

ІНФОРМАТИКА, ОБЧИСЛЮВАЛЬНА ТЕХНІКА ТА АВТОМАТИЗАЦІЯ

УДК 004.4+336

DOI <https://doi.org/10.32838/2663-5941/2019.5-1/08>

Антонюк Д.С.

Державний університет «Житомирська політехніка»

Вакалюк Т.А.

Державний університет «Житомирська політехніка»

Якобчук А.С.

Державний університет «Житомирська політехніка»

Янчук В.М.

Державний університет «Житомирська політехніка»

ПРОЕКТУВАННЯ ПРОГРАМНО-ІМІТАЦІЙНОГО КОМПЛЕКСУ ВІЗУАЛІЗАЦІЇ ТА УПРАВЛІННЯ ПЕРСОНАЛЬНИМИ ФІНАНСАМИ

У статті описано процес розробки програмно-імітаційного комплексу візуалізації та управління персональними фінансами. Висока динамічність соціально-економічних систем, що притаманна сучасному суспільству та міжнародній економіці вимагає від спеціалістів різних професій наявності теоретичних знань та практичних умінь у галузях економічної теорії, фінансів та управління. Зручний комплекс, спрямований на візуалізацію індивідуальних фінансових можливостей дає змогу зацікавити та наочно представити спроможність людини в економічних відносинах. Функціональні можливості системи дозволять застосовувати її для особистих економічних цілей та саморозвитку.

Визначено функціональні та нефункціональні вимоги до даного засобу. Описано модулі та класи, що були ключовими при розробці програмно-імітаційного комплексу. Представлено діаграми активності ППК за різних умов. Розроблено слабозв'язану архітектуру додатку та визначено порядок взаємодії класів в процесі роботи програми; реалізовано простий інтерфейс, що дозволяє швидко навчитись користуватись додатком та максимально інформативно візуалізувати економічні можливості користувача.

Користувачі цього сервісу мають доступ до наступних функцій: візуалізації фінансових можливостей користувача; перегляду прикладів економічних планувань; зміни поточного віку, віку виходу на пенсію та тривалості життя; редагування очікуваних щомісячних витрат на пенсії; редагування очікуваних щомісячних прибутків на пенсії; можливості зміни типу щомісячних збережень від фіксованої суми до відсотків від заробітної плати; можливості зміни заробітної плати з певного віку; можливості здобуття освітнього рівня в певному віці; мати змогу додавати додаткові надходження у вигляді довгострокових депозитів або ж одноразових продажів майна; мати змогу додавати додаткові витрати у вигляді кредитів або ж купівлі майна; мати змогу обирати валюту для додаткових витрат або надходжень.

Ключові слова: програмно-імітаційні комплекси, візуалізація, управління, персональні фінанси, розробка.

Постановка проблеми. Знання класичних економічних теорій, історичних закономірностей та обізнаність в основах поведінкової економіки та ірраціональної складової прийняття індивідуальних та колективних рішень дають людині змогу приймати відповідальні рішення та забезпечувати стабільність та адаптивність як особисту, так і сімейну, а також – інституціональну [8].

Зручний комплекс, спрямований на візуалізацію індивідуальних фінансових можливостей дає змогу зацікавити та наочно представити спроможність людини до економічних перспектив. Функціональні можливості системи дозволять застосовувати її для особистих економічних цілей та саморозвитку.

Висока динамічність соціально-економічних систем, що притаманна сучасному суспільству та

міжнародній економіці вимагає від спеціалістів різних професій наявності теоретичних знань та практичних умінь у галузях економічної теорії, фінансів та управління [8].

Аналіз останніх досліджень і публікацій. Застосування програмно-імітаційних комплексів у різних сферах життєдіяльності розглядали В. Адамс (W.K. Adams) [1], Е. Кастронова (E. Castronova) [3], О.О. Мацюк [11], В.А. Пермінова [13], С. Фортман-Рое (S. Fortmann-Roe) [4], О.Б. Шендерук [13] та інші.

Зокрема, П.Г. Банщиків та В.Я. Паздрій описують досвід проведення занять і змагань із застосуванням програмно-імітаційних комплексів (ПК) “Sigam-Market” та “ViAL+” [9; 10], С.О. Мотуз розглядає можливості застосування ПК у процесі проведення тренінгів з біржових торгів [12], К. Елдріч розглядає можливості використання ПК в галузі освіти, а також проблеми добору і побудови симуляцій, поєднання освітніх цілей з ігровими механіками, типи симуляцій і перспективи розвитку сфери використання ПК в освіті [2]. С. Фортмана-Рое розглядає інструменту побудови моделей та симуляцій над ними InsightMaker [4]. Цей програмний комплекс дозволяє будувати моделі економічних систем та явищ, а також проводити симуляцію поведінки економічних систем у часі.

Формулювання цілей статті. Метою роботи є розробка програмно-імітаційного комплексу з управління персональними фінансами.

Виклад основного матеріалу дослідження. Головним завданням реалізації web-орієнтованої системи «Програмно-імітаційний комплекс візуалізації та управління персональних фінансів» є підвищення компетентності окремих людей в плануванні персональних фінансів.

Програми для персонального бюджету не можуть якимось чином вплинути на ваш дохід, це те, що більшою мірою залежить від самої людини. Кращий вибір програмного забезпечення може зберегти час і вирішити проблеми бюджету. Також такі додатки допоможуть створювати та дотримуватись певного бюджетного планування, а багато з них матимуть додаткові корисні функції залежно від потреб [7].

При реалізації «Програмно-імітаційного комплексу візуалізації та управління персональних фінансів» необхідно врахувати позитивні можливості ПК з економічним спрямуванням. Основними характеристиками нового комплексу повинні бути: простий та зручний у використанні інтерфейс, в той же час в ньому має бути багато функцій для детальнішої симуляції.

Для реалізації веб-сервісу «Програмно-імітаційний комплекс візуалізації та управління персональних фінансів» було вирішено обрати архітектурний патерн MVC. MVC є одним з найбільш популярних патернів у веб-орієнтованій архітектурі, адже з його допомогою створюються гнучкі та функціональні додатки, які легко підтримувати та розширювати [5]. MVC складається з трьох частин, від яких і пішла його назва (рис. 1).

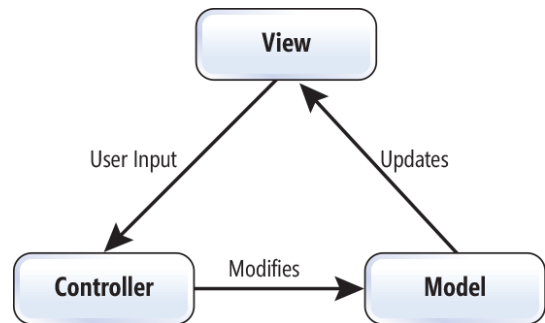


Рис. 1. Узагальнена архітектура MVC

Розглянемо їх особливості більш детально:

Модель-представлення-контролер – архітектурний шаблон, який використовується під час проектування та розробки програмного забезпечення. Цей шаблон передбачає розподіл системи на три пов’язані між собою частини: модель даних, вигляд (інтерфейс) та модуль керування. Шаблон використовується для відокремлення даних (моделі) від інтерфейсу користувача (вигляду) так, щоб зміни інтерфейсу користувача мінімально впливали на роботу з даними, а зміни в моделі даних могли здійснюватися без змін інтерфейсу користувача.

Мета використання шаблону – гнучкий дизайн програмного забезпечення, що полегшує подальші зміни чи розширення програми, а також є можливість повторно використовувати окремі компоненти програми. Окрім цього, використання шаблону у системах сприяє їх впорядкованості і робить їх більш зрозумілими за рахунок зменшення складності.

Також для розробки ПК було використано шаблон MVVM. MVVM застосовується для відокремлення моделі та її відображення. Це необхідно для надання можливості їх незалежної зміни. Наприклад, розробник працює над реалізацією логіки роботи з даними, а дизайнер – з користувацьким інтерфейсом [6].

Шаблон MVVM був створений з метою розділення роботи дизайнера і програміста. Дизайнери інтерфейсів краще знають, чого бажають користувачі, ніж розробники програмного коду. Тому буде

краще, коли дизайнер створить графічний інтерфейс, а програміст напише код, який реалізовуватиме логіку цього інтерфейсу.

Шаблон MVVM поділяється на три частини:

- модель, яка являє собою набір даних, що потрібні для роботи додатку;
- вид, який являє собою графічний користувацький інтерфейс, тобто вікно, кнопки і так далі;
- модель вигляду, з однієї сторони є абстракцією вигляду, а з іншої є обгорткою даних моделі, що має зв'язуватись. Тобто вона містить модель, яка перетворена до вигляду, а також містить у собі команди, якими може скористатися вигляд для впливу на модель. Фактично ViewModel призначена для того, щоб: здійснювати зв'язок між моделлю та виглядом, відслідковувати зміни в даних, що зроблені користувачем, відпрацьовувати логіку роботи вигляду.

Користувачі даного сервісу мають доступ до наступних функцій:

1. Візуалізації фінансових можливостей користувача.
2. Перегляду прикладів економічних плануваль.
3. Зміни поточного віку, віку виходу на пенсію та тривалості життя.
4. Редагування очікуваних щомісячних витрат на пенсії.
5. Редагування очікуваних щомісячних прибутків на пенсії.

6. Можливості зміни типу щомісячних збережень від фіксованої суми до відсотків від заробітної плати.

7. Можливості зміни заробітної плати з певного віку.

8. Можливості здобуття освітнього рівня в певному віці.

9. Мати змогу додавати додаткові надходження у вигляді довгострокових депозитів або ж одноразових продажів майна.

10. Мати змогу додавати додаткові витрати у вигляді кредитів або ж купівлі майна.

11. Мати змогу обирати валюту для додаткових витрат або надходжень.

Для програмно-імітаційного комплексу було створено діаграму прецедентів, що зображена на рис. 2.

Такий ПК має відповідати функціональним вимогам:

1. Можливість збереження інформації: система повинна зберігати введені користувачем дані і надавати можливість керувати ними.
2. Можливість перегляду прикладів економічного планування: системи повинна надавати користувачу можливість переглянути приклад візуалізації даних певних осіб.
3. Проведення аналітичних операцій: система повинна відображати інформацію відповідно до введених користувачем даних.
4. Можливість додавання додаткових надходжень та витрат: система повинна надати

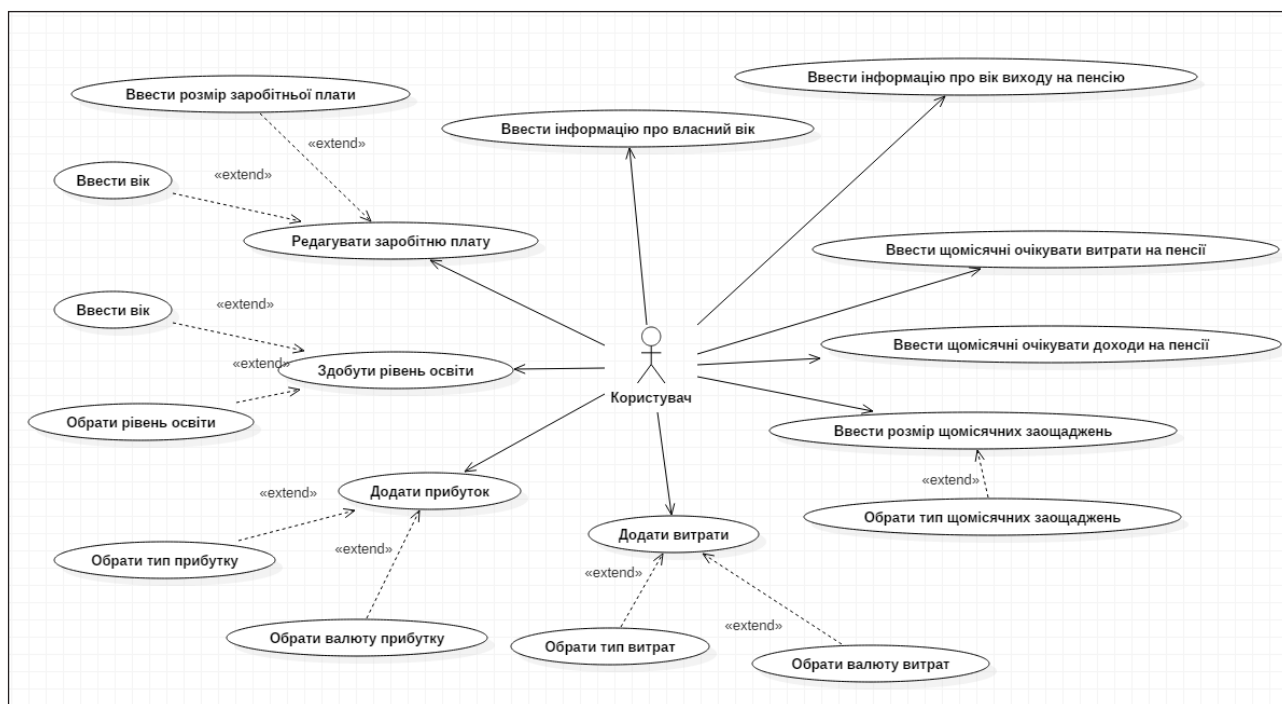


Рис. 2. Діаграма варіантів використання для актора «Користувач»

можливість додавання додаткових надходжень або витрат.

Та нефункціональним вимогам:

1. Сприйняття:
 - Час, потрібний для навчання роботи з системою для звичайних користувачів – 1 година, а для досвідчених – 30 хвилин.
 - Час відповіді системи для звичайних запитів не повинен перевищувати 2 секунди, а для більш складних запитів – 4 секунди.
 - Інтерфейс представлення системи має бути зручним для користувача та не потребувати від нього додаткової підготовки.
2. Надійність
 - Доступність – час, потрібний для обслуговування системи не повинен перевищувати 10% від загального часу роботи.
 - Середній час безперервної роботи – 15 робочих днів.
 - Максимальна норма помилок та дефектів в роботі системи – 1 помилка на 1000 запитів користувача.

3. Продуктивність

– Система повинна підтримувати мінімум 1000 одночасно працюючих користувачів.

4. Можливість експлуатації

– Масштабування – система повинна мати можливість збільшувати потужності (продуктивність) зі збільшенням кількості користувачів так, щоб це не відображалось негативно на її роботі.

Для реалізації проекту обрана технологія ASP.NETMVC та мова програмування C#.

Під час проектування було вирішено, що рішення буде складатись з двох проектів, які міс-

титимуть тематично згруповані класи, що забезпечуватимуть функціональні вимоги системи.

Збірка “PocketBudget” містить конфігурування системи та відповідає за рівень відображення (рис. 3). У збірці можна звернути увагу на наступні модулі:

- Dependency Injection – клас, призначений для конфігурування залежностей модулів.
- Personal Finances – клас, що надає доступ до реалізованих модулів.

Також ця збірка містить модулі мапінгу моделей, контролери, представлення, методи розширення, моделі представлення та Javascript скрипти.

На діаграмі класів зі збірки “PocketBudget” представлені наступні класи (рис. 3):

1. DependencyInjection – клас, в якому відбувається конфігурація залежностей модулів при запуску додатку.
2. AdditionalIncomeResolver – клас, призначений для мапінгу додаткових надходжень з моделі вигляду до моделі.
3. AdditionalCostResolver – клас, призначений для мапінгу додаткових витрат з моделі вигляду до моделі.
4. HomeController – контролер, в якому відбувається основна логіка роботи програми. Клас містить методи, що повертають модель PathViewModel зі змінним станом у результаті виконання операцій.
5. EducationController – контролер, який відповідає видачу правильних освітніх рівнів для кожної професії.
6. SalaryController – контролер, що відповідає за повернення поточних періодів заро-

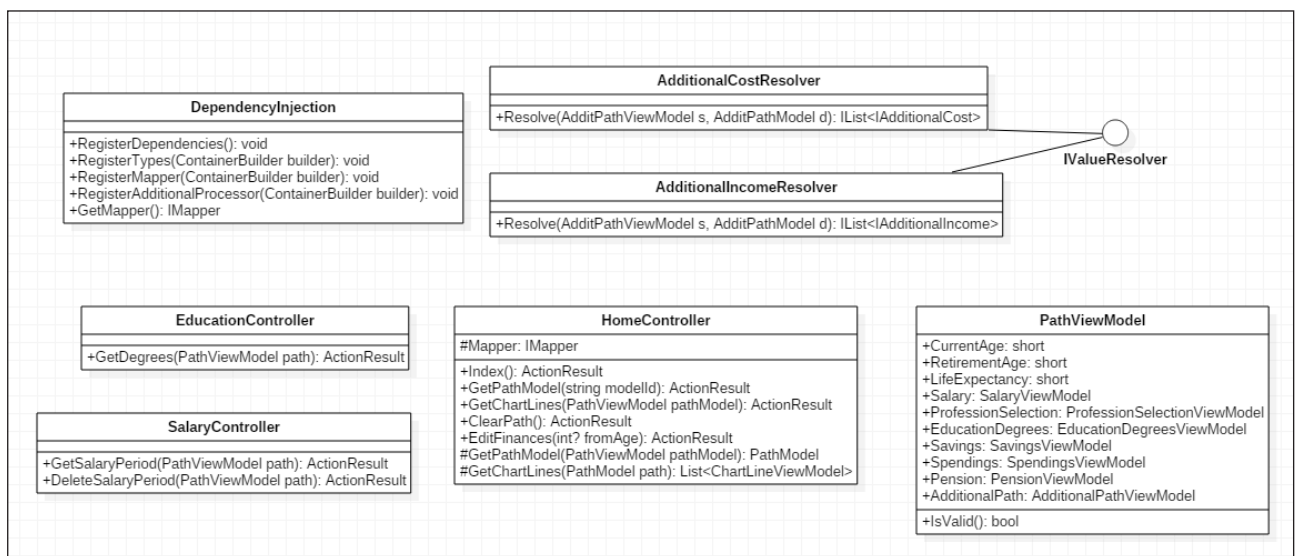


Рис. 3. Діаграма деяких класів зі збірки «PocketBudget»

бітньої плати та видалення періоду заробітної плати.

7. PathViewModel – клас, який є основною моделлю виглядом для візуалізації фінансових можливостей та містить в собі всю необхідну інформацію про введені користувачем дані.

Збірка “Business” містить у собі модулі для аналітичних розрахунків управління персональними фінансами та бізнес моделі сутностей системи (рис. 4):

1. Моделі, що є бізнес-сутностями: AdditionalPathModel, EducationModel, PathModel.

2. ChartManager – клас, що відповідає за повернення даних для візуалізації контролеру, реалізує інтерфейс IChartManager.

3. FixedSavingsStrategy – клас, що реалізовує метод базового абстрактного класу отримання суми заощаджень при фіксованому типі збережень.

4. PercentageSavingsStrategy – клас, що реалізовує метод базового абстрактного класу отримання суми заощаджень при відсотковому типі збережень.

5. AdditionalPathProcessor – клас, що відповідає для розрахунку параметрів додаткових надходжень та витрат.

6. SaleIncomeStep, DepositIncomeStep – класи, що відповідають за розрахунок розміру додаткових надходжень від продажів та депозитів відповідно.

7. PurchaseCostStep, CreditCostStep – класи, що відповідають за розрахунок роз-

міру додаткових витрат від купівлі та кредитів відповідно.

Програмно-імітаційний комплекс візуалізації та управління персональних фінансів використовує дані введені користувачем в реальному часі та проводить розрахунки враховуючи їх. Оскільки в системі непередбачена реєстрація та збереження всіх даних користувачів, було прийнято рішення використовувати XML файли для зберігання лише деяких даних. Основними модулями системи є модуль бізнес логіки, що відповідає за розрахунок та обробку даних економічних можливостей.

Після завантаження головної сторінки програмно-імітаційного комплексу (рис. 5) користувач може виконати наступні дії: переглянути приклади даних користувачів, редагувати поточний вік, вік виходу на пенсію та очікувану тривалість життя, ввести витрати та доходи на пенсії, щомісячні заощадження, рівень освіти та заробітну плату а також є можливість додати додаткові надходження або витрати.

Під час додавання додаткових надходжень або витрат користувач має обрати тип доходів (депозит, продаж) або витрат (кредит, купівля), ввести вік коли додаткові дані повинні бути враховані та ввести суму та валюту обраних надходжень або витрат. Введені дані буде провалідовано та відправиться запит на сервер. Результат операції відобразиться користувачеві.

Під час додавання заробітної плати (рис. 7) користувач натискає на кнопку для відображення нового

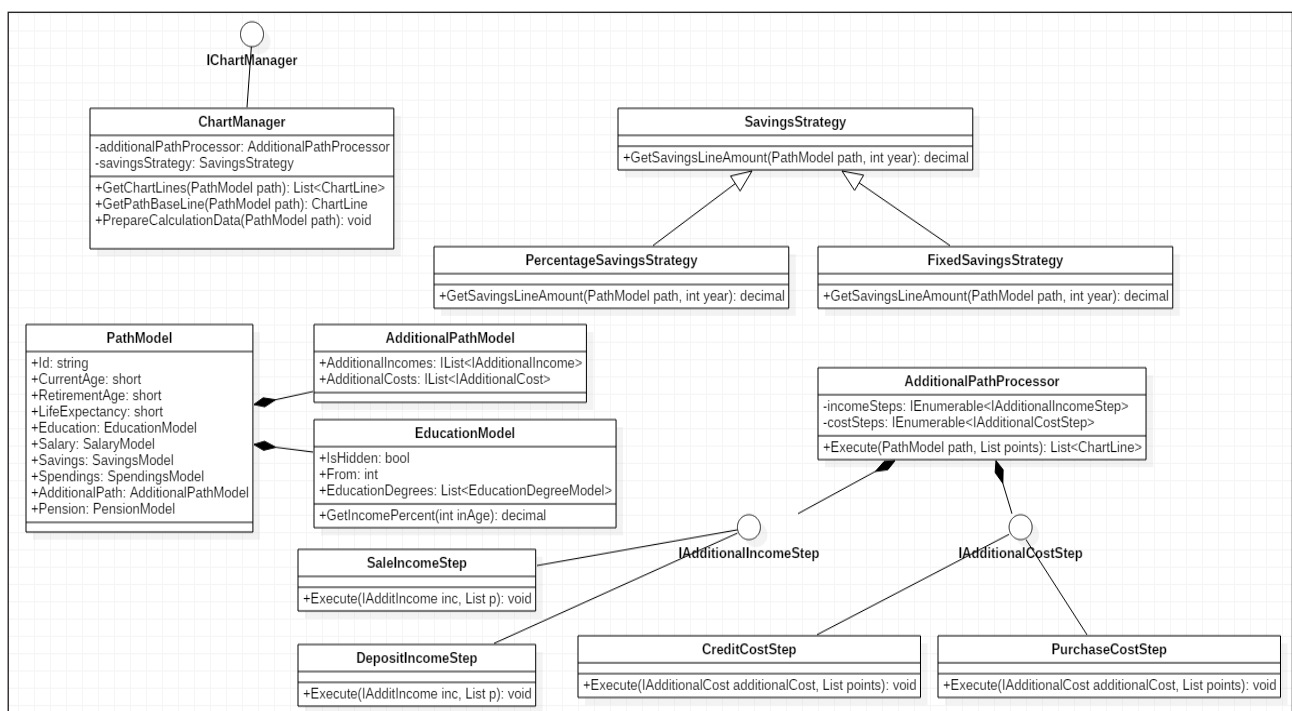


Рис. 4. Діаграма деяких класів збірки «Business»

періоду заробітної плати, для цього в систему йде запит для повернення необхідного періоду. Після отримання результату користувач вводить необхідні дані та зберігає зміни, система обробляє зміни з урахуванням всіх інших параметрів та повертає оновлену візуалізацію даних клієнту.

Для написання слабозв'язного коду було використано патерн програмування інверсія управління. Його суть полягає у написанні коду для інтерфейсів, а не до залежностей. Реалізація патерну виконується за допомогою бібліотеки Autofac. Бібліотека надає можливість налаштування залежностей використовуючи методи розширення:

- RegisterType.
- RegisterInstance.

Обидва методи пов'язують конкретну реалізацію з інтерфейсом, різниця полягає у тривалості збереження зареєстрованої сутності. Метод RegisterType створює новий екземпляр для кожного HTTP запити, тоді як RegisterInstance створює один екземпляр на весь час роботи додатку.

Варто розуміти, для правильної роботи патерну інверсія управління, всі залежності мають бути зареєстровані в корні додатку – місці, що знаходиться якомога ближче по стеку викликів до програмного коду, що ініціює додаток.

При реалізації сервісу потрібно стежити за розділенням різних слоїв додатку. У такому випадку всі сутності розділені на 2 види:

- Сутність бізнес логіки.
- Сутність візуалізації.

Розподілення допомагає в написанні слабозв'язаного коду, проте призводить до задач, які треба було вирішити. Зокрема, це задача централізованої та інкапсульованої передачі даних між об'єктами різних рівнів. Щоб вирішити цю задачу було використано бібліотеку AutoMapper, що надає можливість абстрагувати логіку передачі даних з однієї сутності іншій.

Також у додатку було реалізовано функціональність розрахунку щомісячних збережень за допомогою патерну Стратегія. Шаблон належить до шаблонів поведінки. Його ідея полягає в тому, щоб створити декілька схем поведінки одного об'єкту та винести це в окремий клас. За типом щомісячних збережень обирається алгоритм, за яким буде розраховуватись сума щомісячних збережень. Використання цього патерну дозволяє відмовитись від використання перемикачів або умовних операторів, але це призведе і до збільшення кількості об'єктів.

Для розгортання веб-сервісу потрібно виконати наступні кроки:

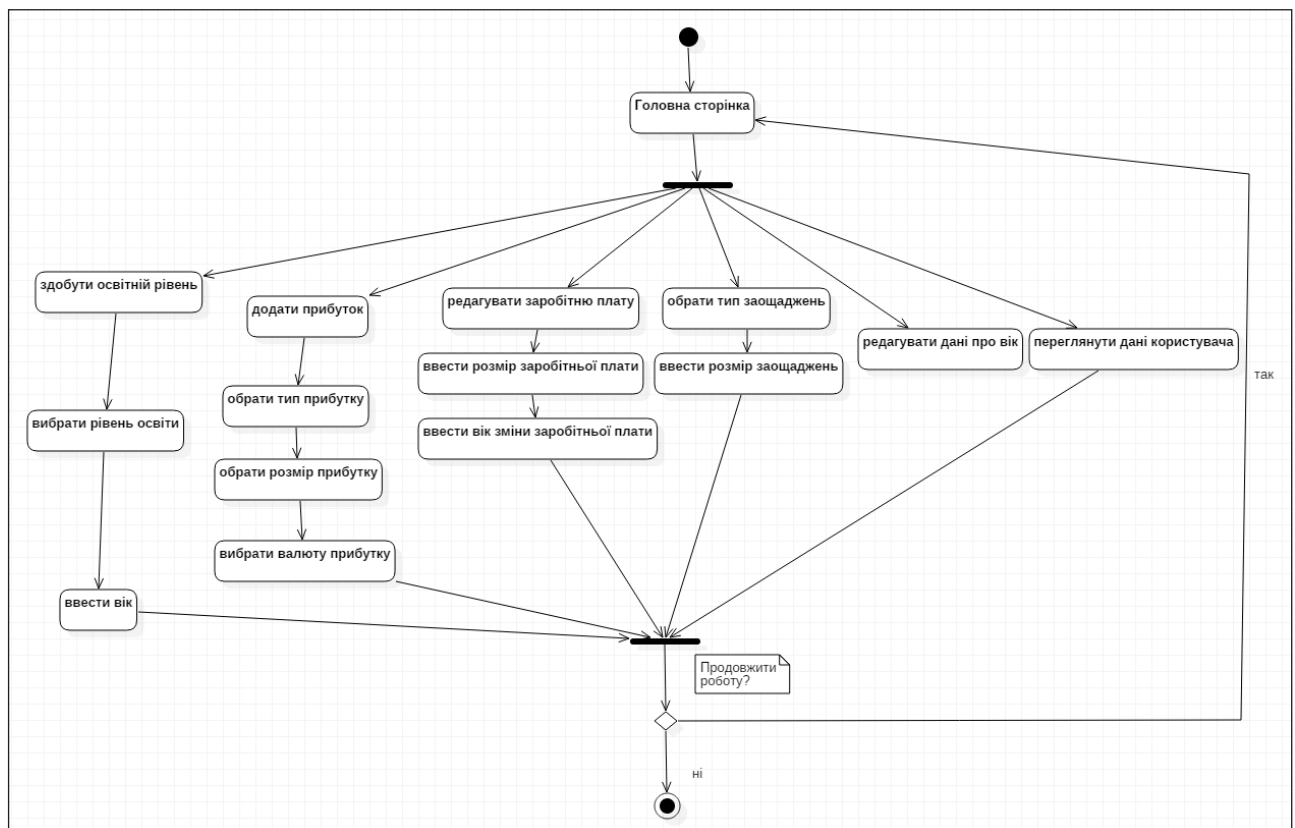


Рис. 5. Загальна діаграма активності

1. Створити пакет розгортання за допомогою технології MSBuild.
2. Створити пул та веб-сайт в IIS.
3. Пакет розгортання розмістити в папці, на яку сси лається веб-сайт.
4. Відновити дані прикладів користувачів в XML-файлі.
5. Запустити пул та веб-сайт в IIS.

При завантаженні головної сторінки користувач має змогу переглянути візуалізацію економічного шляху за замовчуванням. Візуалізація представлена у вигляді графіку, справа від якого наведено основні параметри, по яким було побудовано наведений графік. Знизу графіку наведений вік користувача та відповідний йому рік, зліва сума заощаджених грошей. Відповідно до початкових налаштувань, на графіку також можна спостерігати лінію, яка умовно показує час, коли користувач вийде на пенсію.

На головній сторінці користувач має змогу редагувати основну інформацію або ж додати

додаткові надходження чи витрати до свого фінансового планування. Натиснувши кнопку редагування, можна побачити форму редагування інформації про користувача та його фінансові плани.

Перед тим, як розпочати вводити власні дані, у користувача є можливість переглянути приклади економічних планувань умовних людей, натиснувши на відповідну кнопку в формі редагування інформації.

Після цього, для початку, введемо основні дані про вік користувача, вік виходу на пенсію, очікувану тривалість життя, витрати та доходи на пенсії та розмір щомісячних збережень до настання пенсійного віку.

Після введення основних даних та вибору типу щомісячних заощаджень «відсотки від зарплати» користувач може змінити свої дані про заробітну платню. Натиснувши на кнопку «редагувати», перед користувачем відобразиться додаткове меню з інформацією про його заробітну плату. В

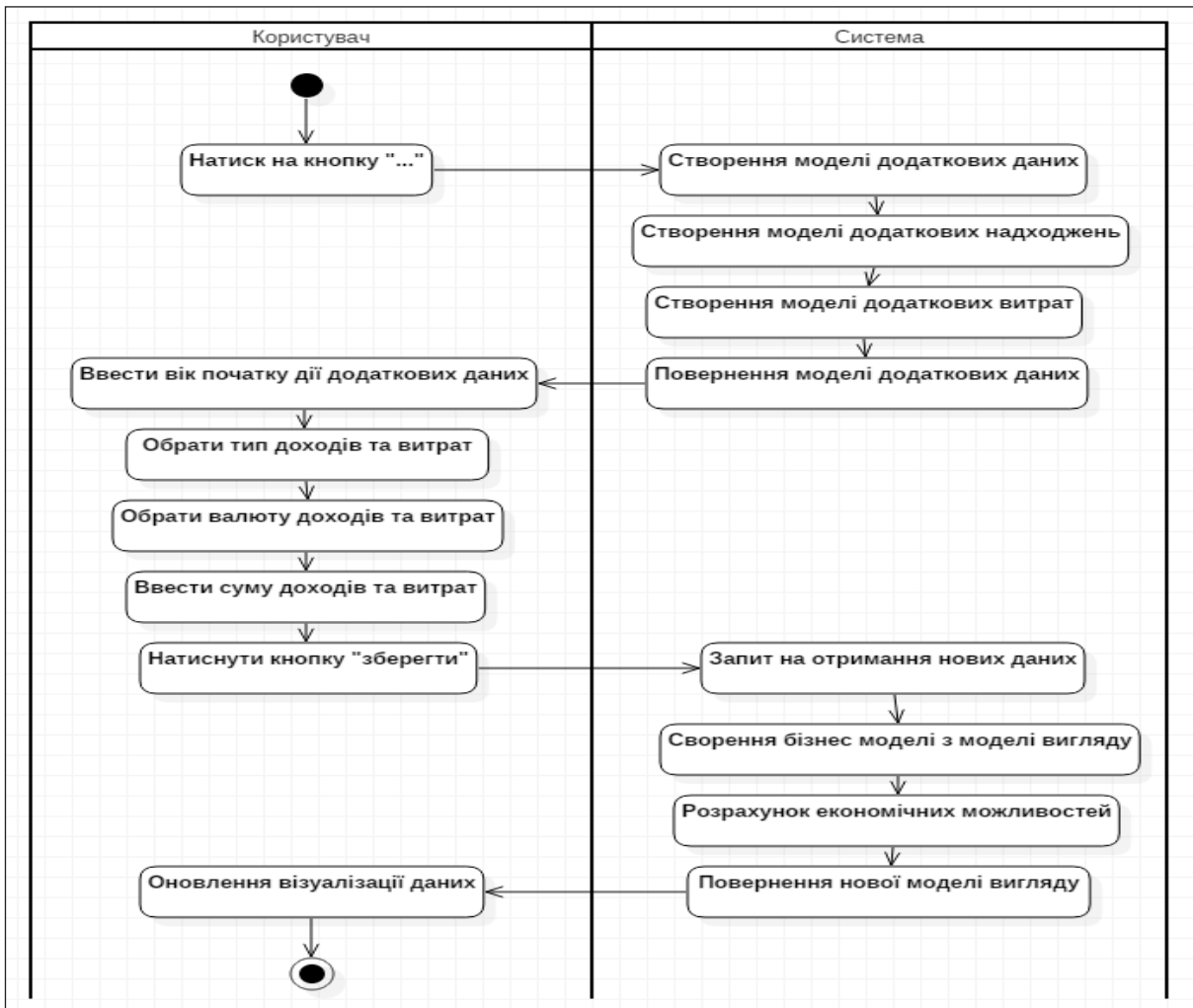


Рис. 6. Діаграма активності для прецеденту «додаткові надходження»

даному меню користувач може додати або видалити додатковий період його заробітної плати. Для того, щоб додати період, потрібно натиснути на відповідну кнопку в меню та заповнити поля періоду: вік на розмір заробітної плати. Після вводу даних необхідно зберегти редаговані дані, натиснувши відповідно кнопку.

Додавши новий період, основний графік зміниться, використовуючи зум на графіку можна побачити, як викривляється графік зі збільшенням зарплатні.

Навівши курсор на графік, можна побачити інформацію про вік користувача, рік та очікувану суму заощаджень та вибраний момент.

На інформаційному блоці, натиснувши на кнопку «...», перед користувачем відобразиться меню з додаванням додаткових надходжень або витрат. У меню користувач має змогу додати доходи у вигляді депозиту чи продажу майна та витрат у вигляді кредиту чи купівлі майна. Для додавання депозиту потрібно обрати валюту заощаджень та вказати суму, яку буде покладено на депозит.

Аналогічні поля потрібно заповнити при додаванні додаткових витрат.

Після введення інформації про додаткові надходження та витрати зберігаємо зміни. Відповідно до введених параметрів буде перерахований графік економічного планування користувача.

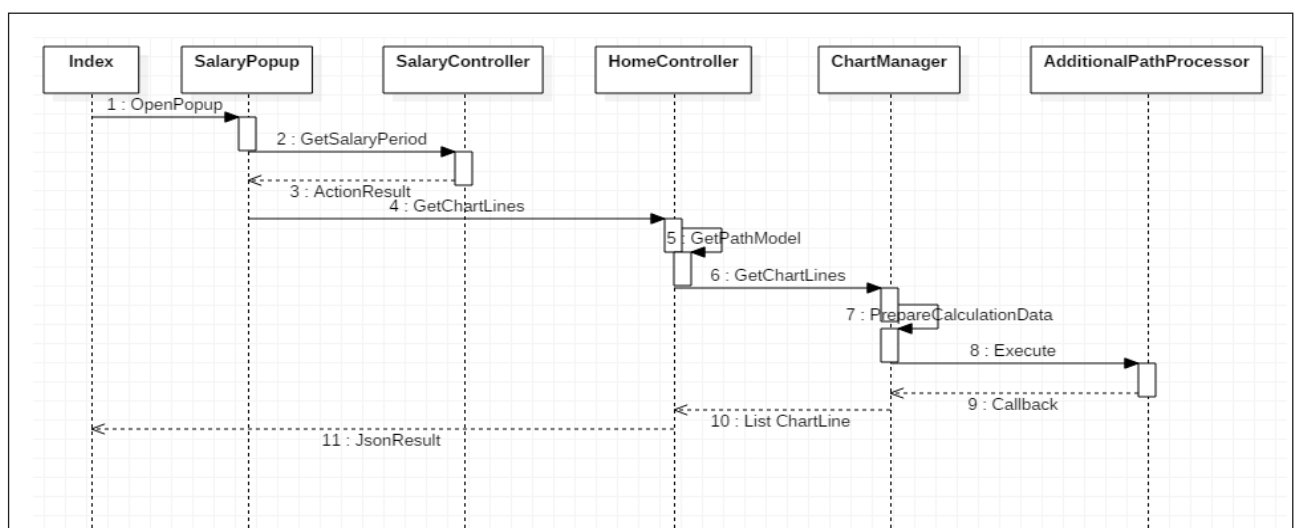


Рис. 7. Діаграма послідовності «Додавання періоду заробітної плати»

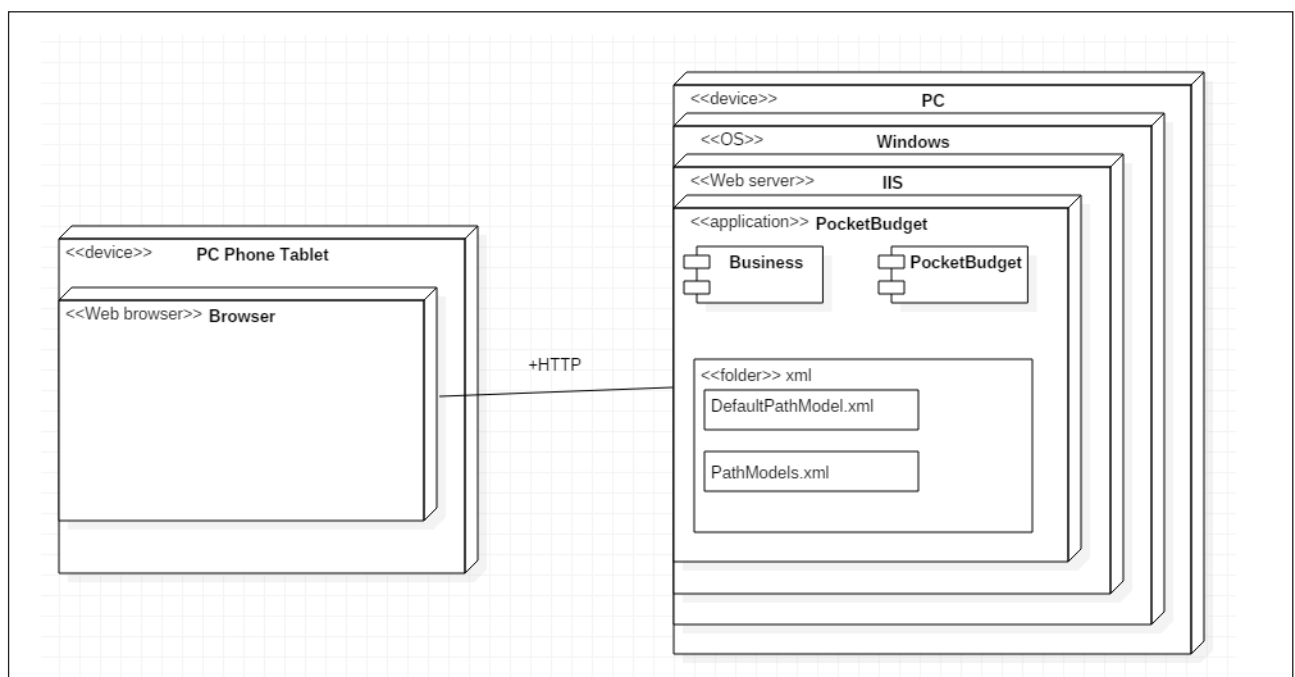


Рис. 8. Діаграма розгортання системи



Рис. 9. Введення базової інформації користувача

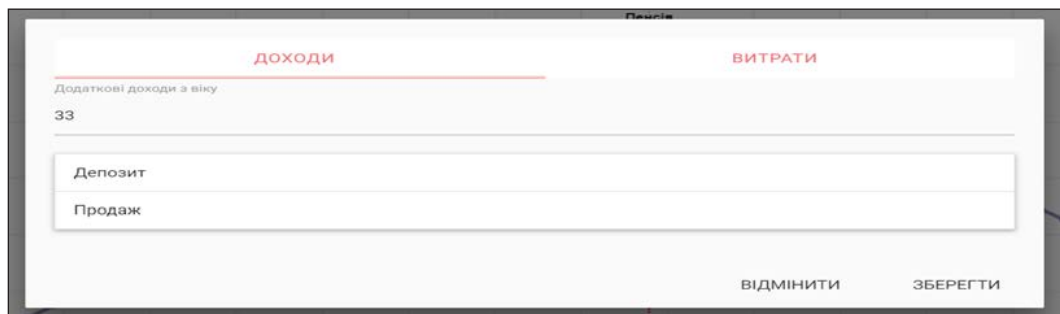


Рис. 10. Меню додавання додаткових надходжень чи витрат

Також користувач має змогу вимкати та вмикати зміни щодо додаткових надходжень чи витрат, натиснувши на відповідну назву над графіком, після чого графік буде перебудовано без врахування вимкнених параметрів.

Ще одним способом збільшити свій дохід для користувача є здобуття одного з освітніх рівнів. Кожний освітній рівень додає приріст до заробітної плати. Щоб здобути освітній рівень, користувачу потрібно натиснути «редагувати» напроти освіти, перед ним відобразиться меню з вводом необхідних даних, де потрібно буде обрати освітній рівень та ввести вік, коли його буде здобуто.

Натиснувши кнопку «зберегти» в меню додавання освітнього рівня, графік буде перебудовано з врахуванням здобутої освіти.

Висновки. У роботі було спроектовано та реалізовано «Програмно-імітаційний комплекс візуалізації та управління персональних фінансів: функціональність бізнес-логіки», для цього було: розроблено слабозв'язану архітектуру додатку та визначено порядок взаємодії класів в процесі роботи програми; реалізовано простий інтерфейс, що дозволяє швидко навчитись користуватись додатком та максимально інформативно візуалізувати економічні можливості користувача.

Реалізована система може застосовуватись для зручного планування власних фінансів.

У подальшому комплекс може бути вдосконалений шляхом додавання додаткового функціоналу для прогнозування, аналізу, рекомендацій щодо фінансового планування.

Депозит

Додати депозит

Депозит в ₴ 14.00% річних

100000

1

Депозит в \$ 3.75% річних

Депозит в € 2.35% річних

Рис. 11. Додавання депозиту

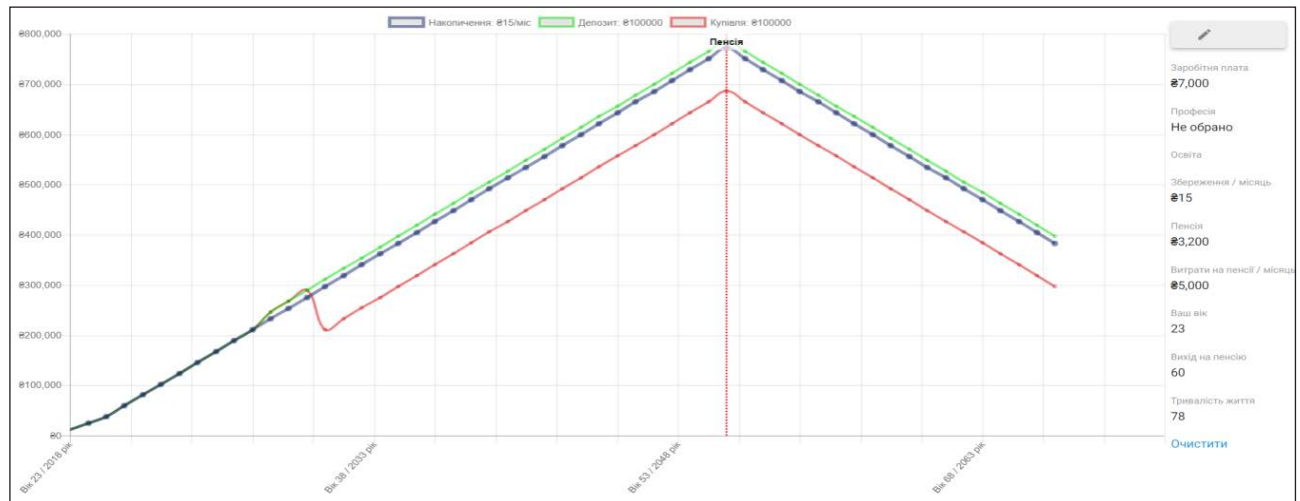


Рис. 12. Змінений графік економічного планування користувача

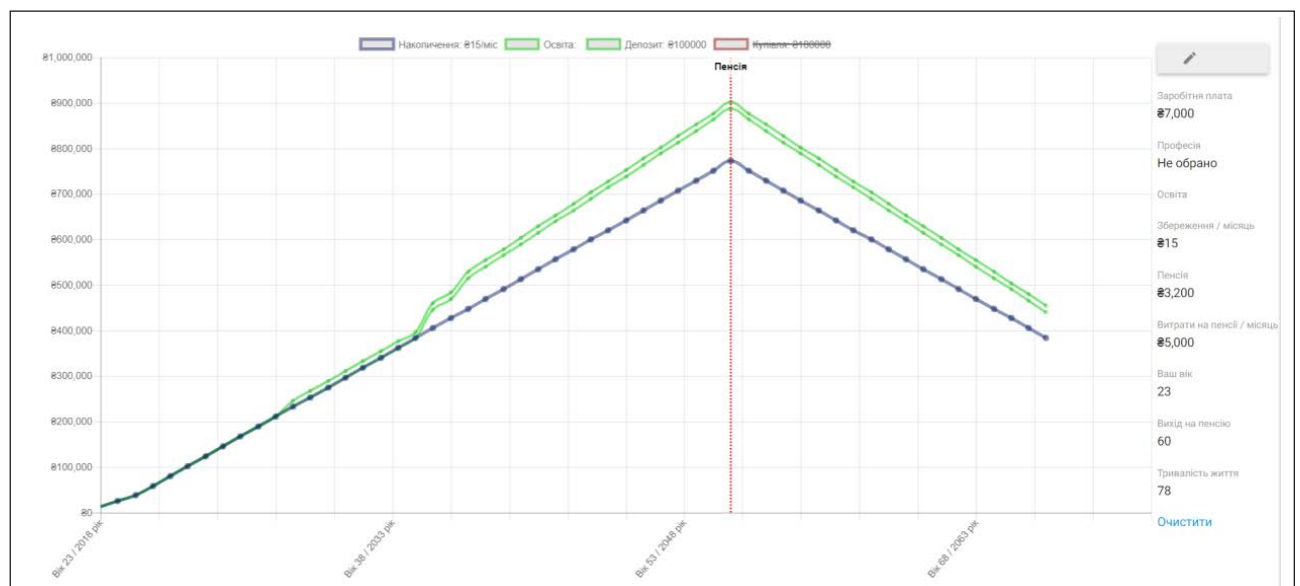


Рис. 13. Вплив освіти на заробітну плату

Список літератури:

1. Adams W.K. Student engagement and learning with PhET interactive simulations. URL: http://www.jdes.tyc.edu.tw/PhET2013/publications/MPTL_2010_PhET_final.pdf.
2. Aldrich C. Learning by Doing: A Comprehensive Guide to Simulations, Computer Games, and Pedagogy in E-Learning and Other Educational Experiences. San Francisco: Pfeiffer, A Wiley Imprint. URL: <http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0787977357.html>
3. Castronova Edward. Exodus to the Virtual World: How Online Fun is Changing Reality. PalgraveMacmillan. URL: <http://us.macmillan.com/exodustothetvirtualworld/edwardcastronova>.
4. Fortmann-Roe Scott. Insight Maker: A general-purpose tool for web-based modeling & simulation. *Simulation Modelling Practice and Theory*. № 47. 2014. P. 28–45. URL: https://www.researchgate.net/publication/263127834_Insight_Maker_A_general-purpose_tool_for_web-based_modeling_simulation.
5. Model-View-Controller. URL: <https://ru.wikipedia.org/wiki/Model-View-Controller>.
6. Model-View-ViewModel. URL: <https://uk.wikipedia.org/wiki/Model-View-ViewModel>.
7. The 9 Best Personal Software apps in 2017. URL: <https://www.thebalance.com/top-10-budget-software-apps-1293609>.
8. Антонюк Д.С. Зарубіжний досвід використання програмно-імітаційних комплексів економічного спрямування в освіті. *Інформаційні технології в освіті*. 2016. № 2. С. 140–153.
9. Баншиков П.Г., Грищенко О.В., Паздрій В.Я. Моделювання ринкового середовища компаній у навчально-тренінгових технологіях (на прикладі бізнес-симуляції “ViAI+”). *Проблеми освіти: МОН України Інститут інноваційних технологій і змісту освіти*, 2011. № 65. С. 7–12.
10. Баншиков П.Г., Кизенко О.О., Скитьова Г.С. Особливості проведення інтерактивних занять під час комплексного тренінгу на основі бізнес-симуляції “Sigam-Market”. *Аудиторна робота викладача і студента: досвід і напрями вдосконалення: Науково-методична конференція ДВНЗ «КНЕУ імені Вадима Гетьмана» (21 лютого 2012 р., м. Київ)*. Київ : КНЕУ, 2012. С. 7–12.
11. Мацюк О.О. Комп’ютерні симуляції як засіб формування професійної компетентності майбутніх перекладачів в умовах інформаційного суспільства. *Вісник Національної академії Державної прикордонної служби України*. 2013. Вип. 5.
12. Мотуз С.А. Комплексное использование биржевых симуляторов как наиболее эффективное направление их применения. *Управление экономическими системами*. 2013. № 12. С. 59.
13. Шендерук О.Б., Пермінова В.А. Технології ситуативного моделювання. Симуляції або імітаційні ігри. URL: http://www.rusnauka.com/26_SSN_2010/Philologia/71388.doc.htm. (дата звернення: 19.08.2019).

Antoniuk D.S., Vakalik T.A., Yakobchuk A.S., Yanchuk V.M.

DESIGN OF THE BUSINESS-SIMULATION SOFTWARE COMPLEX FOR PERSONAL FINANCE VISUALIZATION AND MANAGEMENT

The article describes Design of the business-simulator software for personal finance visualization and management. The present highly-dynamic state of the socio-economics systems, specific for the modern society and international economy, requires professionals of diverse range of specialties to have theoretical knowledge and practical skills in the areas of economic theory, finance and management. Convenient business-simulation complex oriented to the visualization of the personal financial abilities enables igniting interest and easily present person’s abilities in economic relations. Functionality of the complex allows to use it for defining personal economic and financial goals and self-development.

Functional and non-functional requirements to the tool have been defined. Key modules and classes have been defined. Activity diagrams for the variety of use-cases within business-simulation complex were described. Weakly-connected architecture has been defined as appropriate for the complex; the class-interaction order and sequences were developed; user-friendly interface allows quick onboarding process and informative financial possibilities visualization.

The following functionality is available for the users of business-simulation complex: financial abilities of the user visualization; review of the economic planning example; user parameters, such as age, age of the retirement and life expectancy setup; editing of the expected monthly expenses when retired; editing of the expected monthly income when retired; ability to define the savings schemas, such as fixed amount of percentage of the income; defining salary level changes at the different age; gaining specific degree at the certain age; different type of income definition, such as long-time certificates of deposit or one time asset sale; variety of spending registration, among which loans and asset purchasing are implemented in the complex; particular currency for the income or expense might be defined.

Key words: business-simulator, visualization, management, personal finance, software design.