = true;

for(int i=0;i<sct.getRowCount();i++){

if (!isDouble(sct.getValueAt(i, 1).toString())){

companies.show("Вы ввели неправильно значение в таблице."

+ " Столбец довложений, период:"+ i);

return false;

}

if (!isDouble(sct.getValueAt(i, 2).toString())){

companies.show("Вы ввели неправильно значение в таблице."

+ " Столбец снятий, период:"+ i);

return false;

}

}

return checked;

}

public static boolean isDouble(String s) {

boolean isValidBoolean = false;

try{

Double.parseDouble(s);

isValidBoolean = true;

}catch (NumberFormatException ex){}

return isValidBoolean;

}

static void log(Object o){

System.out.println(o);

}

public static void main(String args[]) {

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new selectedCompany().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;

private javax.swing.JButton jButton3;

private javax.swing.JButton jButton4;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JTable sct;

// End of variables declaration

}

Клас database

package server;

import java.sql.\*;

import java.sql.Connection;

import java.util.ArrayList;

import java.util.Iterator;

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+"';");

}

protected void insertUser(User u) throws SQLException{

int i = 0;ResultSet rows = getUserByImail(u.get\_email());

while(rows.next()){++i;}

if(i == 0){

String prstmt= "INSERT INTO users (name,secondname,tel,email) "

+ "VALUES(?,?,?,?);";

PreparedStatement stmt = conn.prepareStatement(prstmt);

stmt.setString(1, u.get\_name());

stmt.setString(2, u.get\_secondname());

stmt.setString(3, u.get\_telephone());

stmt.setString(4, u.get\_email());

stmt.executeUpdate();

rows.close();

stmt.close();

}

}

protected void insertCompanyData(client.companiesToSent companies, String email) throws SQLException{

Iterator i = companies.getCampanyIterator();

String queryDelete = "DELETE FROM CLIENTS\_AND\_COMPANIES WHERE email='"+email+"';";

PreparedStatement stmt1 = conn.prepareStatement(queryDelete);

stmt1.execute();

queryDelete = "DELETE FROM COMPANIES WHERE email='"+email+"';";

stmt1 = conn.prepareStatement(queryDelete);

stmt1.execute();

stmt1.close();

while(i.hasNext()){

client.company c = (client.company) i.next();

String prstmt= "INSERT INTO CLIENTS\_AND\_COMPANIES (company\_name,percents,period,email,depo) VALUES(?,?,?,?,?);";

PreparedStatement stmt = conn.prepareStatement(prstmt);

stmt.setString(1, c.get\_name());

stmt.setString(2, c.get\_persent());

stmt.setString(3, c.get\_period());

stmt.setString(4, email);

stmt.setString(5, c.get\_depo());

stmt.executeUpdate();

for(int ind=0;ind<Integer.parseInt(c.get\_period());ind++){

String st= "INSERT INTO COMPANIES (period,plus,minuc,reinvesting,company\_name,email) VALUES(?,?,?,?,?,?);";

PreparedStatement s = conn.prepareStatement(st);

s.setString(1, String.valueOf(ind));

s.setString(2, c.get\_plus(ind));

s.setString(3, c.get\_minus(ind));

s.setString(4, String.valueOf(c.get\_reinvesting(ind)));

s.setString(5, c.get\_name());

s.setString(6, email);

s.executeUpdate();

}

}

}

protected client.companiesToSent getObjectToRestore(String email) throws SQLException{

client.companiesToSent c = new client.companiesToSent();

Statement stmt = conn.createStatement();

ResultSet r = stmt.executeQuery("Select \* from CLIENTS\_AND\_COMPANIES where email='"+email+"';");

//stmt.close();

while(r.next()){

Statement stmt1 = conn.createStatement();

ResultSet resData = stmt1.executeQuery("Select \* from COMPANIES where email='"+email

+"'AND company\_name='"+r.getString("company\_name")+"';");

ArrayList<String> plus=new ArrayList<>();

ArrayList<String> minus=new ArrayList<>();

ArrayList<String> reinvesting=new ArrayList<>();

while(resData.next()){

plus.add(resData.getString("plus"));

minus.add(resData.getString("minuc"));

reinvesting.add(resData.getString("reinvesting"));

}

client.company company = new client.company(r.getString("company\_name"),

r.getString("depo"),r.getString("percents"),r.getString("period"),plus,minus,reinvesting);

c.companiesToSend.add(company);

}

//r.close();

return c;

}

}

Клас emailSending

package server;

import java.io.UnsupportedEncodingException;

import java.util.Properties;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.mail.Message;

import javax.mail.MessagingException;

import javax.mail.Session;

import javax.mail.Transport;

import javax.mail.internet.AddressException;

import javax.mail.internet.InternetAddress;

import javax.mail.internet.MimeMessage;

public class EmailSending {

static Session session;

EmailSending(){

Properties props = new Properties();

props.put("mail.smtp.host","smtp.gmail.com");

props.put("mail.smtp.sosketFactory.port","465");

props.put("mail.smtp.socketFactory.class","javax.net.ssl.SSLSocketFactory");

props.put("mail.smtp.auth","true");

props.put("mail.smtp.port","465");

session = Session.getInstance(props,new javax.mail.Authenticator() {

@Override

protected javax.mail.PasswordAuthentication getPasswordAuthentication() {

return new javax.mail.PasswordAuthentication("skyliner270594@gmail.com", "1062543zz");

}

});

}

public static void sendEmail(String to, String from, String fromName, String subject, String text) {

try {

Message msg = new MimeMessage(session);

msg.setFrom(new InternetAddress(from, fromName));

msg.setRecipients(Message.RecipientType.TO,InternetAddress.parse(to));

msg.setSubject(subject);

msg.setText(text);

Transport.send(msg);

} catch (AddressException ex) {

Logger.getLogger(EmailSending.class.getName()).log(Level.SEVERE, null, ex);

} catch (MessagingException | UnsupportedEncodingException ex){

Logger.getLogger(EmailSending.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

Клас listOfUser

package server;

import java.io.IOException;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.JOptionPane;

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();

}

protected void set\_online(String email, boolean flag){

buf\_u.stream().filter((u) -> (email.equals(u.get\_email()))).forEach((u) -> {

u.set\_online(flag);

});

refresh\_list();

}

public final void getAllUsers() throws SQLException{

//buf\_u.clear();

ResultSet rs = db.getAllUsers();

while(rs.next()){

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

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

buf\_u.add(c);

}

}

public 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++){

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

return true;

}

}

return false;

}

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 сообщение", "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 сообщение":{

int [] selectedUsers = tcl.getSelectedRows();

if(selectedUsers.length == 0 ){

JOptionPane.showMessageDialog(rootPane,"Вы не выбрали клиентов для отправки сообщения");

}else{

String subject = JOptionPane.showInputDialog("Введите заголовок");

String message = JOptionPane.showInputDialog("Введите сообщение");

EmailSending sending = new EmailSending();

for(int i : selectedUsers){

sending.sendEmail(tcl.getValueAt(i, 4).toString(), "skyliner270594@gmail.com", "Petr Bulsyuk", subject, message);

}

JOptionPane.showMessageDialog(rootPane,"Сообщения успешно отправлены");

}

}; break;

case "Выделить всех":{

tcl.selectAll();

};break;

default:break;

}

}

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

}

void add\_new\_user(User u){

try {

db.insertUser(u);

} catch (SQLException ex) {

Logger.getLogger(ListOfUsers.class.getName()).log(Level.SEVERE, null, ex);

}

}

// Variables declaration - do not modify

private javax.swing.JComboBox<String> choiseMade;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JButton made;

private javax.swing.JTable tcl;

// End of variables declaration

}

Клас server

package server;

import java.io.\*;

import java.net.\*;

import java.sql.SQLException;

import java.text.SimpleDateFormat;

import java.util.\*;

import java.util.logging.Level;

import java.util.logging.Logger;

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;

private static Database db;

public Server(int port, ServerForm sg) {

try {

this.sg = sg;

this.port = port;

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

al = new ArrayList<>();

db = new Database();

} catch (ClassNotFoundException | SQLException | InstantiationException | IllegalAccessException ex) {

}

}

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,SAVE\_DATA=5,RESTORE\_DATA=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.");

lu.set\_online(ct.get\_email(),true);

}else{

//sendObject(ACCESS+";"+false);

//ct.close();

display("Новый клиент "+ct.get\_name()+" "+ct.get\_secondname()+" добавился.");}

lu.add\_new\_user(ct.get\_user());

//lu.getAllUsers();

lu.refresh\_list();

};break;

case QUIT:{

lu.set\_online(ct.get\_email(),false);

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

ct.close();

};break;

case SAVE\_DATA:{

String email = mes[1];

display("Клиент "+email+" запросил сохранение данных.");

new ServerThreadToSaveData(email).start();

display("Данные сохранены.");

};break;

case RESTORE\_DATA:{

String email = mes[1];

display("Клиент "+email+" запросил восстановление данных.");

new SendDataToClient(email).start();

display("Данные высланы.");

};break;

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

}

}

public class SendDataToClient extends Thread {

String email = "";

SendDataToClient(String e){

this.email = e;

}

@Override

public void run(){

try {

Socket socket1;

int portNumber = 1777;

String str = "";

System.out.println(email);

socket1 = new Socket(InetAddress.getLocalHost(), portNumber);

ObjectInputStream oisS = new ObjectInputStream(socket1.getInputStream());

ObjectOutputStream oosS = new ObjectOutputStream(socket1.getOutputStream());

oosS.writeObject(db.getObjectToRestore(email));

while ((str = (String) oisS.readObject()) != null) {

System.out.println(str);

oosS.writeObject("bye");

if (str.equals("bye"))

break;

}

oisS.close();

oosS.close();

socket1.close();

} catch (IOException | ClassNotFoundException ex) {

} catch (Exception ex) {

Logger.getLogger(Server.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

}

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");

}

// for a client who logoff

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;}

}

}

static class ServerThreadToSaveData extends Thread{

String email="";

ServerThreadToSaveData(String e){

this.email = e;

}

public void run() {

try {

ServerSocket servSocket;

Socket fromClientSocket;

int cTosPortNumber = 1777;

String str1;

client.companiesToSent comp;

servSocket = new ServerSocket(cTosPortNumber);

System.out.println("Waiting for a connection on " + cTosPortNumber);

fromClientSocket = servSocket.accept();

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

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

while ((comp = (client.companiesToSent) ois.readObject()) != null) {

db.insertCompanyData(comp, email);

oos.writeObject("bye bye");

break;

}

oos.close();

ois.close();

fromClientSocket.close();

servSocket.close();

} catch (IOException | ClassNotFoundException ex) {

} catch (SQLException ex) {

Logger.getLogger(Server.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

class ClientThread extends Thread {

// the socket where to listen/talk

protected Socket socket;

private int id;

protected User user;

private ObjectMessenger om;

ClientThread(Socket socket) throws Exception {

// a unique id

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) {}

}

// what will run forever

@Override

public void run() {

// to loop until LOGOUT

boolean keepGoing = true;

while(keepGoing) {

// read a String

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\_telephone(){return user.get\_telephone();}

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

// try to close everything

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);

}

protected User get\_user(){

return user;

}

}

}

Клас serverForm

package server;

import java.io.IOException;

import java.sql.SQLException;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.xml.parsers.ParserConfigurationException;

import org.xml.sax.SAXException;

public class ServerForm extends javax.swing.JFrame {

private Server s;

public static Database db;

private static final int port = 7777;

private static Server serv;

private static ListOfUsers lu;

public ServerForm(){

initComponents();

}

@SuppressWarnings("unchecked")

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

private void initComponents() {

jScrollPane1 = new javax.swing.JScrollPane();

log = new javax.swing.JTextArea();

start = new javax.swing.JButton();

stop = new javax.swing.JButton();

clientList = new javax.swing.JButton();

administration = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setTitle("Сервер");

log.setColumns(20);

log.setRows(5);

jScrollPane1.setViewportView(log);

start.setText("start");

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

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

startActionPerformed(evt);

}

});

stop.setText("stop");

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

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

stopActionPerformed(evt);

}

});

clientList.setText("list of users");

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

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

clientListActionPerformed(evt);

}

});

administration.setText("administration");

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

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

administrationActionPerformed(evt);

}

});

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

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

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

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addContainerGap()

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

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

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

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

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

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

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 350, javax.swing.GroupLayout.PREFERRED\_SIZE))

);

layout.setVerticalGroup(

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

.addComponent(jScrollPane1, javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(start)

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

.addComponent(stop)

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

.addComponent(clientList)

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

.addComponent(administration)

.addContainerGap(179, Short.MAX\_VALUE))

);

pack();

setLocationRelativeTo(null);

}// </editor-fold>

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

new ServerRunning().start();

}

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

try {

serv.stop();

} catch (IOException ex) {}

}

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

lu.setVisible(true);

lu.refresh\_list();

}

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

}

class ServerRunning extends Thread {

public void run() {

try {serv.start();

} catch (Exception ex) {}

}

}

public static void main(String args[]) {

java.awt.EventQueue.invokeLater(new Runnable() {

@Override

public void run() {

ServerForm s = new ServerForm();

s.setVisible(true);

serv = new Server(port, s);

try{

db = new Database();

lu=new ListOfUsers(db);

serv.set\_list\_user(lu);

lu.setVisible(false);

} catch (ParserConfigurationException | SAXException | IOException ex) {

System.out.println(ex);

} catch (SQLException | ClassNotFoundException | InstantiationException | IllegalAccessException ex) {

Logger.getLogger(ServerForm.class.getName()).log(Level.SEVERE, null, ex);

}

}

});

}

void appendEvent(String string) {

log.append(string);

}

// Variables declaration - do not modify

private javax.swing.JButton administration;

private javax.swing.JButton clientList;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JTextArea log;

private javax.swing.JButton start;

private javax.swing.JButton stop;

// End of variables declaration

}

Клас user

package server;

public class User {

private String name,secondname,tel,email;

private boolean online;

protected int item\_number;

User(boolean online,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 String get\_telephone(){return this.tel;}

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

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;

}

}