Міністерство освіти і науки України

Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій

Кафедра кібербезпеки та захисту інформації

ДОПУСТИТИ ДО ЗАХИСТУ:
завідувачка кафедри кібербезпеки
та захисту інформації
_____Наталія ЛУКОВА-ЧУЙКО
«14» червня 2022р.

ПОЯСНЮВАЛЬНА ЗАПИСКА

дипломної роботи

бакалавра

(назва освітнього рівня)

галузь знань 12 Інформаційні технології
(шифр і назва галузі знань)

спеціальність 125 Кібербезпека
(код і назва спеціальності)

освітня програма Кібербезпека
(назва освітньої програми)

на тему: «Захист інформації під час обміну у Web-просторі»

Виконавець: студентка IV курсу, групи КБ-42

_____ Наталія ЛЕБЄДЄВА
(підпис) (прізвище ім'я по-батькові)

	Прізвище, ініціали	**Підпис**
Керівник	Микола БРАЇЛОВСЬКИЙ	

Нормоконтроль	Юрій ЩЕБЛАНІН	

Київ 2022

Міністерство освіти і науки України

Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій

Кафедра кібербезпеки та захисту інформації

ЗАТВЕРДЖЕНО:
завідувачка кафедри кібербезпеки
та захисту інформації
_____Наталія ЛУКОВА-ЧУЙКО
«01» листопада 2021 р.

ЗАВДАННЯ
на виконання дипломної роботи

спеціальності 125 Кібербезпека
(код і назва спеціальності)

освітньої програми Кібербезпека
(назва освітньої програми)

Студентці **КБ-42** **Лебєдєва Наталія Володимирівна**
(група) (прізвище ім'я по-батькові)

Тема дипломної роботи Захист інформації під час обміну у Web-просторі

1. ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Тема дипломної роботи затверджена на засіданні кафедри кібербезпеки та захисту інформації протокол №5 від 29.10.2021 р.

2. ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Розгляд актуальних питань захисту інформації у вебпросторі, розробка модулю автентифікації за клавіатурним почерком

3. ЗМІСТ РОЗРАХУНКОВО-ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ

Основні поняття захисту інформації, інформаційна безпека, автентифікація, біометричний метод захисту, клавіатурний почерк, модуль автентифікації за клавіатурним почерком, діаграма використання

4. ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Практична цінність Використання модулю автентифікації за клавіатурним почерком у системах, де відбувається обмін інформацією

5. ДАТА ВИДАЧІ ЗАВДАННЯ

Дата видачі завдання: 29 жовтня 2021 року

Завдання видав _____ (підпис) Микола БРАІЛОВСЬКИЙ (ініціали, прізвище)

Завдання прийняв до виконання _____ (підпис) Наталія ЛЕБЄДЄВА (ініціали, прізвище)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів робіт	Строки виконання робіт (початок-кінець)	Відмітка про виконання
1	Уточнення постановки задачі	29.10.2021 – 22.01.2022	*виконано*
2	Аналіз літератури	29.01.2022 – 11.02.2022	*виконано*
3	Розгляд понять інформаційної безпеки, автентифікації	12.02.2022 – 24.02.2022	*виконано*
4	Розгляд біометричного методу захисту	25.02.2022 – 24.03.2022	*виконано*
5	Вибір інструментальних засобів для програмної реалізації	25.03.2022 – 07.04.2022	*виконано*
6	Впровадження засобів та механізмів захисту інформації автентифікації за клавіатурним почерком	08.04.2022 – 20.04.2022	*виконано*
8	Програмна реалізація, тестування	06.05.2022 – 20.05.2022	*виконано*
9	Формування висновків по виконанню	21.05.2022 – 01.06.2022	*виконано*
10	Оформлення пояснювальної записки	02.06.2022 – 06.06.2022	*виконано*
11	Підготовка до захисту	07.06.2022 – 19.06.2022	*виконано*

Завдання видав _____ (підпис) Микола БРАІЛОВСЬКИЙ (ініціали, прізвище)

Завдання прийняв до виконання _____ (підпис) Наталія ЛЕБЄДЄВА (ініціали, прізвище)

Термін подання дипломної роботи до ЕК 06 червня 2022 року

РЕФЕРАТ

Пояснювальна записка дипломної роботи складається зі вступу, трьох розділів, загальних висновків, списку використаних джерел та додатків. Основний текст займає 68 сторінку, включає в себе зміст, вступ, три розділи дипломної роботи, висновки та список джерел. У пояснювальній записці дипломної роботи міститься 8 рисунків.

Метою дослідження дипломної роботи є створення модулю автентифікації користувачів за клавіатурним почерком.

Для досягнення зазначеної мети поставлено наступні завдання:

- проаналізувати поняття інформації та інформаційної безпеки під час її обміну;
- проаналізувати поняття автентифікації, як метод захисту інформації;
- проаналізувати поняття клавіатурного почерку, як метод автентифікації;
- розглянути актуальний стан питання автентифікації користувачів;
- обрати мову програмування;
- обрати середовище розробки;
- визначити додатковий інструментарій для розробки системи автентифікації;
- провести аналіз варіантів діяльності системи автентифікації за клавіатурним почерком;
- провести проектування внутрішньої будови системи автентифікації за клавіатурним почерком;
- провести розробку графічного інтерфейсу користувача системи автентифікації за клавіатурним почерком;
- провести тестування системи автентифікації за клавіатурним почерком.

Об’єктом дослідження засоби підвищення безпечності використання веб-простору.

Предметом дослідження є підвищення безпечності веб-простору за рахунок автентифікації користувачів за клавіатурним почерком.

Практичною цінністю отриманих результатів є використання модулю автентифікації за клавіатурним почерком у системах, де відбувається обмін інформацією.

Ключові слова: інформаційна безпека, автентифікація, типи автентифікації клавіатурний почерк, база даних, мова програмування C#, діаграма використання.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

ANSI	–	American National Standards Institute
ARPANET	–	Advanced Research Projects Agency Network
CIA	–	Confidentiality, integrity and availability
CIL	–	Common Intermediate Language
CLR	–	Common Language Runtime
CLS	–	Common Language Specification
CTS	–	Common Type System
IDE	–	Integrated Development Environment
ISO	–	International Organization for Standardization
IT	–	Information Technology
JIT	–	Just-In-Time
PIN	–	Personal Identification Number
SQL	–	Structured Query Language
SSMS	–	SQL Server Management Studio
TCP/IP	–	Transmission Control Protocol/Internet Protocol
UML		Unified Modeling Language
ООП	–	Об'єктно-орієнтоване програмування
ПЗ	–	Програмне забезпечення

ЗМІСТ

ВСТУП

В сучасному світі зі швидким розвитком різного роду інформаційних технологій росте попит на засоби захисту приватності користувачів в мережі.

Кожного дня створюються і покращуються найрізноманітніші способи забезпечення безпеки користувачів під час використання веб-простору.

Зазвичай, подібні рішення є суто програмними, будь то двохфакторна автентифікація або додаткові рівні шифрування даних.

Біометрична автентифікація при розробці різного роду веб-систем відходить на другий план як не релевантна, проте, існує біометричний алгоритм автентифікації, який при цьому реалізується суто програмно, без використання стороннього апаратного забезпечення, на зразок сканеру відбитку пальця або сканеру сітківки.

Мова йде про засіб автентифікації, на якому базується тематика даної роботи стосовно захисту інформації під час її обміну, а саме автентифікації за клавіатурним почерком.

Об’єктом дослідження є засоби підвищення безпечності використання веб-простору.

Предметом є підвищення безпечності веб-простору за рахунок автентифікації користувачів за клавіатурним почерком.

Основна мета — створення модулю автентифікації користувачів за клавіатурним почерком.

Для досягнення поставленої мети слід виконати наступні завдання:

- проаналізувати поняття інформації та інформаційної безпеки під час її обміну;
- проаналізувати поняття автентифікації, як метод захисту інформації;
- проаналізувати поняття клавіатурного почерку, як метод автентифікації;
- розглянути актуальний стан питання автентифікації користувачів;
- обрати мову програмування;

- обрати середовище розробки;
- визначити додатковий інструментарій для розробки системи автентифікації;
- провести аналіз варіантів діяльності системи автентифікації за клавіатурним почерком;
- провести проектування внутрішньої будови системи автентифікації за клавіатурним почерком;
- провести розробку графічного інтерфейсу користувача системи автентифікації за клавіатурним почерком;
- провести тестування системи автентифікації за клавіатурним почерком.

РОЗДІЛ 1
АНАЛІЗ СУЧАСНОГО СТАНУ ПИТАННЯ

1.1 Інформаційна безпека

Інформаційна безпека, яку іноді скорочують до InfoSec, — це практика захисту інформації шляхом пом’якшення інформаційних ризиків. Це частина управління інформаційними ризиками [1, 2]. Зазвичай це передбачає запобігання або зменшення ймовірності несанкціонованого/неналежного доступу до даних або незаконного використання, розкриття, порушення, видалення, пошкодження, модифікацію, перевірку, запис або знецінення інформації [3]. Це також передбачає заплановані дії для зменшення негативних наслідків таких інцидентів. Захищена інформація може мати будь-яку форму, напр. електронні або фізичні, матеріальні (наприклад, документи) або нематеріальні (наприклад, знання) [4, 5]. Основним напрямком інформаційної безпеки є збалансований захист конфіденційності, цілісності та доступності даних (також відомий як тріада CIA), при цьому зосереджено увагу на ефективній реалізації політики, і все це без перешкод для продуктивності організації [6]. Цього можна досягти за допомогою процесу управління ризиками, структуровану, яка включає:

- визначення понять та визначень інформації, пов'язаних з нею активів, аразом із потенційними загрозами, вразливостями та впливами;
- оцінка ризиків;
- вирішувати, як подолати ризики або обробити їх, тобто уникнути, пом’якшити, розділити або прийняти їх;
- якщо потрібне зниження ризику, вибір або розробка відповідних засобів контролю безпеки та їх впровадження;
- моніторити діяльність, в яку були внесені необхідні коригування та можуть стати в нагоді вирішення буль-якиї проблем, можливі зміни, можливості [7].

Таким чином, наукові діячі, професіонали в сфері забезпечення інформаційної безпеки для стандартизування дисципліни, співпрацюють, щоб запропонувати рекомендації, стандарти (в тому числі стандарти паролів), політику безпеки стандарти стосовно антивірусного ПЗ, брандмауера, шифрувального програмного забезпечення, обізнанності з безпеки та навчання, відповідності тощо. [8]. Ці стандарти можуть бути додатково обумовлені широким спектром нормативно-правовими законів, які можуть впливати на те, які дані доступні, їх обробка, де мають зберігатися, яким чином передаватися та їх знищення [9]. Однак при впровадження стандартів та/чи будь-яких правил в компаніях/організаціях це може мати обмежений характер [10].

Можна виділити властивості інформації, якими є: конфіденційність, цілісність, доступність. Говорячи про цілісність, це означає, що інформація не може бути модифікована неавторизованим користувачем. Конфіденційність - інформацію не можемо отримати, якщо користувач неавторизований. Доступність - це про те, що авторизований користувач може отримувати і використовувати інформацію відповідно до правил політики безпеки швидше. Ці проблеми включають, але не обмежуються ними, стихійні лиха, несправність комп’ютера/сервера та фізичну крадіжку. У той час як паперові бізнес-операції все ще переважають і вимагають власного набору практик інформаційної безпеки, корпоративні цифрові ініціативи все більше наголошуються [24, [25], а забезпеченням інформації зараз зазвичай займаються фахівці з безпеки інформаційних технологій (IT).

І не обов’язково, що комп’ютер може означати домашній робочий стіл [26]. Інакше кажучи, комп'ютер - це будь-який пристрій з процесором і певною кількістю пам'яті. Це можуть бути пристрої від немережевих автономних пристроїв, наприклад такі як мобільні телефони, калькулятори, смартфони та планшетні комп’ютери [27]. Фахівці з IT-безпеки майже завжди є на будь-якому великому підприємстві/установі через характер і цінність даних у великих компаніях [28]. Вони відповідають за захист усіх технологій компанії від зловмисних дій, якими може виявитися кібератаки, при чому ціль отримання важливої особистої

інформації, можливих персональних даних та отримання контролю доступу у внутрішніх системах.

Інформаційна безпека та її сфера дійсності за останні роки значно зросла та розвинулась, у зв'язку із її важливістю в сучасному світі [31]. Також сфера ІБ пропонує багато областей для спеціалізації, куди входить захист мереж і суміжної інфраструктури, захист додатків і баз даних, тестування безпеки, планування безперервності бізнесу, цифрову криміналістичну експертизу, виявлення електронних записів, працевлаштування, аудит інформаційних систем [32].

Загрози.

Разом із розвитком інформаційної безпеки з'являється ряд загроз, від атак на програмне забезпечення, крадіжки особистих даних, персональних даних, отримання доступу до інформаційної інфраструктури, кражі обладнання, що є важливим на сьогоднішній день, також саботаж та вимагання інформації [36, 37]. Більшість людей стикалися з програмними атаками. Віруси [38], хробаки, фішингові атаки та троянські коні – кілька поширених прикладів програмних атак. Крадіжка інтелектуальної власності також була широкою проблемою для багатьох компаній у сфері інформаційних технологій (IT) [39]. Крадіжка особистих даних — це спроба діяти як хтось інший, як правило, щоб отримати інформацію особистого характеру, отримати доступ до важливої інформації за допомогою соціальної інженерії [40, 41]. Крадіжка обладнання або інформації сьогодні стає все більш поширеною через той факт, що більшість пристроїв сьогодні є мобільними [42], схильними до крадіжок, а також вони стали набагато більш бажаними, оскільки обсяг даних збільшується. Говорячи про саботаж, мається на увагу як приклад, знищення вебсайту компанії, мета чого полягає в тому, що з боку клієнтів виникла недовіра [43]. А при вимаганні інформації відбувається крадіжка будь-якої корисної інформації іншої особи, яка може стати корисною для зловмисника, прикладом є отримання платежу в обмін на повернення інформації чи майна назад їх власнику, як у випадку з програмним забезпеченням-вимагачем [44]. І при цьому, існують певні способи особистого захисту від атак, і однією із функціональних запобіжних заходів є періодичне

інформування користувачів [45]. Загрозою для організації можуть бути внутрішні співробітники, користувачі.

І слід зауважити, що інформаційна безпека може розглядатися по-різному залежачи від культури, цінностей, важливості інформації в певному середовищі.

Історія.

Дипломати та військові командири, з перших днів спілкування, розуміли, що необхідність забезпечення певного механізму для захисту конфіденційності кореспонденції та мати певні засоби виявлення фальсифікацій є важливою частиною. [53]. Здебільшого захист був досягнутий шляхом застосування процедурного контролю обробки [55, 56]. Конфіденційна інформація була позначена, щоб вказати, що вона повинна захищатися та транспортуватися довіреними особами, охоронятися та зберігатися в захищеному середовищі або надійному ящику [57]. у міру розширення поштових послуг уряди створювали офіційні організації для перехоплення, розшифровки, читання та повторного запечатування листів (наприклад, Секретне бюро Великобританії, засноване в 1653 році [58]).

У середині дев'ятнадцятого століття були розроблені більш складні системи класифікації, щоб дозволити урядам керувати своєю інформацією відповідно до ступеня чутливості [59]. Наприклад, британський уряд певною мірою кодифікував це з опублікуванням Закону про офіційну таємницю в 1889 році [60]. Розділ 1 закону стосувався шпигунства та незаконного розголошення інформації, а розділ 2 – порушення службової довіри [61]. Незабаром було додано захист суспільних інтересів для захисту розкриття інформації в інтересах держави [62]. Подібний закон був прийнятий в Індії в 1889 році, Закон про офіційну таємницю Індії, який був пов'язаний з британською колоніальною епохою і використовувався для розправи з газетами, які виступали проти політики Раджа [63]. У 1923 році була прийнята нова версія, яка поширювалася на всі питання конфіденційної чи секретної інформації для управління [64]. Під час Першої світової війни багаторівневі системи класифікації використовувалися для передачі інформації на різні фронти та з них, що спонукало до більш широкого використання розділів для створення кодів і

розриву в дипломатичних і військових штабах [65]. Кодування стало більш складним між війнами, оскільки машини використовувалися для скремблування та розшифровки інформації [66].

Започаткування історії інформаційної безпеки було саме встановлення комп'ютерної безпеки. Потреба в цьому виникла під час Другої світової війни [67]. Обсяг інформації, яку поширювали країни Альянсу під час Другої світової війни, вимагав формального узгодження систем класифікації та процедурного контролю [68]. У зв'язку з розвитком дедалі складніших сейфів і складських приміщень, з'явився таємничий діапазон маркувань, які вказували на те, хто може працювати з документами (зазвичай офіцери, а не рядові війська) і де їх слід зберігати. Машина Enigma, яка використовувалася німцями для шифрування даних війни і була успішно розшифрована Аланом Тьюрингом, може вважатися яскравим прикладом створення та використання захищеної інформації [70]. Розвивалися процедури для забезпечення належного знищення документів, і саме недотримання цих процедур призвело до деяких з найбільших розвідувальних переворотів війни (наприклад, захоплення U-570 [70]).

Різні мейнфрейми були підключені до Інтернету під час холодної війни для виконання складних завдань, у процесі комунікації легше, ніж розсилання магнітних стрічок назад і вперед комп'ютерними центрами. Таким чином, Агентство перспективних дослідницьких проектів (ARPA) Міністерства оборони Сполучених Штатів розпочало дослідження доцільності створення мережевої системи зв'язку для торгівлі інформацією в Збройних Силах Сполучених Штатів. У 1968 році доктором Ларрі Робертсом був сформульований проект ARPANET, який пізніше перетворився на те, що відомо як Інтернет [71].

У 1973 році піонер Інтернету Роберт Меткалф виявив, що важливі елементи безпеки ARPANET мають багато недоліків, таких як: «вразливість структури та форматів паролів; відсутність процедур безпеки для комутованих з'єднань; неіснуючі ідентифікації та авторизації користувачів», крім від відсутності засобів контролю та гарантій для захисту даних від несанкціонованого доступу. Хакери мали легкий доступ до ARPANET, оскільки номери телефонів були відомі

громадськості [72]. Через ці проблеми, у поєднанні з постійним порушенням комп’ютерної безпеки, а також експоненційним збільшенням кількості хостів і користувачів системи, «безпеку мережі» часто називали «безпекою мережі» [72].

Кінець двадцятого та перші роки двадцять першого століття відбулися стрімким прогресом у сфері телекомунікацій, обчислювального обладнання та програмного забезпечення, а також шифрування даних [73]. Наявність меншого, потужнішого та менш дорогого обчислювального обладнання зробило електронну обробку даних доступною для малого бізнесу та домашніх користувачів [74]. Запровадження протоколу керування передачею/протоколу мережі (TCP/IP) на початку 1980-х років дозволило різним типам комп’ютерів спілкуватися [75]. Ці комп’ютери швидко з’єдналися через Інтернет [76].

Швидке зростання та широке використання електронної обробки даних та електронного бізнесу, що ведеться через Інтернет, разом із численними явищами міжнародного тероризму викликали потребу в кращих методах захисту комп’ютерів та інформації, яку вони зберігають, обробляють та передають [77]. Академічні дисципліни комп'ютерної безпеки та інформаційного забезпечення з'явилися разом з численними професійними організаціями, які поділяють спільні цілі забезпечення безпеки та надійності інформаційних систем.

Ключові поняття.

Тріада CIA: конфіденційність, цілісність і доступність є основою інформаційної безпеки [78]. Члени класичної тріади InfoSec — конфіденційність, цілісність та доступність — у літературі взаємозамінно згадуються як атрибути безпеки, властивості, цілі безпеки, фундаментальні аспекти, інформаційні критерії, характеристики критичної інформації та основні будівельні блоки [79]. Проте тривають дебати щодо того, чи достатньо цієї тріади CIA для вирішення швидко мінливих технологічних і бізнес-вимог, з рекомендаціями, які слід розглянути над розширенням перетину між доступністю та конфіденційністю, а також співвідношенням між безпекою та конфіденційністю [23]. Інколи пропонувалися інші принципи, такі як «підзвітність»; було зазначено, що такі питання, як невідмовність, погано вписуються в три основні концепції [80].

Вперше ця тріада була згадана в публікації NIST у 1977 році [81]. Міністерства оборони США випустила три принципи кібербезпеки, які полягають у сприйнятливості системи, доступі до недоліків і можливості використання недоліків [86, 87, 88].

Конфіденційність.

Конфіденційність — це можливість не розголошувати інформацію неавторизованим особам, програмам або процесам. Це стосується інформаційної безпеки, оскільки вимагає контролю за доступом до захищеної інформації. Конфіденційність вимагає заходів для забезпечення доступу до інформації лише уповноваженим особам, а неуповноваженим особам заборонено доступ до інформації. Простіше кажучи, конфіденційність означає, що щось є таємним і не повинно передаватися ненавмисним особам або організаціям. Якщо конфіденційність порушена, це може призвести до втрати конфіденційності та розкриття конфіденційної інформації громадськості чи іншим особам. Існує широкий спектр інформації, яка може вважатися конфіденційною, наприклад, фінансова інформація, медична інформація та інша конфіденційна інформація. Одна інформація є більш конфіденційною, ніж інша, і вимагає більш високого рівня конфіденційності [91].

Цілісність.

Розглядаючи цілісність даних можна зазначити, що вона є підтримкою та забезпеченням точності та повноти даних протягом усього їхнього життєвого циклу [92]. Тобто, це означає, що зміна даних не може бути здійснена неавторизованим або невиявленим способом [93]. Системи інформаційної безпеки зазвичай включають засоби контролю для забезпечення їх власної цілісності, зокрема захист ядра або основних функцій як від навмисних, так і від випадкових загроз [95]. Багатоцільові та багатокористувацькі комп’ютерні системи мають на меті розділити дані та обробку таким чином, щоб жоден користувач або процес не могли негативно вплинути на інших: проте засоби контролю можуть не досягти успіху, як ми бачимо в таких інцидентах, як зараження шкідливим програмним забезпеченням, злом, крадіжка даних, шахрайство, а також порушення конфіденційності [96].

У більш широкому сенсі, цілісність — це принцип інформаційної безпеки, який включає людську/соціальну, процесну та комерційну цілісність, а також цілісність даних. Як такий він зачіпає такі аспекти, як достовірність, послідовність, правдивість, повнота, точність, своєчасність та впевненість [97].

Цілісність означає захист від неналежної модифікації та знищення інформації, гарантування тієї, що інформація не може бути змінена непомітно, а також забезпечення цілісності інформації. Це означає, що кіберзагрозу або вразливість до кібератак можна виміряти, порушивши один або кілька її принципів. Цінність заснована на аналізі та хешуванні, щоб шифрувати найкращий захист від так і кіберзагроз, таких як кібершпігунство.

Доступність.

Доступність забезпечує доступність інформації для тих, хто її потребує, що включає своєчасний та надійний доступ, незалежно від часу доби, місця проживання, місцезнаходження чи інших факторів [101].

У сфері інформаційної безпеки доступність часто можна розглядати як одну з найважливіших частин успішної програми інформаційної безпеки. Зрештою, кінцеві користувачі повинні мати можливість виконувати службові функції; забезпечуючи доступність, організація здатна виконувати стандарти, які очікують зацікавлені сторони організації [102]. Це може включати такі теми, як конфігурація проксі-сервера, зовнішній веб-доступ, можливість доступу до спільних дисків і можливість надсилати електронні листи [103]. Керівники часто не розуміють технічну сторону інформаційної безпеки і розглядають доступність як легке рішення, але для цього часто потрібна співпраця багатьох організаційних команд, таких як мережеві операції, операції з розробки, реагування на інциденти та управління політикою/змінами [104]. Успішна команда з інформаційної безпеки включає в себе багато різних ключових ролей, які об’єднують і вирівнюють для ефективного забезпечення тріади CIA [105].

1.2 Автентифікація

Автентифікація — це процес визначення того, чи є хтось чи щось насправді тим, ким воно є. Технологія аутентифікації це про контроль доступу, як метод захисту інформації, де перевіряється чи існує збіг облікових даних певної особи (користувача наприклад) з обліковими даними на серверній частині (базі даних авторизованих користувачів, або сервер аутентифікації даних) При цьому аутентифікація забезпечує безпечні системи, безпечні процеси та безпеку інформації підприємства. Аутентифікація дозволяє організаціям захищати свої мережі, дозволяючи лише перевіреним користувачам або процесам отримати доступність до ресурсів, які є захищені. Захищеними ресурсами можуть бути комп'ютерні системи, мережеві системи, база даних (серверна частина), вебсайти, інші мережеві програми чи служби [3].

Коротше кажучи, автентифікація – це процес перевірки законних прав особи перед випуском захищених ресурсів. Зазвичай це досягається шляхом перевірки унікальної інформації, наданої особою. Цю інформацію можна в цілому розділити на три категорії, а саме на аутентифікацію на основі знань, токенів та біометричних даних, як узагальнено в таблиці 1.1.

Таблиця 1.1

Типи автентифікації

Тип	**Переваги**	**Недоліки**	**Приклади**
Знання	Без зусиль Високе сприйняття	Можна забути Можна підробити	Паролі, PIN
Токен	Дешевий, просте розгортання	Втрата, крадіжка	Смарт-карта, мініпристрої
Біометрія	Незабутній, унікальність	Ціна	Відбитки пільців, голос, клавіатурний почерк

Знання зазвичай розглядають як те, що людина знає, що зазвичай міститься у формі текстурного або графічного пароля, персонального ідентифікаційного номера

(PIN) і коду шаблону. Аутентифікація на основі пароля була встановленим методом контролю доступу в різних системах з останніх трьох десятиліть. Ефективність витрат і проста реалізація були головними причинами постійного домінування пароля. Тим не менш, його здатність забезпечувати впевнену та безпечну аутентифікацію зникла через такі причини, як неправомірне використання пароля та збільшення атак вторгнення. Простий пароль є основним вибором, коли справа доходить до вибору пароля, такого як дата народження, псевдонім, ініціали та звичайні слова словника, які можна легко вгадати або зламати. Щоб погіршити ситуацію, користувачі завжди схильні використовувати один і той самий або схожий пароль для кількох систем. Ці шкідливі звички використання сприяють погіршенню якості аутентифікації на основі знань.

Токен відноситься до об’єкта, який вимагає від користувача фізичного володіння в якості форми аутентифікації. Поширені токени включають, але не обмежуючись ними, картки, кредитні картки та міні-пристрої. Хоча широкомасштабне розгортання є відносно простим, воно має свою слабкість. Токен вразливий до втрати або крадіжки, оскільки користувачеві може бути незручно або важко постійно зберігати його в безпеці. Це означає, що немає гарантії однозначної ідентифікації законного користувача, навіть якщо він володіє маркером. Зазвичай цей недолік можна усунути, використовуючи токен разом із методом, заснованим на знаннях. Таким чином, ці дві сутності разом здійснюють простий двофакторний процес аутентифікації, який виробляє більш надійну аутентифікацію, засновану на припущенні, що секретність знань не порушена.

Біометрія відноситься до певних фізіологічних або поведінкових характеристик, які однозначно пов'язані з людиною. Ця риса дуже відмінна і може бути використана для розрізнення різних людей. Динаміка натискання клавіш відноситься до процесу вимірювання та оцінки ритму набору тексту людини на цифрових пристроях. Таким пристроєм, як правило, є клавіатура комп’ютера, мобільний телефон або сенсорна панель. При взаємодії людини з цими пристроями створюється форма цифрового сліду. Вважається, що ці підписи багаті когнітивними

якостями [9], які є досить унікальними для кожної людини та мають величезний потенціал як особистісний ідентифікатор.

Історія появи біометрії динаміки натискання клавіш датується кінцем 19 століття, коли телеграфна революція була на піку. Це був основний інструмент міжміського зв'язку в ту епоху. Телеграфісти могли легко розрізняти один одного, просто прислухаючись до стукання крапок і тире. У той час як телеграфний ключ слугував пристроєм введення, клавіатура комп'ютера, мобільна клавіатура та сенсорний екран є поширеними пристроями введення в 21 столітті. Крім того, було відзначено, що шаблон натискання клавіш має ті самі нейрофізіологічні фактори, які роблять рукописний підпис унікальним [11], коли люди покладалися на підтвердження особистості протягом багатьох століть. Насправді, шаблон натискання клавіш може надати ще більш унікальну функцію для аутентифікації, яка включає тривалість натискання клавіші та затримки, швидкість набору тексту та тиск при введенні. Серед найбільш ранніх значущих досліджень динаміки натискань клавіш з аутентифікації була проведена робота, з тих пір ця область поступово набирала обертів.

1.3 Клавіатурний почерк

Ще в 1860 році досвідчені телеграфісти зрозуміли, що можуть розпізнати кожну людину за унікальним ритмом постукування. Для навченого вуха м'яке дотик кожного оператора може бути так само впізнаваним, як голос члена сім'ї.

Під час Другої світової війни військова розвідка використовувала методологію, відому як «Кулак відправника», щоб визначити унікальний спосіб введення «крапок» і «тире» у повідомленні азбукою Морзе. Його використовували, щоб відрізнити друга від ворога. Темп і стиль спілкування дозволили експертам-операторам визначити, хто перебуває на іншому кінці.

Дослідження біометричних даних динаміки натискань клавіш збільшуються, особливо в останнє десятиліття. Основна мотивація цих зусиль пов'язана з тим, що біометричні дані динаміки натискання клавіш є економічними і можуть бути легко

інтегровані в існуючі системи безпеки комп’ютера з мінімальними змінами та втручанням користувача. Численні дослідження були проведені з точки зору пристроїв збору даних, представлення ознак, методів класифікації, експериментальних протоколів та оцінок. Однак сучасне обширне опитування та оцінка поки що недоступні. В кібербезпеці даний метод розпізнавання користувачів за його унікальністю також набула сенсу. Це метод відноситься до біометричного методу захисту, як автентифікація користувачів, що має назву клавіатурний почерк, який складається із набору динамічних параметрів при роботі на клавіатурі. Принцип дуже простий: людина вводить інформацію, використовуючи клавіші на клавіатурі, таким чином вона виробляє свій особистий стиль набору тих чи інших слів. Даний стиль є унікальним і практично не повторний. Визначити можна за такими властивостями:

- кількість пальців, які задіяні під час набору тексту;
- тривалість натискання самих клавіш;
- проміжок часу між натисканнями клавіш;
- використання основної чи додаткової частини клавіатури;
- характер здвоєних чи строєних натискань;
- улюблене поєднання гарячих клавіш тощо.

Необхідним завданням при аутентифікації користувача за допомогою динамічного натискання, тобто клавіатурного почерку є "навчання" програми, яка проводитиме автентифікацію. "Навчання" – накопичення інформації, яка описує особливості роботи кожного користувача з клавіатурою. Потім відбувається обробка інформації.

Першим етапом обробки даних є фільтрація. Вхідний потік даних перетворюється таким чином, щоб він не містив інформацію про "службові" клавіші - клавіші управління курсором, функціональні клавіші і т.п.

Другим етапом є виділення інформації, що відноситься до характеристик користувача:

- помилки та їх кількість;
- певний період часу утримання клавіш;

- інтервали між натисканнями клавіш;
- число перекриттів між кнопками;
- швидкість з якою користувач набирає текст;
- ступінь аритмічності під час набору.

Далі введені дані обробляються та розраховані еталонні параметри зберігаються в серверній частині бази даних.

Розглянемо переваги та недоліки використання клавіатурного почерку.

Переваги.

Унікальність Подію натискання клавіші можна виміряти з точністю до мілісекунд за допомогою програмного забезпечення. Таким чином, недоцільно відтворювати власний шаблон натискання клавіш у такій високій роздільній здатності без величезних зусиль.

Низька вартість впровадження та розгортання На відміну від традиційних фізіологічних біометричних систем, таких як розпізнавання відбитків долоні, райдужної оболонки ока та відбитків пальців, які покладаються на спеціальні пристрої та апаратну інфраструктуру, розпізнавання динаміки натискання клавіш повністю реалізується програмно. Перевага низької залежності від спеціалізованого обладнання не тільки може значно знизити витрати на розгортання, але й створює ідеальний сценарій для впровадження в середовищі віддаленої автентифікації.

Прозорість та неінвазивність. Одним із важливих параметрів динаміки натискання клавіш, які біометричні дані мають порівняно з іншими параметрами, є ступінь прозорості, яку вона забезпечує. Він не вимагає жодних змін або мінімальних змін у поведінці користувача, оскільки фіксація шаблону натискання клавіш здійснюється за допомогою внутрішньої реалізації програмного забезпечення. У більшості випадків користувач може навіть не підозрювати, що він захищений додатковим рівнем аутентифікації. Ця простота сприяє не тільки розробнику системи, але й тим кінцевим користувачам, які мають невелику технічну підготовку або взагалі не мають її.

Збільшення міцності пароля та терміну служби. Пароль був найбільш поширеним методом аутентифікації особи, незважаючи на те, що системи, які

покладаються виключно на єдиний набір облікових даних, є слабкими та вразливими. Дослідники визначили біометрію динаміки натискання клавіш як ймовірне рішення, яке здатне принаймні додати додатковий рівень захисту та збільшити термін служби пароля. Біометрія динаміки натискання клавіш надає можливість поєднати простоту схеми паролів із підвищеною надійністю, пов'язаною з біометричними даними. Використовуючи біометричні дані динаміки натискання клавіш, користувач може зосередитися на створенні надійного пароля, уникаючи при цьому перевантаження різними наборами паролів.

Запобігання реплікації та додаткова безпека. Шаблони натискання клавіш важче відтворити, ніж письмові підписи. Це тому, що більшість систем безпеки допускають лише обмежену кількість помилкових спроб введення перед блокуванням облікового запису. Крім того, інтеграція біометричних даних динаміки натискання клавіші залишає атаку випадкового вгадування пароля застарілою [14], а вкрадені облікові дані стають зовсім незначними, оскільки успішне володіння секретним ключем є лише умовою всього ланцюга аутентифікації. Навіть якщо він буде скомпрометований, новий біометричний шаблон для введення можна легко відновити, вибравши новий пароль.

Постійний моніторинг та аутентифікація. Постійний моніторинг та аутентифікація часто залишаються поза увагою, але вони є відносно важливими. Біометрія динаміки натискання клавіш пропонує спосіб постійної перевірки [15] законної особистості користувача. Поки триває взаємодія користувача із системою через пристрої введення, шаблон натискання клавіш можна постійно контролювати та переоцінювати.

Недоліки.

Нижча точність. Біометрія динаміки натискання клавіш є нижчою з точки зору точності автентифікації через зміни в ритмі введення, викликані зовнішніми факторами, такими як травми, втома або відволікання. Проте й інші біометричні системи такі чинники не обійшли стороною.

Нижча постійність. Більшість поведінкових біометричних даних, як правило, мають меншу постійність порівняно з фізіологічними біометричними. Шаблон

набору тексту людини може поступово змінюватися після звикання до пароля, дозрівання навичок друку, адаптації до пристроїв введення та інших факторів навколишнього середовища. Однак дослідники рекомендували методи постійного оновлення збереженого профілю натискання клавіш, які можуть вирішити цю проблему.

1.4 Аналіз існуючих рішень

Питання автентифікації по клавіатурному почерку користувача на сьогоднішній день не є популярним і не розробляється на достатньому рівні. На відміну від методів фізіологічної біометрії використання методів поведінкової біометрії заснованої на клавіатурному почерку не потребує додаткових пристроїв. З урахуванням цього можна зробити висновок про відсутність як таких рішень, які б надавали функціонал автентифікації по клавіатурному почерку, особливо у відкритому доступі. Ми можемо лише побачити які компанії, як наприклад TypingDNA, ID Control, BehavioSec розробляють подібні програми для автентифікації за клавіатурним почерком, але що досі не розкрито та не популярно в нашій державі – Україні.

На даний момент проблематика реалізації механізму автентифікації по клавіатурному почерку розроблена виключно в якості математичних моделей, які достатньо далекі від практичної реалізації і представляють собою лише теоретичний базис для подальших практичних досліджень.

1.5 Постановка задачі

Основна задача полягає в створенні модуля автентифікації користувача за клавіатурним почерком, без зайвих механік на зразок системи, яка б захищалася даним механізмом або надлишково детального інтерфейсу.

Основні задачі, які необхідно виконати для досягнення повноцінності функціоналу можна звести до таких пунктів:

- наявність варіацій текстів для виявлення клавіатурного почерку;
- реалізація механізму побудови моделі клавіатурного почерку;
- можливість зареєструвати нового користувача з побудовою моделі клавіатурного почерку;
- можливість авторизуватися за клавіатурним почерком.

Висновки за розділом 1

В першому розділі дипломної роботи, згідно поставлених завдань, був проведений аналіз сучасного питання стосовно таких понять як інформація, інформаційна безпека, разом з цим розглянуті питання, чому інформація така важлива, які можливі загрози існують під час обміну інформацією і в цілому яка її цінність у веб-просторі. Згідно визначення її цінності, питанням постало яким чином можна захистити інформацію, спираючись на властивості захисту - конфіденційність, цілісність та доступність. Таким чином, були розглянуті та дослідженні більш детально властивості та поняття автентифікації, які типи існують, та визначивши такий метод захисту інформації, було обрано розробити біометричний метод захисту, а саме за клавіатурним почерком, тобто динамікою натискання клавіш. Таким чином забезпечуючи інформаційну безпеку користувача за його біометричними даними, де великою перевагою даного методу є його точна ідентичність, так як біометрію підробити потребує багато зусиль, а отже забезпечити надійну безпеки під час обміну інформацією у веб-просторі.

РОЗДІЛ 2
ВИБІР ІНСТРУМЕНТАЛЬНИХ ЗАСОБІВ ВИКОНАННЯ ЗАДАЧІ

2.1 Вибір мови програмування

Мова програмування, яка була обрана під час написання дипломної роботи для здійснення мети стосовно створення методу захисту інформації під час автентифікації за клавіатурним почерком є C#, мова яка була створена відомою всім корпорацією Microsoft, де підтримується середовище .NET Framework. Розробником якої вважається Андерс Хейльсберг (Anders Hejlsberg), який був в свої часи відомим спеціалістом з програмування C#, мова яка походить від таких відомих мов початківців як C та C++. Дана мова успадкувала, як приклад синтаксис ключові слова і оператори, а від такої мови програмування як C++ - вдосконалену об’єктну модель. І більш того, C# є тісно пов'язаним із мовою програмування Java.

C# та Java багато в чому відрізняються один від одного, але мають загальне походження, вони схожі один на одного як «близькі», але не як «кровні родичі». Подібність полягає в тому, що в обох мовах застосовується проміжний код для забезпечення безпеки та переносимості, відмінностями є лише деталі реалізації. підтримується розподілене. Схожість полягає в тому, що як в C# так і в Java існують можливості для перевірки помилок при виконанні, впроваджене забезпечення безпеки та кероване виконання, і знову, відмінності криються у деталях реалізації. Як приклад, C# надає доступ до покажчиків – це засоби програмування, які підтримуються в C++, на відміну від Java. Отже, C# поєднує у собі ефективність, властиву C++, та властиву типову безпеку характерну Java. Між ефективністю та безпекою, компроміси в C# врівноважені та абсолютно прозорі.

Останньою версією мови є C# 10.0, яка була випущена в 2021 році в .NET 6.0 [17, 18].

Як вже зазначалося, C# є мовою розробки програмного забезпечення на досить популярній платформі, характерній ООП .NET від корпорації Microsoft.

За допомогою C#, можна розвивати різні типи надійних і широких пристосунків, такиї як:

- застосунки для Windows (Window applications);
- додатки веб-сервісів (Web service applications);
- програми для бази даних (Database applications);
- веб-додатки (Web applications);
- розповсюджені програми (Distributed applications).

Історія.

До кінця 1970-х років масштаби багатьох проектів наблизилися до меж, з якими вже не могли впоратися методики структурного програмування взагалі та мова С зокрема. Для вирішення цієї проблеми було відкрито новий напрямок у програмуванні - так зване об'єктно-орієнтоване програмування (ООП). Застосовуючи метод ООП, програміст міг працювати з більшими програмами. Але головна складність у тому, що З, найпоширеніший тоді мова, не підтримував ООП. Прагнення створення об'єктно-орієнтованого варіанта. З зрештою призвело до появи С++. Наступним важливим кроком у розвитку розробки на мовах програмуваннях вважається розробка на Java, яка взагалі спочатку мала назву Oak, тобто дуб, в компанії Sun Microsystems, у 1991 році. Джеймс Гослінг (James Gosling) вважається "рушійною силою" у розробці Java був, але немала роль у роботі над цією мовою належить також Патріку Ноутону (Patrick Naughton), Крісу Уорту (Chris Warth), Еду Френку (Ed Frank) та Майку Шерідану (Mike Sheridan). А вже потім був випущений у вигляді альфа-версії у середині 2000 року C#. Головним розробником C# був Андерс Хейльсберг — один із провідних у світі фахівців з мов програмування, який може похвалитися низкою помітних досягнень у цій галузі. Досить сказати, що у 1980-ті роки він був автором дуже вдалої і що мала велике значення розробки — мови Turbo Pascal, витончена реалізація якої стала зразком для створення всіх наступних компіляторів.

Мова C# безпосередньо пов'язана з С, С++ і Java. І це невипадково. Адже це три найпоширеніші і визнані у всьому світі мови програмування. Крім того, на момент створення C# практично всі професійні програмісти вже володіли С, С++

або Java. Завдяки тому, що C# побудований на такій міцній і зрозумілій підставі, перейти цією мовою з C, C++ або Java не мало особливих труднощів. А оскільки і Хейльсбергу не потрібно (та й небажано) було винаходити велосипед, він міг зосередитися безпосередньо на вдосконаленнях і нововведеннях в C#.

Незважаючи на те що в Java успішно вирішуються багато питань переносимості програм в середовищі Інтернету, його можливості все ж таки обмежені. Йому, зокрема, бракує міжмовної можливості взаємодії, яка називається також багатомовним програмуванням. Це можливість коду, написаного однією мовою, легко взаємодіяти з кодом, написаним іншою мовою. Міжмовна можливість взаємодії потрібна на побудову великих, розподілених програмних систем. Вона бажана також для створення окремих компонентів програм, оскільки найбільш цінним компонентом вважається той, який може бути використаний у різних мовах програмування і в найбільшому числі операційних середовищ.

Існують певні версії в історії C#, з дня започаткування мови програмування C#, та значною віхою історія розвитку C# став випуск версії 3.0. У зв'язку з впровадженням багатьох нових властивостей у версії C# 2.0 можна було очікувати деякого уповільнення розвитку C#, оскільки програмістам потрібен час для їх освоєння, але цього не сталося. З появою версії 3.0 корпорація Microsoft впровадила низку нововведень, які цілком змінили загальне уявлення про програмування. До цих нововведень відносяться, серед іншого, лямбда-вираження, мова інтегрованих запитів (LINQ), методи розширення та неявно типізовані змінні. Звичайно, всі ці нові можливості дуже важливі, оскільки вони помітно вплинули на розвиток цієї мови, але серед них особливо виділяються дві: мова інтегрованих запитів (LINQ) і лямбда-вираження. Мова LINQ і лямбдавираждения вносять абсолютно новий акцент у програмування на C# і ще глибше підкреслюють його провідну роль у безперервній еволюції мов програмування.

Було вже визначено важливість платформи .NET Framework, та стосовно призначення платформи, вважається, що це служить середовищем для підтримки розробки програм, та можливе виконання розподілення компонентів додатків при цьому. Також, забезпечується спільне використання мов програмування, які

підтримуються, забезпечення безпеки, впроваджена модель програмування для платформи Windows, та не менш важливе – переносимість програм.

Що ж до взаємозв'язку з C#, то визначено, що існують два важливих елементи в середовищі .NET Framework, перший з них – CLR (Common Language Runtime), або загальномовге середовище виконання. Це система, яка управляє виконанням програм. Серед інших переваг CLR як складова частина середовища це те, що впроваджена підтримка багатомовного програмування, та забезпечена переносимість та безпека [31].

Наступний, другий важливий елемент – бібліотека класів, яка надає створеним програмам доступ до середовища виконання. Так, якщо потрібно виконати операцію введення-виводу, наприклад вивести що-небудь на екран, то для цієї мети використовується бібліотека класів .NET. Досить сказати, що клас — це об'єктно-орієнтована конструкція, яка допомагає організувати програми. Якщо програма обмежується засобами, які визначаються в бібліотеці класів .NET, то така програма може виконуватися скрізь, де підтримується середовище виконання .NET. А оскільки в C# бібліотека класів .NET використовується автоматично, то програми на C# свідомо виявляються переносимими у всі .NET Framework.

Типізація.

Типізована мова, яка вважається стогою має типи даних, такі як: byte, sbyte, short, ushort, int, uint, long, ulong, float, double, char, bool, decimal, string, object, та відіграють особливе значення. Як і в будь-якій мові програмування, в C#, також типи даних створюють операції, та контролюються з боку компілятора, недопустимі операції не компілюються, а це означає дозвіл виключити помилки та забезпечити надійність програм. Іншою характеристикою типізації є те, що для забезпечення контролю типів, всі значення та вирази мають належати до певного типу, в іншому випадку «безтипові» змінні – в цій мові не існують. І тип значення може визначати операції, які виконуються на типом, а отже операція одного типу, яка дозволена, може бути недопустимою іншого.

В мові програмування C# існує строгий логічний тип даних, який інакше називається bool (має структуру Boolean). В полях public static readonly string

FalseString, public static readonly string TrueString логічні значення true і false містяться в зручній формі. Так, якщо вивести вміст поля FalseString за допомогою методу WriteLine(), на екрані з'явиться рядок "False". А також, керуючі оператори поділяються на три категорії: оператори вибору, до яких належать оператори if і switch, ітераційні оператори, у тому числі оператори циклу for, while, do-while та foreach, а також оператори переходу: break, continue, goto, return та throw. За винятком оператора throw, який є невід'ємною частиною вбудованого C# механізму обробки виняткових ситуацій.

Перераховані члени розміщуються у власній області дії. При цьому глобальні змінні та функції не дозволяється використовувати. Говорячи про члени та методи, маємо зазначити, що вони мають бути оголошені всередині класів. А статичні члени відкритих класів можуть замінювати глобальні змінні та функції. На відміну C та C++, локальні змінні не можуть затінювати блочні змінні, які охоплюються.

Методи є підпрограми, які маніпулюють даними, визначеними у класі, а у багатьох випадках вони надають доступ до цих даних. Як правило, інші частини програми взаємодіють із класом за допомогою його методів. Метод складається з одного або кількох операторів. У грамотно написаному коді C# кожен метод виконує лише одну функцію. Кожен метод має своє ім'я, яким він викликається. Загалом, методу, як ім'я можна привласнити будь-який дійсний ідентифікатор. Слід, однак, пам'ятати, що ідентифікатор Main() зарезервований для методу, з якого починається виконання програми. Крім того, як імена методів не можна використовувати ключові слова C#.

Властивості.

В мові програмування C# існує підтримка класів з певними властивостями. Вони можуть бути простими функціями доступу з резервним полем або реалізовувати функції геттера та встановлення.

Починаючи з C# 3.0 доступний синтаксичний цукор автоматично реалізованих властивостей [66], де засіб доступу (гетер) і мутатор (сетер) інкапсулюють операції з одним атрибутом класу.

Простір імен.

Простір імен визначає область оголошень, в якій можна зберігати одну множину імен окремо від іншої. По суті, імена, оголошені в одному просторі імен, не вступатимуть у конфлікт із аналогічними іменами, оголошеними в іншій області. Так, у бібліотеці класів для середовища .NET Framework, яка є бібліотекою класів C#, використовується простір імен System. Саме тому рядок коду using System; зазвичай вводиться на початку будь-якої програми на C# [67].

Код мови C# має таку властивість як компілюватися в додатки або є збіркою, яка має розширеннями exe або dll мовою CIL. Потім, запускаючи програму на виконання відбувається JIT-компіляція (Just-In-Time) в машинний код, який потім виконується. Програма може бути великою, тобто важко, а через це мати велику кількість інструкцій і через це, як результат компілюватиметься буде лише та частина програми, до якої безпосередньо йде звернення. Тобто при зверненні до іншої частини коду, вона буде скомпільована з CIL в машинний код. А вже до завершення роботи програми, частина програми буде скомпільована. Це, в свою чергу про продуктивність та її підвищення.

Програма, створена на C#, також може мати назву керований код (анг. managed code), це говорить за те, що управляється програма загальномовним середовищем, вже згадуваним в даній роботі, CLR і що ця програма створена на основні платформи . NET, і при цьому загальномовне середовище завантажує програму і, якщо це потрібно, очищає пам’ять. Але наприклад, якщо програма створена на C++, існує такі програми, які компілюються над загальну мову CIL, як приклад C#, VB.NET чи F#, а звичайний машинний код. В даному випадку, платформа .NET вже не керує програмою. У

З цього можна сказати, що у той час платформа .NET може надавати такі можливості як взаємодія з некерованим кодом.

Доступ до пам'яті.

Покажчики адреси пам’яті в C# використовуємо лише в блоках, зазвичай вони позначені як небезпечні, а отже, це означає, що програми з небезпечним кодом мають відповідні дозволи для виконання. Більшість доступу до об’єктів

здійснюється через безпечні посилання на об’єкт, це говорить за те, що вказується на «живий» об’єкт, або мають мати чітко визначенне значення null. Екземпляр типу значення «некерований» - на нього вказує небезпечний покажчик, який, в свою чергу не містить жодних посилань на зібрані сміття об’єкти, рядок, масив, або блок пам’яті, який виділив стек.

Не позначений код як небезпечний, все ще може зберігати та маніпулювати покажчиками за допомогою типу System.IntPtr, але не розіменовувати їх. Керована пам'ять не може бути звільнена явно; замість цього він автоматично збирає сміття.

Це процес, збору сміття, може вирішити проблему витоку пам’яті, при цьому звільнюючи того, хто пише код від відповідальності за звільнення пам’яті, яка, в більшості випадків, більше не знадобиться і не потрібна. Але, код, який зберігає посилання на об’єкти протягом тривалого часу, не можемо для нього зазнавати більшого використання пам’яті ніж потрібно, і вже після звільнення останнього посилання на об’єкт пам’яті, пам’ять стає доступною для збирання сміття.

Але при цьому є винятки. Методи в стандартних бібліотеках за деяких обставин регулярно генерують системні винятки, і діапазон виключених винятків зазвичай має документуватися. Наприклад, розглядаючи спеціальні винятки для класів, вони можуть бути визначені і при цьому дозволяють запровадити обробку для конкретних обставин при необхідних потреб [69].

Система загального типу.

Незважаючи на всі переваги, які CLR середа дає керований код, для максимального використання його разом із програмами, написаними на інших мовах, він повинен підчинятися загальноязиковій специфікації (Common Language Specification — CLS), яка визначає ряд загальних властивостей для різних .NET-сумісних мов. Соответствие CLS особливо важливо при створенні програмних компонентів, призначених для застосування в інших мовах. У CLS в якості підмножества входить загальна система типів (Common Type System — CTS), в якій визначаються правила, до яких відносяться типи даних. Тобто, що в C# підтримується як CLS, так і CTS, що є уніфікованою системою типів [78].

Дана уніфікована система типів дає таке розуміння, що типи (в тому числі і примітивні цілі числа) відносяться до підкласів так званого класу System.Object.

Розпакування і бокс.

Операцією перетворення об’єкта типу значення у значення відповідального еталонного типу [78], має назву боксування та є неявним.

Операцією перетворення значення опорного типу у значення типу називається розпакуванням. Воно вимагає в C# явного приведення типу. Тобто об’єкт із коробкою типу T можна розпакувати лише до T (або T, що допускає значення NULL) [79].

```
int foo = 42; // Тип значення.

object bar = foo; // foo упаковується в панель.

int foo2 = (int)bar; // Розпакований назад до типу значення.
```

Реалізація.

Корпорація Майкрософт очолює розробку компіляторів C# з відкритим вихідним кодом і набору інструментів. Перший компілятор, Roslyn, компілює на проміжну мову (IL), а другий, RyuJIT [88], є компілятором JIT (точно вчасно), який є динамічним і виконує оптимізацію «на льоту» та компілює IL у рідний код для інтерфейсу ЦП [89]. RyuJIT є відкритим вихідним кодом і написаний на C++ [90]. Roslyn повністю написаний на керованому коді (C#), був відкритий, а функціональні можливості з’явилися як API. Таким чином, розробники можуть створювати інструменти рефакторингу та діагностики [5, 91]. Дві гілки офіційної реалізації: .NET Framework (з закритим кодом, лише для Windows) і .NET Core (з відкритим вихідним кодом, кросплатформний); зрештою вони об’єдналися в одну реалізацію з відкритим кодом: .NET 5.0 [92]. У .NET Framework 4.6 новий компілятор JIT замінив колишній.[88, 93].

Розглянемо деякі інші компілятори в C#, деякі які з ним можуть включати бібліотеку класів .NET, реалізацію, яка називається Common Language Infrastructure:

- Mono, проект, який спонсорувала корпорація Microsoft, та має властивість забезпечувати компілятор C# з відкритим вихідним кодом, повну

реалізацію CLI з відкритим кодом (як вони відображаються в специфікації ECMA, включаючи необхідні бібліотеки фреймворків) і майже повну реалізацію бібліотек класів NET. до .NET Framework 3.5.

- Ланцюжок інструментів Elements від RemObjects включає RemObjects C#, який компілює код C# у загальну проміжну мову .NET, байт-код Java, Cocoa, байт-код Android, WebAssembly та рідний машинний код для Windows, macOS та Linux.
- Проект DotGNU (зараз припинений) також забезпечив компілятор C# з відкритим вихідним кодомІгровий движок Unity використовує C# як свою основну мову сценаріїв [94].

На усе вищезазначене, стосовно функціоналу, який підтримує мова програмування, операційні системи, які підтримуються, простота вивчення, тощо, згідно цього була обрана мова програмування C#, для розробки даного проекту.

Таким чином, спираючись на детальне пояснення та дослідження усіх властивостей даної мови програмування як C#, то як результатом дослідження можна зазначити, що під розробку для автентифікації за клавіатурним почерком, з можливою інтеграцією дана мова програмування задовольняє умовам написання проектної частини даної роботи.

2.2 Вибір середовища розробки

Visual Studio є інтегрованим середовищем розробки програм, створеним корпорацією Microsoft. Таке середовище дає можливість правити, компілювати, виконувати та налагоджувати програми на C#, не залишаючи це грамотно організоване середовище. Visual Studio надає не тільки всі необхідні засоби для роботи з програмами, але допомагає правильно організувати їх. Вона виявляється найбільш ефективною для роботи над великими проектами, хоча може бути з тим самим успіхом використана і для розробки невеликих програм, як приклад, створення модулю для автентифікації користувачів за клавіатурним почерком.

Багатофункціональною програмою, яка підтримує багато аспектів розробки програмного забезпечення та є визначеною Visual Studio IDE є інтегроване середовище розробки IDE, іншими словами, це стартова панель для редагування, створення кода, його налагодження та потім для публікації програми. Окрім налагоджувача та стандартного редактора, вісуал студіо включає додаткові інструменти коду, графічні дизайнери, компілятори та інші корисні функції задля покращення процесу розробки програм. Серед ключових вікон та функціональних можливостей Visual Studio слід вказати наступне, що існують такі основні можливості:

- Керування файлами коду. Безпосередньо це вікно знаходиться в верхньому правому куті, який ще має назву провідник рішень, який дає можливість керувати та переміщатися між файлами коду, що робить процес дуже легким у використанні. Solution Explorer може допомогти впорядкувати ваш код, згрупувавши файли в рішення та проекти.
- В центри редактора відображається вміст файлу, де існує можливість редагування коду, створення інтерфейсу користувача тощо.
- Можлива інтеграфія з Git. Де можна відстежувати робочі елементи та поширювати код з іншими.

Розглянемо деякі популярні функції Visual Studio, за допомогою яких можна покращити продуктивність під час розробки програм та які включають в себе:

- Завитки та швидкі дії. Завитки — це хвилясті підкреслення, які сповіщають про помилки або потенційні проблеми в коді під час введення. Ці візуальні підказки допоможуть вам негайно виправити проблеми, не чекаючи виявлення помилок під час збірки або виконання. Якщо навести курсор на завитку, можна побачите більше інформації про помилку. На лівому полі також може з’явитися лампочка, яка показує швидкі дії, які можна виконати, щоб виправити помилку.
- Рефакторинг. Це деякі операції, такі як: інтелектуальне перейменування змінних, вилучення рядків, зміна порядку параметрів методу тощо.

- IntelliSense. В більшості тут мається на увазі набір функцій, з інформацією, яка нам відображається і несе в собі вміст про код, і в деяких випадках, записують невеликі фрагменти коду. Це як мати базову документацію в редакторі, тож не доведеться шукати інформацію про типи в іншому місці.

Це найголовніші можливості, які потрібно знати та пам’ятати про них під час написання програми в середовищі розробки.

Вбудована підтримка, керування джерелами – це те що не включає в себе Visual Studio, натомість існує дві альтернативи способу інтеграції систем керування кодом з IDE [19]. Source Control VSPackage може надати власний налаштований інтерфейс користувача. А саме, плагін керування кодом, за допомогою використання інтерфейсу керування вихідних кодом від Microsoft (MSSCCI) надає необхідний набір функцій, які згодом використовуємо для реалізації різноманітних функцій керування кодом, який, в свою чергу, має стандартний інтерфейс користувача Visual Studio [20, 21]. Вперше MSSCCI використовували для інтеграції так званого Visual SourceSafe з Visual Studio версії 6.0, але потім він був відкритий за допомогою Visual Studio SDK. Для Visual Studio .NET 2002 було наявним використовувати MSSCCI версії 1.1, в ту чергу, коли для Visual Studio .NET 2003 – MSSCCI версії 1.2. Вже згодом, у роках 2005, 2008 та 2010 середовище розробки Visual Studio використало MSSCCI версії 1.3, в якій вже була додана підтримка видалення або перейменування, та асинхронного відкриття [21].

Редактор коду.

Редактор Visual Studio надає багато функцій, які полегшують написання коду та тексту та керування ним. Це дає можливість розгортати та згортати різні блоки коду за допомогою контуру. Дізнатися більше про код за допомогою IntelliSense, браузера об’єктів та ієрархії викликів. Знайти код від «перейти», або «перейти до визначення», та «знайти всі посилання», за допомогою цих функій навігаціїї можна найти необхідний код. До цього відноситься функції такі як «вставляти блоки коду з фрагментами коду», а також створювати код за допомогою таких функцій, як Generate From Usage.

Особливості редактора.

- Підсвітка синтаксису. Деякі елементи синтаксису коду та файлів розмітки забарвлені по-різному, щоб розрізняти їх. Наприклад, ключові слова (наприклад, використання в C# та імпорт у Visual Basic) мають один колір, а типи (наприклад, Console та Uri) — іншого кольору. Інші елементи синтаксису також розфарбовані, наприклад рядкові літерали та коментарі. C++ використовує колір, щоб розрізняти типи, перерахування та макроси, серед інших маркерів.
- Помилки та попереджувальні знаки. При додаванні коду або будуванні свого рішення, можна побачити (а) різнокольорові хвилясті підкреслення (відомі як завитки) або (б) в коді з'являються лампочки. Червоні завитки позначають синтаксичні помилки, сині позначають помилки компілятора, зелені позначають попередження, а фіолетові позначають інші типи помилок. Швидкі дії пропонують виправлення проблем і полегшують застосування виправлення.
- Відповідність брекетів. Коли точка вставки розміщується на відкритій дужці у файлі коду, вона і закриваюча дужка виділяються. Ця функція дає негайний зворотній зв'язок про неправильне розміщення або відсутність брекетів. Можна ввімкнути або вимкнути відповідність дужок за допомогою налаштування автоматичного виділення роздільників.
- Візуалізатор структури. Пунктирні лінії з'єднують відповідні дужки у файлах коду, завдяки чому легше бачити відкриваються та закриваючі дужки. Це допоможе швидше знайти код в базі кодів.
- Номери рядків. Номери рядків можуть відображатися на лівому полі вікна коду. За замовчуванням вони не відображаються тощо.

Налагоджувач.

Visual Studio містить налагоджувач, який працює, на рівні джерела, на рівні машини. Налагоджувач Visual Studio дозволяє встановлювати точки зупинки (які дозволяють тимчасово зупинити виконання в певній позиції) і спостерігати (які відстежують значення змінних у міру виконання виконання) [29]. Точки зупину можуть бути умовними, тобто вони запускаються, коли умова виконується. Код можна переступати, тобто запускати один рядок (вихідного коду) за раз [30]. Він може або перейти до функцій для налагодження всередині нього, або переступити

через нього, тобто виконання тіла функції недоступне для перевірки вручну [30]. Налагоджувач підтримує редагування та продовження, тобто дозволяє редагувати код під час його налагодження. Під час налагодження, якщо вказівник миші наводить курсор на будь-яку змінну, її поточне значення відображається у підказці («підказки даних»), де його також можна змінити за бажанням. Під час кодування налагоджувач Visual Studio дозволяє вручну викликати певні функції з вікна інструмента «Негайне». Параметри методу вказуються у вікні "Негайне" [31].

Дизайнер

Visual Studio містить безліч візуальних дизайнерів, які допомагають у розробці додатків. Ці інструменти включають:

- Конструктор Windows Forms.

Конструктор Windows Forms використовується для створення додатків GUI за допомогою Windows Forms. Розміщенням можна керувати, розміщуючи елементи керування всередині інших контейнерів або закріплюючи їх збоку форми. Елементи керування, які відображають дані (наприклад, текстове поле, поле зі списком і подання сітки), можна прив’язати до джерел даних, таких як бази даних або запити. Елементи керування, пов’язані з даними, можна створити, перетягнувши елементи з вікна джерел даних на поверхню проектування [32]. Інтерфейс користувача пов’язаний з кодом за допомогою моделі програмування, керованої подіями. Конструктор генерує код C# або VB.NET для програми.

- Конструктор WPF.

Конструктор WPF, який має кодову назву Cider і представлений був вже в Visual Studio 2008 року. Як і конструктор Windows Forms, він підтримує метафору перетягування. Він використовується для створення інтерфейсів користувача, орієнтованих на Windows Presentation Foundation. Він підтримує всі функції WPF, включаючи прив’язування даних і автоматичне керування макетом. Він генерує код XAML для інтерфейсу користувача. Згенерований файл XAML сумісний з Microsoft Expression Design, продуктом, орієнтованим на дизайнера. Код XAML пов’язується з кодом за допомогою моделі коду позаду.

Веб-дизайнер/розробник.

Visual Studio також містить редактор і конструктор веб-сайтів, які дозволяють створювати веб-сторінки шляхом перетягування віджетів. Він використовується для розробки додатків ASP.NET і підтримує HTML, CSS і JavaScript. Для зв'язування з кодом ASP.NET використовується модель відставання коду. Починаючи з Visual Studio 2008, механізм компонування, який використовується веб-дизайнером, використовується спільно з Expression Web, що припиняється. Існує також підтримка ASP.NET MVC для технології MVC у вигляді окремого завантаження [34] та проекту ASP.NET Dynamic Data, доступного від Microsoft [35].

Дизайнер класу.

Конструктор класів використовується для створення та редагування класів (включаючи їх членів та їх доступ) за допомогою моделювання UML. Конструктор класів може генерувати схеми коду C# і VB.NET для класів і методів. Він також може генерувати діаграми класів із рукописних класів.

Конструктор даних.

Конструктор даних можна використовувати для графічного редагування схем бази даних, включаючи типізовані таблиці, первинні та зовнішні ключі та обмеження. Його також можна використовувати для оформлення запитів у графічному вигляді.

Конструктор карт.

Починаючи з Visual Studio 2008, конструктор зіставлення використовується LINQ to SQL для проектування зіставлення між схемами бази даних і класами, які інкапсулюють дані. Нове рішення з підходу ORM, ADO.NET Entity Framework, замінює та покращує стару технологію.

Видання.

Visual Studio доступна для Windows і Mac. Visual Studio для Mac має багато тих же функцій, що й Visual Studio для Windows, і оптимізована для розробки кросплатформних і мобільних додатків.

Існує три версії Visual Studio: Community, Professional і Enterprise.

2.3 Вибір додаткового інструментарію

Microsoft SQL Server базується на попередніх випусках для розвитку SQL Server як платформи, яка надає вам вибір мов розробки, типів даних, локальних або хмарних середовищ та операційних систем.

Видання.

Роками Microsoft випускав різні версії SQL серверу, з різними наборами функцій, для більшої і малої аудиторії, хто міг і де використовувати, націлений на різних користувачів. Це видання для Підприєвства (SQL Server Enterprise Edition), Стандарний, Мережевий (Інтернет), Бізнес-аналітика, Робочі групи.

- Підприємство.

SQL Server Enterprise Edition відмінність полягає цієї версії в тому, що даний випуск включає в себе механізм, який є основним для баз даних, включає також додаткові служби, набір інструментів, за допомогою яких можна створювати кластера SQL Server і безпосередньо керувати ними. В керування входить керування базами даних (до 524 петабайт) і адресація 12TB пам'яті, разом з цим, ця версія підтримує 640 логічних процесорів (ядер ЦП) 10].

- Стандартний.

Стандартне видання SQL Server це також про механізм, який є основним для баз даних, і служби. Різниця з попередньою версією (SQL Server Enterprise Edition) в тому, що існує підтримка меншу кількість активних екземплярів (кількість вузлів у кластері) і не включає деякі функції високої доступності, такі як гаряче додавання пам'яті (дозволяє додавати пам'ять під час роботи сервера), і паралельні індекси.

- Інтернет.

Веб-видання SQL Server – це варіант із низьким спільним капіталом для веб-хостингу.

- Бізнес-аналітика.

Представлено в SQL Server 2012 і зосереджено на самообслуговуванні та корпоративному бізнес-аналітиці. Він включає в себе можливості Standard Edition та

інструменти Business Intelligence: Power Pivot, Power View, семантичну модель BI, xVelocity in-memory analytics, Data Quality Services Master Data Services [11].

- Робоча група (SQL Server Workgroup Edition).

Ця версія є базовою, безкоштовною, яка включає основні функції, але не входять додаткові служби.

- Експрес (SQL Server Express Edition).

Також безкоштовна, із основним механізмом баз даних. Існують обмеження щодо використання одного тільки процесора 1 ГБ пам’яті, 10 ГБ файлів бази даних, але при цьому немає обмежень в іншому, тільки в пам’яті справа [13]. Ця версія призначена яу заміна MSDE.

Зберігання даних.

Сховище даних — це база даних, яка можна описати як набір таблиць із стовпцями, які є типізованими. При обиранні серверу де зберігатиметься інформація про авторизованих користувачів, важливим фактором була підтримка різних типів програмування, такі як: Char, Integer, Float, Decimal, Varchar, Text, де SQL Server дуже підходить для виконання цієї ролі. Це також про дозвіл визначити та використовувати визначені користувачем складені типи, тобто UDP. Також, у вигляді віртуальних таблиць, представлень (так звані динамічні представлення керування або DMV), даний Microsoft SQL Server дозволяє їх використовувати.

Стосовно фізичного зберігання таблиці, рядки таблиць поділили на серію розділів (від 1 до n, та які є пронумеровані). Користувачем визначається розмір розділу, by default в одному розділі знаходяться всі рядки. Таблиця має таку властивість, як розбиття на кілька розділів, щоб поширити базу даних по кластеру комп’ютера. В кожному цьому розділі рядки зберігаються або в B-дереві, або в структурі купи. Якщо таблиця має пов’язаний кластеризований індекс для швидкого пошуку рядків, рядки зберігаються в порядку відповідно до їх значень індексу, а індекс надає B-дерево. Дані знаходяться в листковому вузлі листів та інших вузлах, що зберігають значення індексів для листових даних, доступних з відповідних вузлів. Якщо індекс не є кластеризованим, рядки не сортуються відповідно до ключів індексу. Індексоване представлення має ту ж структуру зберігання, що і

індексована таблиця. Таблиця без кластеризованого індексу зберігається в невпорядкованій структурі купи. Однак таблиця може мати некластеризовані індекси, щоб дозволити швидко отримати рядки. Структура купи, в деяких ситуаціях має такі переваги в продуктивності перед кластеризованою структурою. І купи, і B-дерева можуть охоплювати декілька одиниць розподілу [24].

Управління буфером.

Управління буфером на сервері SQL Server полягає в тому, що сервер буферує в оперативній пам’яті сторінки, робиться це для мінімізації дискового вводу-виводу. Розміром 8 КБ сторінка може бути буферизована в пам’яті, а набір усіх сторінок, які зараз буферизуються, називається буферним кешем. Обсяг пам'яті, доступний SQL Server, визначає, скільки сторінок буде кешуватися в пам'яті. Буферний кеш керується диспетчером буферів. Сторінка оновлюється на диску диспетчером буферів лише в тому випадку, якщо деякий час не було посилань на кеш у пам’яті. Під час запису сторінок назад на диск використовується асинхронний ввод-вивод, при якому операція вводу-виводу виконується у фоновому потоці, тому іншим операціям не доводиться чекати завершення операції введення-виводу. Кожна сторінка записується разом із контрольною сумою, коли вона записана. Під час читання сторінки її контрольна сума знову обчислюється і збігається із збереженою версією, щоб переконатися, що сторінка не була пошкоджена або не підроблена тим часом [25].

Паралелізм і блокування.

Даний сервер SQL Server надає таку можливість як використовувати одну і ту ж базу даних одночасно для кількох клієнтів. Це говорить контролювання одночасного доступу до даних, які є спільними, задля забезпечення цілісності даних, це пояснюється тим, що при оновленні одних і тих самих даних клієнтів, клієнти можуть намагатися прочитати ці дані, які перебуваються у процесі зміни іншим клієнтом. В SQL Server забезпечує два режими контролю паралельності: песимістичний і оптимістичний. При використанні песимістичного контролю паралельності, SQL Server контролює, за допомогою блокування, одночасний доступ. Ці блокування можуть бути ексклюзивними або спільними. Якщо говорити

за ексклюзивне блокування, можна сказати, що користувачеві надається ексклюзивний доступ до даних, тобто ніхто інший не може отримати доступ до даних, доки утримується блокування. Коли ми говоримо про використання під час зчитування деяких даних (тобто декілька користувачів можуть переглядати дані, які є заблокованими спільним блокуванням), але при цьому не отримують ексклюзивне блокування, тоді ми говоримо за спільне блокування. При цьому, чекати доведеться останнім, в той час поки всі спільні блокування будуть готові до використання, тобто звільнені.

Ще одним важливим інтегрованим середовищем є SQL Server Management Studio (SSMS) призначене для керування будь-якою інфраструктурою SQL, від SQL Server до бази даних SQL Azure. SSMS надає інструменти для налаштування, моніторингу та адміністрування екземплярів SQL Server і баз даних. Використовуйте SSMS для розгортання, моніторингу та оновлення компонентів рівня даних, які використовуються вашими програмами, а також створення запитів і сценаріїв [2].

SSMS — це один із інструментів керування SQL Server, незалежно від вашого місцезнаходження, який використовується для розробки запитів і керування базами даних і сховищами даних через персональний комп’ютер або хмару [3].

Насправді SSMS є інтегрованим середовищем, яке надає інструменти для налаштування, моніторингу та адміністрування екземплярів і баз даних SQL Server [4].

SSMS є Object Explorer є центральною особливістю, яка дозволяє користувачеві переглядати, вибирати та діяти з будь-яким з об’єктів на сервері [5]. Він також поставив окрему версію Express, яку можна було безкоштовно завантажити, однак, яка дозволяє підключатися до будь-якого екземпляра SQL Server Express останнім версім SSMS та цільком керувати ним. Також, прикол в тому, що Корпорація Microsoft включила для старих версій SQL Server зворотну сумісність, що в свою чергу дозволяє новішій версії SSMS підключатися до старих версій SQL Server.

Visual Studio 2010 з використанням WPF також базувалася програмно на оболонці для інтерфейсу користувача, починаючи з одинадцятої версії. У вісімнадцятій версії і новіші засновані на ізольованій оболонкі Visual Studio 2017 [6].

У червні 2015 року Microsoft оголосила про свій намір випускати майбутні версії SSMS незалежно від випусків системи баз даних SQL Server [7].

Окрім основних можливостей SQL серверу від корпорації Microsoft, можна сказати також за безпеку, так як у світі цифрових технологій дані є новою валютою. Кіберзлочинці використовують різноманітні експлойти для крадіжки даних із серверів організації, і одним із найпоширеніших засобів крадіжки даних є атака на платформи баз даних Microsoft SQL Server. Щоб захистити базу даних, системним адміністраторам необхідно знати, як можна виявити, атакувати і навіть зловживати Microsoft SQL Server, якщо перетворити його на «зомбі» — комп'ютер, зламаний хакером і використаний як засіб для атаки на системи та дані. Тобто окрім того, що для авторизованих користувачів буде існувати місце для збереження їх інформації, можна також сказати, що це гарна можливість забезпечити захист даного сервера.

В IT-інфраструктурі, серверна частина завжди відігравала велику роль, так як на сервері зберігається майже вся чутлива і нечутлива інформація, при втраті якої, існують великі наслідки, зазвичай не дуже приємні. А тому, мову SQL варто знати кожному фахівцю з кібербезпеки. Натомість розглянемо, які SQL надає переваги, які роблять цю частину його більш популярним у галузі науки про дані. Можна відзначити, що це дійсно ідеальна мова для запитів, яка дозволяє фахівцям з даних і користувачам спілкуватися з базою даних. Наведемо найкращі переваги:

- Не треба бути великим програмістом, щоб вміти розібратися із базою даних.

SQL не вимагає великої кількості рядків кодування для управління системами баз даних. Натомість можна легко отримати доступ до бази даних і підтримувати її за допомогою простих синтаксичних правил SQL. Ці прості правила роблять SQL зручним для користувача.

- Стандартизована мова.

Що вже говорить за те, що SQL відповідає давно встановленим стандартам ISO і ANSI, які пропонують єдину платформу по всьому світу для всіх користувачів.

- Швидка обробка запитів.

Забезпечення швидкого і ефективного доступу до великої кількості даних з бази даних, яка здійснюється за допомогою запитів SQL. Операції вставки, видалення та оновлення даних також виконуються за менший час.

- Портативність.

Структуровану мову запитів можна легко використовувати на настільних комп'ютерах, ноутбуках, планшетах і навіть смартфонах. Його також можна використовувати з іншими програмами відповідно до вимог користувача.

- Більше ніж один перегляд даних.

Мова SQL також допомагає у створенні кількох переглядів структури бази даних для різних користувачів бази даних.

- Інтерактивна мова.

Це говорить за можливість легко вивчити і зрозуміти мову SQL. Також можливістю є використання цієї мови для спілкування з базою даних, оскільки це проста мова запитів. Ця мова також використовується для отримання відповідей на складні запити за кілька секунд.

Недоліки SQL.

З перевагами SQL він також має деякі недоліки, а саме:

- Вартість. Вартість експлуатації деяких версій SQL висока. Ось чому деякі програмісти не можуть використовувати мову структурованих запитів.

- Складність інтерфейсу. Іншим великим недоліком є те, що інтерфейс мови структурованих запитів є складним, що ускладнює користувачам SQL його використання та керування.

- Частковий контроль бази даних. Правила бізнесу приховані. Таким чином, спеціалісти з обробки даних і користувачі, які використовують цю мову запитів, не можуть мати повний контроль над базою даних.

Висновки за розділом 2

Другий розділ даної роботи включає в себе питання стосовно вибору яких інструментів потребує програмна реалізація, з яким середовищем сумісна обрана мова програмування, та розглянутий сервер бази даних, де зберігатимуться дані про зареєстрованих користувачів. А саме мовою програмування служить C#, як середовище – Visual Studio та Microsoft SQL Server. В ході роботи були розглянуті питання чому саме були обрані дані інструменти та їх переваги та недоліки.

РОЗДІЛ 3
ПРОЕКТУВАННЯ І РОЗРОБКА СИСТЕМИ АВТЕНТИФІКАЦІЇ ЗА КЛАВІАТУРНИМ ПОЧЕРКОМ

3.1 Проектування варіантів використання

Одним із важливих кроків при розробці програми є її проектування. Для цього існує представлення взаємодії користувачів із системою, де можна побачити деякий взаємозв’язок, в свою чергу в яких користувач бере участь, а побачити можна це «проектування» за допомогою існуючої діаграми використання, про яку піде розмова в цій роботі. Говорячи за діаграму використання, існує разом з цим, діаграма випадків використання яка в свою чергу ідентифікує різні типи користувачів системи та різні випадки використання, і однією із можливостей є те, що здійснюється супровід з іншими типами діаграм. Тобто в уніфікованій мові моделювання (UML) діаграма варіантів використання може узагальнити деталі користувачів системи та їхню взаємодію з системою. Щоб створити його, використовуватимемо набір спеціалізованих символів і сполучників. Це потрібно для того, щоб можна було явно побачити будь-які взаємозв’язки, при плануванні розробки будь-якої програми.

Це допомагає на високому рівні є забезпечити огляд системи, де мається на увазі, що "схеми використання - це принципи майбутньої системи".

Діаграма варіантів використання не містить багато деталей — наприклад, вона не може моделювати порядок виконання кроків. Натомість правильна діаграма варіантів використання зображує високорівневий огляд взаємозв’язків між варіантами використання, акторами та системами.

Елементами діаграми використання є:

- Актори: вони є користувачами із взаємодією в системи, тобто з самою системою. Організація, людина, або стороння система, яка взаємодіє з системою або додатком – це ті, хто можуть виступати в ролі актора. В цілому це зовнішні об’єкти

в програмі, які хоч якось використовують дані. Як приклад, при розробці методу автентифікації, в ролі актора виступає користувач, а саме я, як студентка та автор цієї роботи.

- Система: це вже комплексні дії, які є послідовними та конкретними і обов’язково має бути зв’язок із цією системою, іншими словами – це сценарій.
- Цілі: Те до чого ми прагнемо – результат, який ми хочемо отримати, який є кінцевим більшості випадків використання. А вже щоб досягнути досягненої мети, ми маємо побудувати зрозумілу для всіх діаграму, тоді можна вважати її успішною.

Діаграму варіантів використання, яка описує можливі дії користувача в системі для автентифікації користувача до системи у веб-просторі зображено на рисунку 3.1.

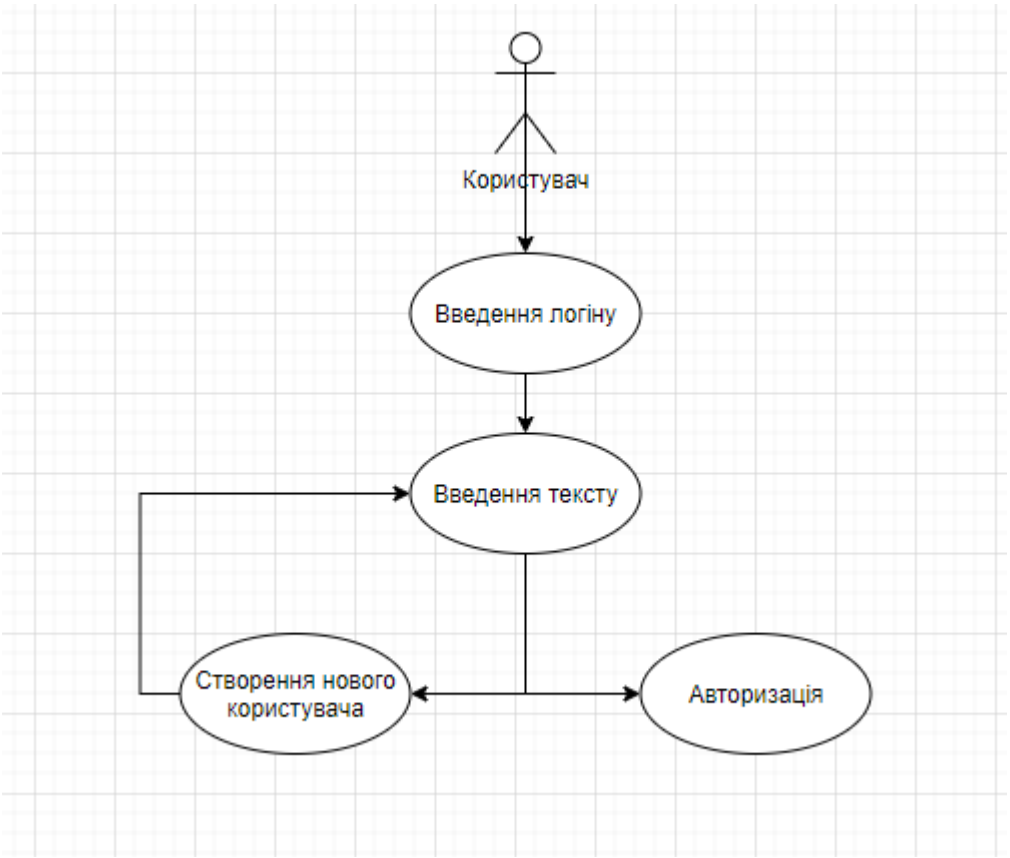

Рисунок 3.1 — Діаграма варіантів використання

3.2 Проектування внутрішньої будови

Діаграма класів уніфікованої мови моделювання (UML) в програмній інженерії —це тип діаграми UML, який описує систему шляхом візуалізації різних типів об’єктів всередині системи та видів статичних зв’язків, які існують між ними. Він також ілюструє операції та атрибути класів.

Також разом з цим, діаграма класів, яка показує будь-якої об'єктно-орієнтованої системи будівельні блоки. Зазвичай, зображується статичний вигляд моделі (або частини моделі), при цьому має бути описані атрибути цієї моделі та можлива поведінка, та не деталізувати методи для виконання операцій. Ці діаграми є корисними, які є ілюстраціями відносин, але вже між класами та інтерфейсами. Для відображення спадкування, складу чи використання та зв’язків є цінними узагальнення, агрегації та асоціації є цінними відповідно. На схемі ці класи представлені вікнами, що містять три відділення:

- Назва класу (у верхньому направлення).
- Атрибути класу. Перша буква маленька, вирівняні за лівим краєм.
- Операції, які може виконувати клас. Також перша буква маленька, і вирівняні за лівим краєм.

При проектуванні системи багато класів ідентифікуються і групуються в схему класів, це ж дуже корисним для визначення статичних відносини між класами. При детальному моделюванні класи концептуального проектування часто поділяють на підкласи.

Залежність використовується для моделювання широкого діапазону залежних зв’язків між елементами моделі. Зазвичай він використовується на початку процесу проектування, коли відомо, що між двома елементами є якийсь зв’язок, але ще зарано знати, що це за взаємозв’язок. Пізніше в процесі проектування залежності будуть стереотипними (доступні стереотипи включають «екземпляр», «трасування», «імпорт» та інші) або замінені на більш специфічний тип з’єднувача.

Для подальшого опису поведінки системи ці діаграми класів можуть бути доповнені діаграмами станів або автоматами станів UML.

Клас асоціації – це конструкція, яка дозволяє з’єднанню асоціації мати операції та атрибути. Асоціації можна назвати, а кінці асоціації можна прикрасити іменами ролей, індикаторами власності, множинністю, видимістю та іншими атрибутами.

Є чотири різні типи асоціацій: двонаправлені, односпрямовані, агрегатні (включаючи складені агрегати) і рефлексивні. Найбільш поширеними є двонаправлені та односпрямовані асоціації.

Агрегації використовуються для зображення елементів, які складаються з менших компонентів. Зв’язки агрегації показані білою ромбоподібною стрілкою, яка вказує на цільовий або батьківський клас [7].

В UML він представлений графічно у вигляді порожнистого ромба на класі хоста, з'єднаного з класом хоста лінією. Більш сильна форма агрегації - композиційна агрегація - показана чорною ромбоподібною стрілкою і використовується там, де компоненти можуть бути включені максимум в одну композицію за раз. Якщо батьківський елемент складеної агрегації видаляється, зазвичай разом із ним видаляються всі його частини; однак частину можна окремо видалити з композиції без необхідності видаляти всю композицію. Приклад: бібліотека та студенти. Тут студенти можуть існувати без бібліотеки, а зв’язок між студентами та бібліотекою є сукупністю.

Графічне представлення узагальнення UML — це форма порожнистого трикутника в кінці суперкласу рядка (або дерева рядків), який з’єднує його з одним або кількома підтипами.

Графічне представлення реалізації UML — представлена як порожнистий трикутник, який знаходиться вкінці інтерфейсу зі штрихами (інакше дерева ліній), та який грає роль з’єднувача між одним або кількома реалізаторами. Проста стрілка використовується наприкінці інтерфейсу з пунктирами, щоб підключити його до користувача. У діаграмах компонентів використовується графічна умова «м'яч і розетка» (реалізатор розміщує кульку або льодяник, а користувач відображає розетку).

Реалізації можна показати лише на діаграмах класів або компонентів. Реалізації. Якщо простими словами це зв’язки між компонентами, інтерфейсами, більше того між класами, та/або пакетами, вони з’єднують елементи замовника з елементами постачальника. Поняття того, що клас або компонент реалізує операції, які запропонував інтерфейс, і на це вказує саме відношення реалізації між класом або компонентом та інтерфейсом.

Представлення асоціації UML - це лінія, що з'єднує два пов'язані класи.

На рисунку 3.2 зображено діаграму класів програмного продукту.

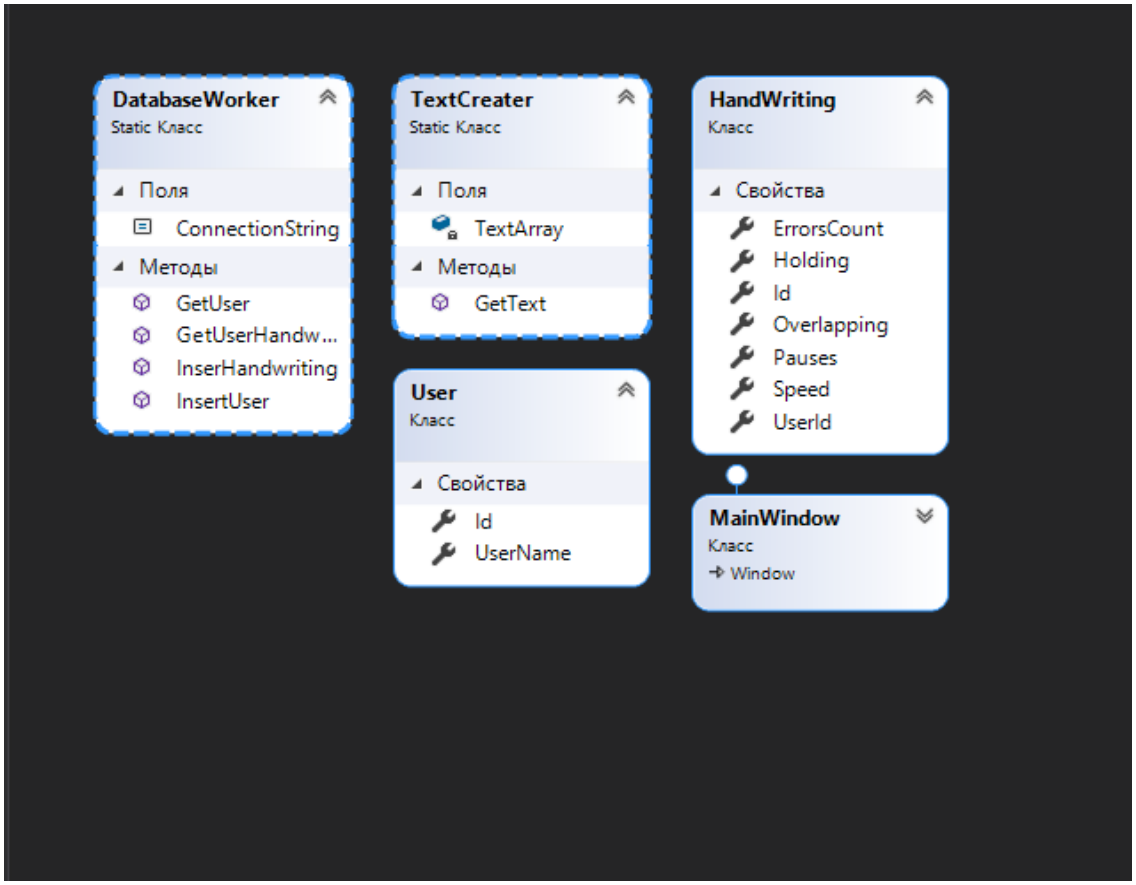

Рисунок 3.2 — Діаграма класів програмного продукту

3.3 Розробка графічного інтерфейсу користувача

Складова графічного інтерфейсу користувача це одна вікона форма, яка в свою чергу викликає декілька інформаційних попапів.

На рисунку 3.3 зображена форма у стартовому форматі.

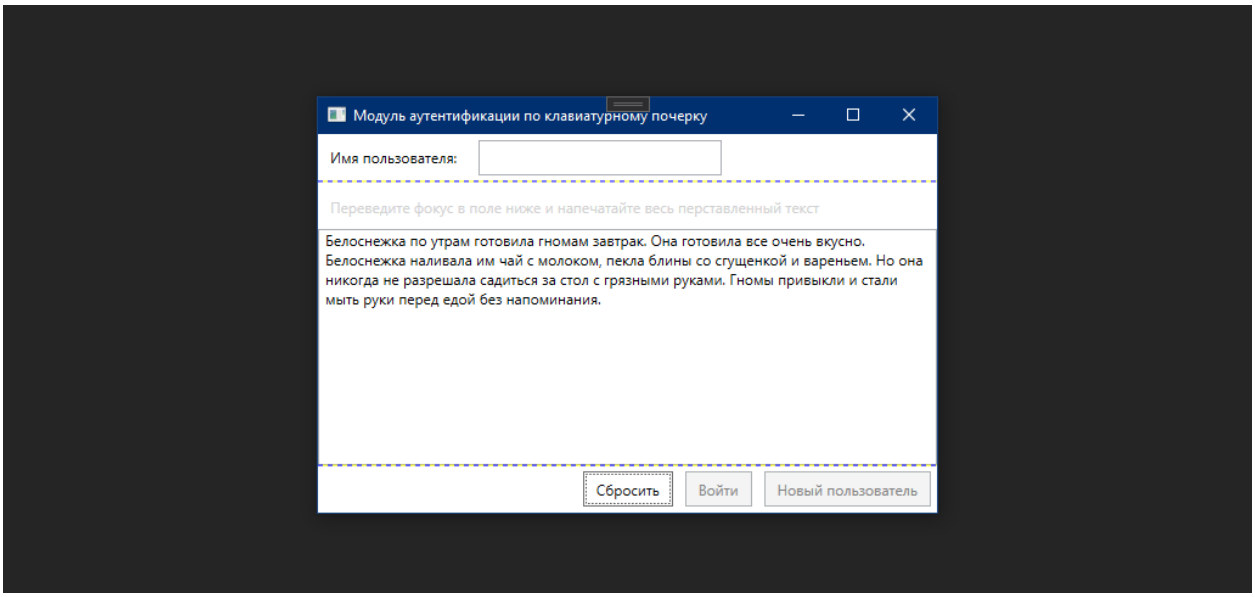

Рисунок 3.3 — Форма у стартовому форматі

На рисунку 3.4 зображена форма у процесі автентифікації.

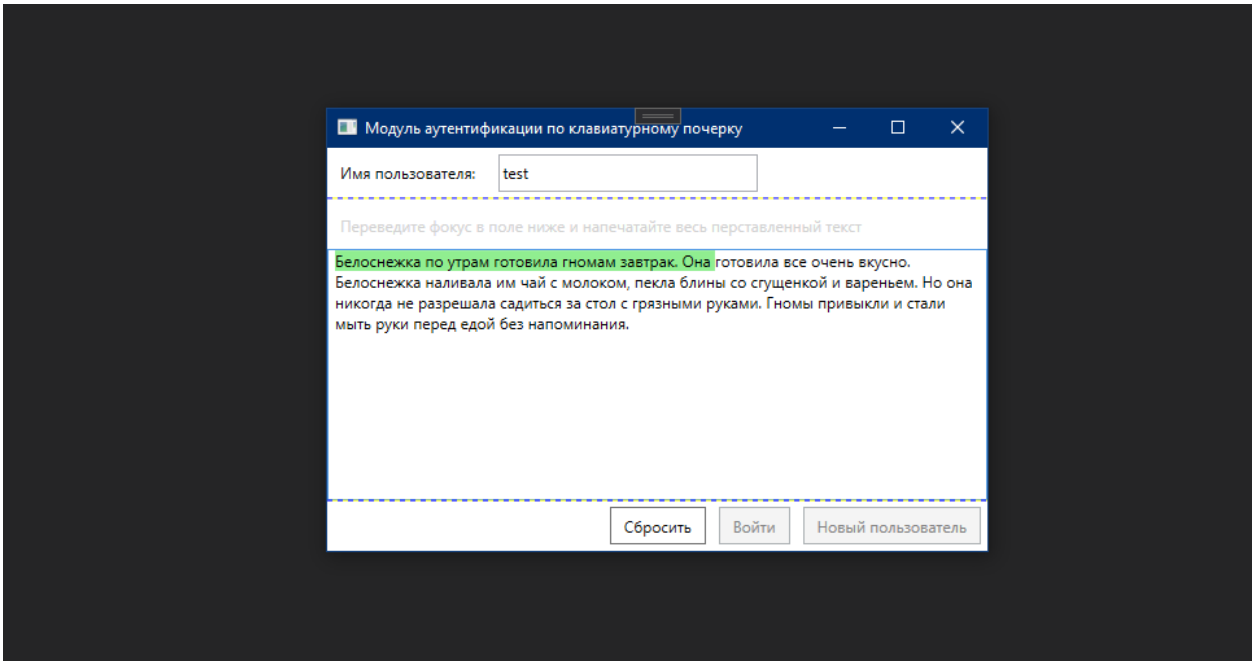

Рисунок 3.4 — Форма у процесі автентифікації

На рисунку 3.5 зображена форма після введення тестового тексту.

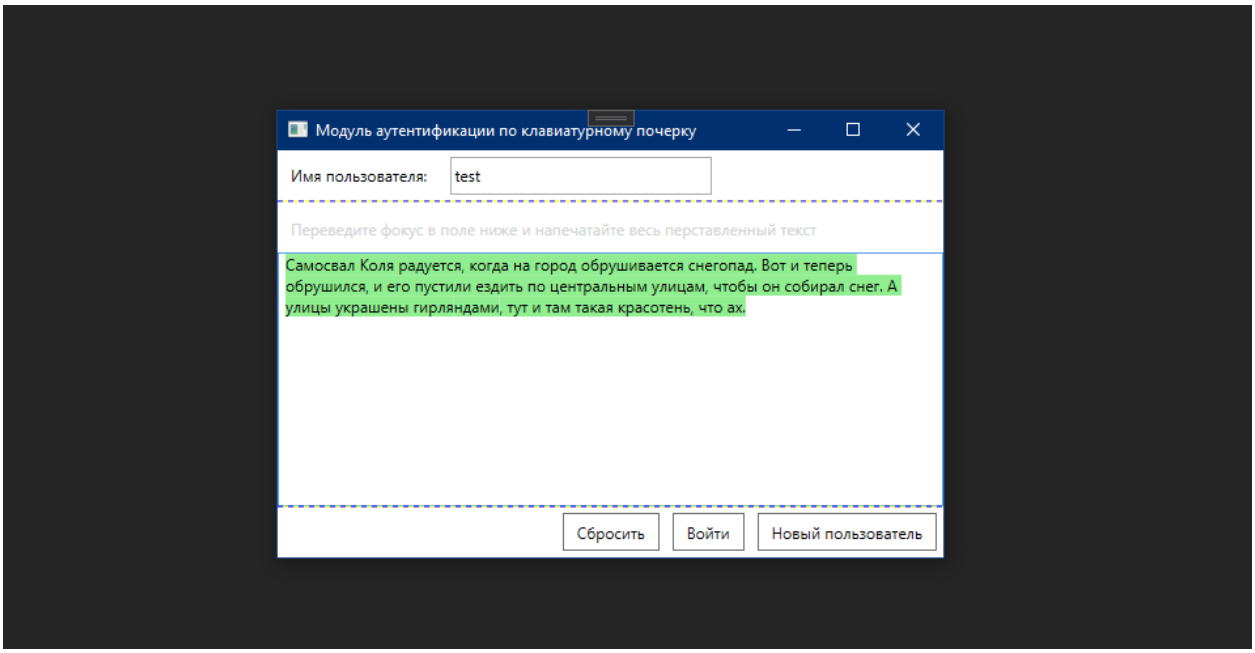

Рисунок 3.5 — Форма після введення тестового тексту

На рисунку 3.6 зображена форма після реєстрації нового користувача.

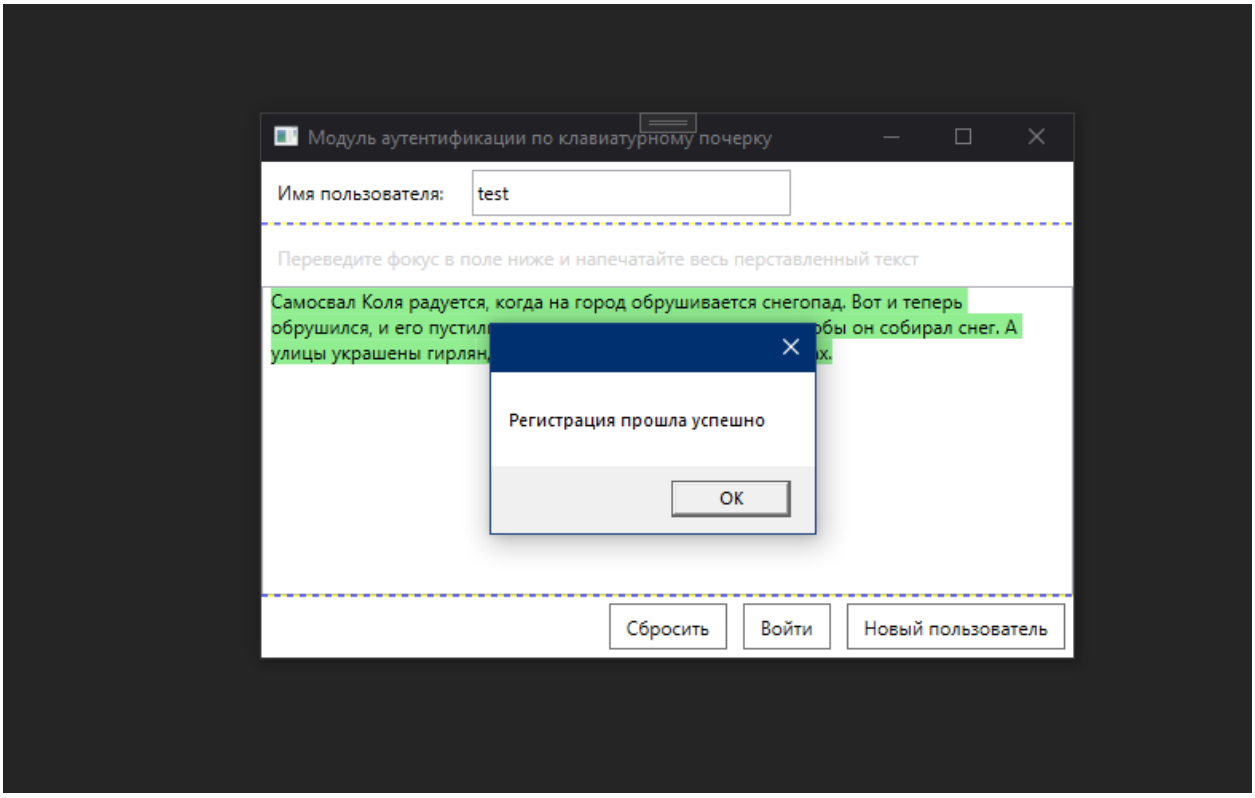

Рисунок 3.6 — Форма після реєстрації нового користувача

На рисунку 3.7 зображена форма з непройденою автентифікацією.

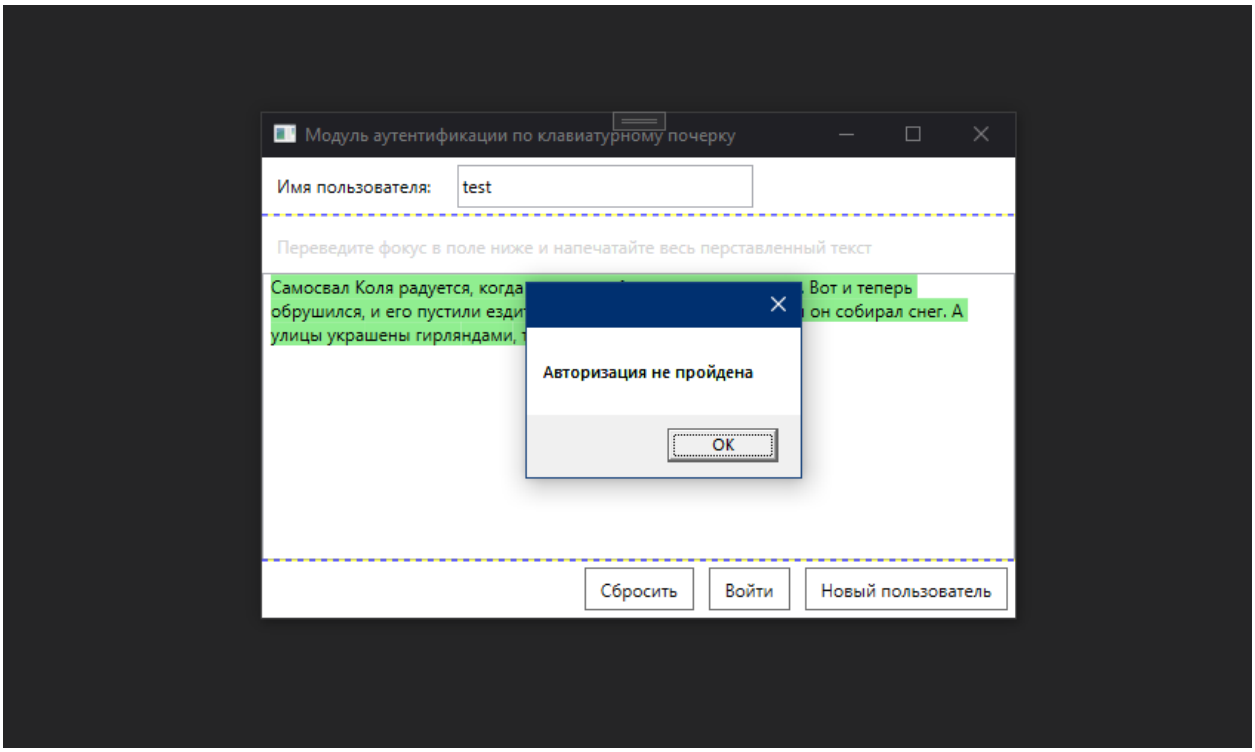

Рисунок 3.7 — Форма відмови в автентифікації

На рисунку 3.8 зображена форма пройденої автентифікації.

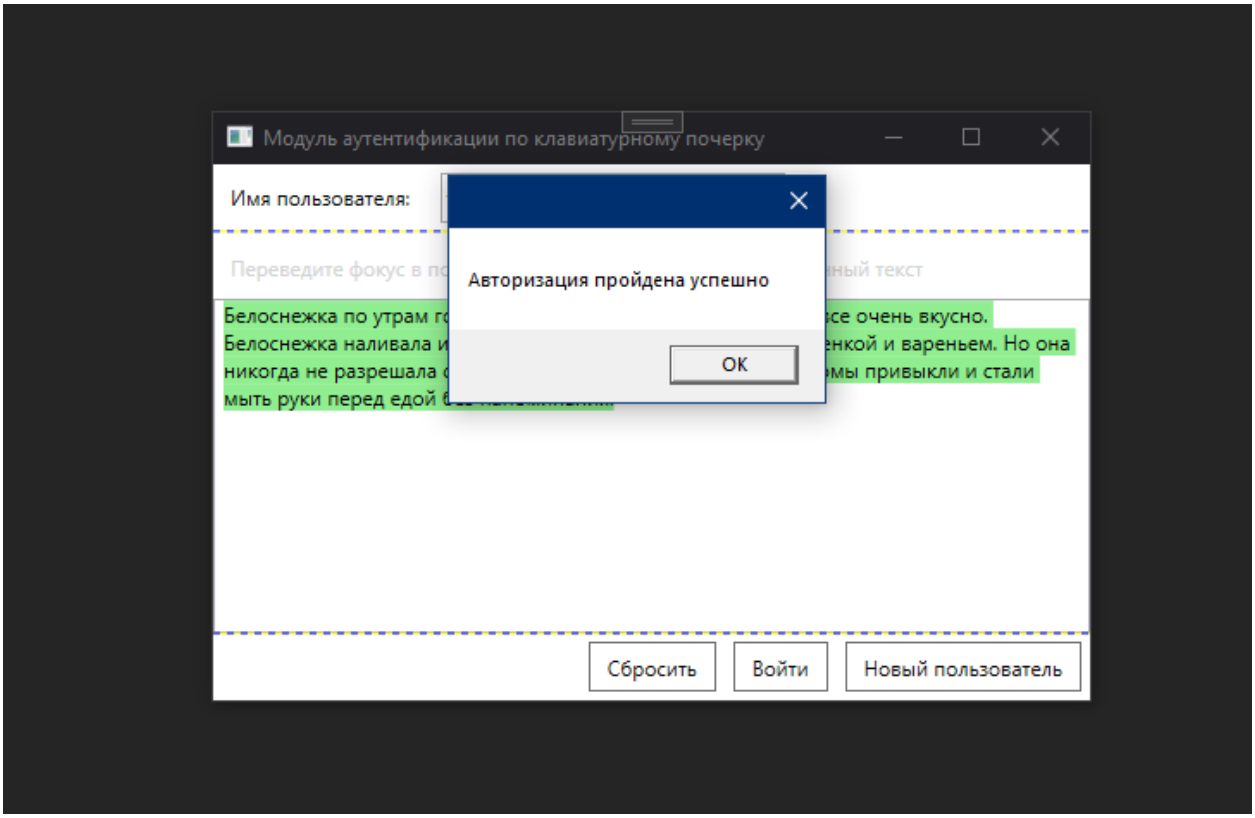

Рисунок 3.8 — Форма пройденої автентифікації

3.4 Тестування

Задля тестування скористаємося методом тестування, який має назву тестування чорного ящика, цей метод тестування для програмного забезпечення, де відбувається перевірка функціональних можливостей додатків реалізованих програмно, при цьому не треба знати внутрішню структуру коду або інші деталі реалізації та внутрішніх шляхів. В основному увага тестування чорного ящика зосереджується на введенні та виводі програмних додатків і повністю базується на так званих вимогах та специфікаціях програмного забезпечення (ПЗ). Також, цей метод називається та відоме як поведінкове тестування або, як ще іноді його називають тестуванням на основі специфікації [1].

Для перевірки за цим методом, нам не треба знати спеціальні знання коду програми, внутрішньої структури або іноді знання програмування загалом [2]. Тому що тестер вже знає, що має робити програмне забезпечення, при цьому може не знати як це воно робить. Як приклад, тестувальнику треба лише знати, що натомість має бути вихідним результатом, але він може не знати яким чином дане програмне забезпечення виробляє вихідні дані [3].

Висновки за розділом 3

В третьому розділі дипломної роботи було здійснено проектування і розробка системи автентифікації за клавіатурним почерком. Перш ніж написати програму, треба було представити її у вигляді діаграми, яким чином користувач взаємодіяв би із системою, для цього була запропонована діаграма варіантів використання та продемонстровано програму візуально. Разом з цим була виконана демонстрація використання класів. І продемонстровано роботу тестування модулю автентифікації з можливістю інтеграції в іншу систему. Виходячи з результатів тестування, можна зробити висновок, що модуль працює коректно і готовий до інтеграції в реальній системи.

ВИСНОВКИ

Основна мета даної роботи — створення модулю автентифікації користувачів за клавіатурним почерком.

Для досягнення поставленої мети було виконано наступні завдання:

- проаналізувати поняття інформаційної безпеки;
- проаналізувати поняття автентифікації;
- проаналізувати поняття клавіатурного почерку;
- розглянути стан питання;
- обрати мову програмування;
- обрати середовище розробки;
- обрати додаткові інструменти;
- провести аналіз варіантів діяльності системи автентифікації за клавіатурним почерком;
- провести проектування внутрішньої будови системи автентифікації за клавіатурним почерком;
- провести розробку графічного інтерфейсу користувача системи автентифікації за клавіатурним почерком;
- провести тестування системи автентифікації за клавіатурним почерком.

Завдяки чіткому виконанню завдань, поставлених на початку роботи, в результаті отримано повноцінний модуль автентифікації за клавіатурним почерком, який може бути інтегрований в реальні системи і використовуватися в реальних умовах.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Joshi, Chanchala; Singh, Umesh Kumar (August 2017). "Information security risks management framework – A step towards mitigating security risks in university network". Journal of Information Security and Applications. 35: 128–137. doi:10.1016/j.jisa.2017.06.006. ISSN 2214-2126.

2. Fletcher, Martin (14 December 2016). "An introduction to information risk". The National Archives. Retrieved 23 February 2022.

3. "SANS Institute: Information Security Resources". www.sans.org. Retrieved 2020-10-31.

4. Daniel, Kent D.; Titman, Sheridan (2001). "Market Reactions to Tangible and Intangible Information". SSRN Electronic Journal. doi:10.2139/ssrn.274204. ISSN 1556-5068. S2CID 154366253.

5. Kerstin., Fink (2004). Knowledge Potential Measurement and Uncertainty. Deutscher Universitätsverlag. ISBN 978-3-322-81240-7. OCLC 851734708.

6. Keyser, Tobias (2018-04-19), "Security policy", The Information Governance Toolkit, CRC Press, pp. 57–62, doi:10.1201/9781315385488-13, ISBN 978-1-315-38548-8, retrieved 2021-05-28

7. Danzig, Richard (1995-06-01). "The Big Three: Our Greatest Security Risks and How to Address Them". Fort Belvoir, VA. Retrieved 18 January 2022.

8. Lyu, M.R.; Lau, L.K.Y. (2000). "Firewall security: policies, testing and performance evaluation". Proceedings 24th Annual International Computer Software and Applications Conference. COMPSAC2000. IEEE Comput. Soc: 116–121. doi:10.1109/cmpsac.2000.884700. ISBN 0-7695-0792-1. S2CID 11202223.

9. "How the Lack of Data Standardization Impedes Data-Driven Healthcare", Data-Driven Healthcare, Hoboken, NJ, USA: John Wiley & Sons, Inc., p. 29, 2015-10-17, doi:10.1002/9781119205012.ch3, ISBN 978-1-119-20501-2, retrieved 2021-05-28

10. Lent, Tom; Walsh, Bill (2009), "Rethinking Green Building Standards for Comprehensive Continuous Improvement", Common Ground, Consensus Building and

Continual Improvement: International Standards and Sustainable Building, 100 Barr Harbor Drive, PO Box C700, West Conshohocken, PA 19428-2959: ASTM International, pp. 1–1–10, doi:10.1520/stp47516s, ISBN 978-0-8031-4507-8, retrieved 2021-05-28

11. Cherdantseva Y. and Hilton J.: "Information Security and Information Assurance. The Discussion about the Meaning, Scope and Goals". In: Organizational, Legal, and Technological Dimensions of Information System Administrator. Almeida F., Portela, I. (eds.). IGI Global Publishing. (2013)

12. ISO/IEC 27000:2009 (E). (2009). Information technology – Security techniques – Information security management systems – Overview and vocabulary. ISO/IEC.

13. Committee on National Security Systems: National Information Assurance (IA) Glossary, CNSS Instruction No. 4009, 26 April 2010.

14. ISACA. (2008). Glossary of terms, 2008. Retrieved from http://www.isaca.org/Knowledge-Center/Documents/Glossary/glossary.pdf

15. Pipkin, D. (2000). Information security: Protecting the global enterprise. New York: Hewlett-Packard Company.

16. B., McDermott, E., & Geer, D. (2001). Information security is information risk management. In Proceedings of the 2001 Workshop on New Security Paradigms NSPW '01, (pp. 97 – 104). ACM. doi:10.1145/508171.508187

17. Anderson, J. M. (2003). "Why we need a new definition of information security". Computers & Security. 22 (4): 308–313. doi:10.1016/S0167-4048(03)00407-3.

18. Venter, H. S.; Eloff, J. H. P. (2003). "A taxonomy for information security technologies". Computers & Security. 22 (4): 299–307. doi:10.1016/S0167-4048(03)00406-1.

19. GOLD, S (December 2004). "Threats looming beyond the perimeter". Information Security Technical Report. 9 (4): 12–14. doi:10.1016/s1363-4127(04)00047-0. ISSN 1363-4127.

20. Parker, Donn B. (January 1993). "A Comprehensive List of Threats To Information". Information Systems Security. 2 (2): 10–14. doi:10.1080/19393559308551348. ISSN 1065-898X.

21. Sullivant, John (2016), "The Evolving Threat Environment", Building a Corporate Culture of Security, Elsevier, pp. 33–50, doi:10.1016/b978-0-12-802019-7.00004-3, ISBN 978-0-12-802019-7, retrieved 2021-05-28

22. Бучик, С. С.; Юдін, О. К.; Нетребко, Р. В. (2016-12-21). "The analysis of methods of determination of functional types of security of the information-telecommunication system from an unauthorized access". Problems of Informatization and Management. 4 (56). doi:10.18372/2073-4751.4.13135. ISSN 2073-4751.

23. Samonas, S.; Coss, D. (2014). "The CIA Strikes Back: Redefining Confidentiality, Integrity and Availability in Security". Journal of Information System Security. 10 (3): 21–45. Archived from the original on 2018-09-22. Retrieved 2018-01-25.

24. "Gartner Says Digital Disruptors Are Impacting All Industries; Digital KPIs Are Crucial to Measuring Success". Gartner. 2 October 2017. Retrieved 25 January 2018.

25. "Gartner Survey Shows 42 Percent of CEOs Have Begun Digital Business Transformation". Gartner. 24 April 2017. Retrieved 25 January 2018.

26. Forte, Dario; Power, Richard (December 2007). "Baseline controls in some vital but often-overlooked areas of your information protection programme". Computer Fraud & Security. 2007 (12): 17–20. doi:10.1016/s1361-3723(07)70170-7. ISSN 1361-3723.

27. Low-voltage switchgear and controlgear. Device profiles for networked industrial devices, BSI British Standards, doi:10.3403/bsen61915, retrieved 2021-05-28

28. Fetzer, James; Highfill, Tina; Hossiso, Kassu; Howells, Thomas; Strassner, Erich; Young, Jeffrey (November 2018). "Accounting for Firm Heterogeneity within U.S. Industries: Extended Supply-Use Tables and Trade in Value Added using Enterprise and Establishment Level Data". Cambridge, MA. doi:10.3386/w25249. S2CID 169324096.

29. "Secure estimation subject to cyber stochastic attacks", Cloud Control Systems, Emerging Methodologies and Applications in Modelling, Elsevier: 373–404, 2020, doi:10.1016/b978-0-12-818701-2.00021-4, ISBN 978-0-12-818701-2, S2CID 240746156, retrieved 2021-05-28

30. 1955-, Nijmeijer, H. (Hendrik) (2003). Synchronization of mechanical systems. World Scientific. ISBN 978-981-279-497-0. OCLC 262846185.

31. "Chapter 1. How students' use of computers has evolved in recent years". dx.doi.org. doi:10.1787/888933277851. Retrieved 2021-05-28.

32. Information technology. Security techniques. Competence requirements for information security management systems professionals, BSI British Standards, doi:10.3403/30342674, retrieved 2021-05-29

33. "Information Security Qualifications Fact Sheet" (PDF). IT Governance. Retrieved 16 March 2018.

34. Ma, Ruiqing Ray (March 2016). "Flexible Displays Come in Many Forms". Information Display. 32 (2): 4–49. doi:10.1002/j.2637-496x.2016.tb00883.x. ISSN 0362-0972.

35. H., Rahim, Noor (March 2006). Human Rights and Internal Security in Malaysia: Rhetoric and Reality. Defense Technical Information Center. OCLC 74288358.

36. Aut, Kramer David Author (2018-09-14). "Nuclear theft and sabotage threats remain high, report warns". Physics Today. doi:10.1063/pt.6.2.20180914a. ISSN 1945-0699. S2CID 240223415. {{cite journal}}: |first1= has generic name (help)

37. Edward., Wilding (2 March 2017). Information risk and security : preventing and investigating workplace computer crime. ISBN 978-1-351-92755-0. OCLC 1052118207.

38. Stewart, James (2012). CISSP Study Guide. Canada: John Wiley & Sons, Inc. pp. 255–257. ISBN 978-1-118-31417-3.

39. "2.2. Productivity growth has been trending down in many sectors". dx.doi.org. doi:10.1787/734700048756. Retrieved 2021-05-28.

40. "Identity Theft: The Newest Digital Attackking Industry Must Take Seriously". Issues in Information Systems. 2007. doi:10.48009/2_iis_2007_297-302. ISSN 1529-7314.

41. Anna, Wendel-Persson, Fredrik Ronnhed (2017). IT-säkerhet och människan : De har världens starkaste mur men porten står alltid på glänt. Umeå universitet, Institutionen för informatik. OCLC 1233659973.

42. Enge, Eric (5 April 2017). "Stone Temple". Cell phones

43. Shao, Ruodan; Skarlicki, Daniel P. (2014). "Sabotage toward the Customers who Mistreated Employees Scale". PsycTESTS Dataset. doi:10.1037/t31653-000. Retrieved 2021-05-28.

44. Kitchen, Julie (June 2008). "7side – Company Information, Company Formations and Property Searches". Legal Information Management. 8 (2): 146. doi:10.1017/s1472669608000364. ISSN 1472-6696. S2CID 144325193.

45. Young, Courtenay (2018-05-08), "Working with panic attacks", Help Yourself Towards Mental Health, Routledge, pp. 209–214, doi:10.4324/9780429475474-32, ISBN 978-0-429-47547-4, retrieved 2021-05-28

46. "Introduction: Inside the Insider Threat", Insider Threats, Cornell University Press, pp. 1–9, 2017-12-31, doi:10.7591/9781501705946-003, ISBN 978-1-5017-0594-6, retrieved 2021-05-28

47. "Table 7.7 France: Comparison of the profit shares of non-financial corporations and non-financial corporations plus unincorporated enterprises". dx.doi.org. doi:10.1787/888933144055. Retrieved 2021-05-28.

48. "How Did it All Come About?", The Compliance Business and Its Customers, Basingstoke: Palgrave Macmillan, 2012, doi:10.1057/9781137271150.0007, ISBN 978-1-137-27115-0, retrieved 2021-05-28

49. Gordon, Lawrence; Loeb, Martin (November 2002). "The Economics of Information Security Investment". ACM Transactions on Information and System Security. 5 (4): 438–457. doi:10.1145/581271.581274. S2CID 1500788.

50. Cho Kim, Byung; Khansa, Lara; James, Tabitha (July 2011). "Individual Trust and Consumer Risk Perception". Journal of Information Privacy and Security. 7 (3): 3–22. doi:10.1080/15536548.2011.10855915. ISSN 1553-6548. S2CID 144643691.

51. Stewart, James (2012). CISSP Certified Information Systems Security Professional Study Guide Sixth Edition. Canada: John Wiley & Sons, Inc. pp. 255–257. ISBN 978-1-118-31417-3.

52. Gillett, John (March 1994). "The cost-benefit of outsourcing: assessing the true cost of your outsourcing strategy". European Journal of Purchasing & Supply Management. 1 (1): 45–47. doi:10.1016/0969-7012(94)90042-6. ISSN 0969-7012.

53. "2.1. Despite strong growth, Austria has lost some ground since the early 1990s". dx.doi.org. doi:10.1787/645173688502. Retrieved 2021-05-29.

54. "Introduction : Caesar Is Dead. Long Live Caesar!", Julius Caesar's Self-Created Image and Its Dramatic Afterlife, Bloomsbury Academic, 2018, doi:10.5040/9781474245784.0005, ISBN 978-1-4742-4578-4, retrieved 2021-05-29

55. Suetonius Tranquillus, Gaius (2008). Lives of the Caesars (Oxford World's Classics). New York: Oxford University Press. p. 28. ISBN 978-0-19-953756-3.

56. Singh, Simon (2000). The Code Book. Anchor. pp. 289–290. ISBN 978-0-385-49532-5.

57. Tan, Heng Chuan. Towards trusted and secure communications in a vehicular environment (Thesis). Nanyang Technological University. doi:10.32657/10356/72758.

58. Johnson, John (1997). The Evolution of British Sigint: 1653–1939. Her Majesty's Stationery Office. ASIN B00GYX1GX2.

59. Willison, Matthew (2018). "Were Banks Special? Contrasting Viewpoints in Mid-Nineteenth Century Britain". SSRN Electronic Journal. doi:10.2139/ssrn.3249510. ISSN 1556-5068. S2CID 169606130.

60. Ruppert, K. (2011). "Official Secrets Act (1889; New 1911; Amended 1920, 1939, 1989)". In Hastedt, G.P. (ed.). Spies, Wiretaps, and Secret Operations: An Encyclopedia of American Espionage. Vol. 2. ABC-CLIO. pp. 589–590. ISBN 9781851098088.

61. "2. THE CLAYTON ACT: A consideration of section 2, defining unlawful price discrimination.", The Federal Anti-Trust Law, Columbia University Press, pp. 18–28, 1930-12-31, doi:10.7312/dunn93452-003, ISBN 978-0-231-89377-0, retrieved 2021-05-29

62. Maer, Lucinda; Gay (30 December 2008). "Official Secrecy" (PDF). Federation of American Scientists.

63. "The Official Secrets Act 1989 which replaced section 2 of the 1911 Act", Espionage and Secrecy (Routledge Revivals), Routledge, pp. 267–282, 2016-06-10, doi:10.4324/9781315542515-21, ISBN 978-1-315-54251-5, retrieved 2021-05-29

64. "Official Secrets Act: what it covers; when it has been used, questioned". The Indian Express. 2019-03-08. Retrieved 2020-08-07.

65. Singh, Gajendra (November 2015). ""Breaking the Chains with Which We were Bound": The Interrogation Chamber, the Indian National Army and the Negation of Military Identities, 1941–1947". Brill's Digital Library of World War I. doi:10.1163/2352-378 6_dlws1_b9789004211452_019. Retrieved 2021-05-28.

66. Duncanson, Dennis (June 1982). "The scramble to unscramble French Indochina". Asian Affairs. 13 (2): 161–170. doi:10.1080/03068378208730070. ISSN 0306-8374.

67. Whitman et al. 2017, pp. 3.

68. "Allied Power. Mobilizing Hydro-Electricity During Canada'S Second World War", Allied Power, University of Toronto Press, pp. 1–2, 2015-12-31, doi:10.3138/9781442617117-003, ISBN 978-1-4426-1711-7, retrieved 2021-05-29

69. Glatthaar, Joseph T. (2011-06-15), "Officers and Enlisted Men", Soldiering in the Army of Northern Virginia, University of North Carolina Press, pp. 83–96, doi:10.5149/9780807877869_glatthaar.11, ISBN 978-0-8078-3492-3, retrieved 2021-05-28

70. Sebag–Montefiore, H. (2011). Enigma: The Battle for the Code. Orion. p. 576. ISBN 9781780221236.

71. Whitman et al. 2017, pp. 4–5.

72. Whitman et al. 2017, p. 5.

73. "Twentieth-Century Wisdom for Twenty-First-Century Communities", Thomas Merton, The Lutterworth Press, pp. 160–184, 2012-04-26, doi:10.2307/j.ctt1cg4k28.13, ISBN 978-0-7188-4069-3, retrieved 2021-05-29

74. Murphy, Richard C. (2009-09-01). "Building more powerful less expensive supercomputers using Processing-In-Memory (PIM) LDRD final report". doi:10.2172/993898.

75. "A Brief History of the Internet". www.usg.edu. Retrieved 2020-08-07.

76. "Walking through the view of Delft - on Internet". Computers & Graphics. 25 (5): 927. October 2001. doi:10.1016/s0097-8493(01)00149-2. ISSN 0097-8493.

77. DeNardis, L. (2007). "Chapter 24: A History of Internet Security". In de Leeuw, K.M.M.; Bergstra, J. (eds.). The History of Information Security: A Comprehensive Handbook. Elsevier. pp. 681–704. ISBN 9780080550589.

78. Perrin, Chad. "The CIA Triad". Retrieved 31 May 2012.

79. Sandhu, Ravi; Jajodia, Sushil (2000-10-20), "Relational Database Security", Information Security Management Handbook, Four Volume Set, Auerbach Publications, doi:10.1201/9780203325438.ch120, ISBN 978-0-8493-1068-3, retrieved 2021-05-29

80. Stoneburner, G.; Hayden, C.; Feringa, A. (2004). "Engineering Principles for Information Technology Security" (PDF). csrc.nist.gov. doi:10.6028/NIST.SP.800-27rA.

A. J. Neumann, N. Statland and R. D. Webb (1977). "Post-processing audit tools and techniques" (PDF). US Department of Commerce, National Bureau of Standards. pp. 11-3--11-4.

81. "oecd.org" (PDF). Archived from the original (PDF) on May 16, 2011. Retrieved 2014-01-17.

82. "GSSP (Generally-Accepted system Security Principles): A trip to abilene". Computers & Security. 15 (5): 417. January 1996. doi:10.1016/0167-4048(96)82630-7. ISSN 0167-4048.

83. Slade, Rob. "(ICS)2 Blog".

84. Aceituno, Vicente. "Open Information Security Maturity Model". Retrieved 12 February 2017.

85. "George Cybenko – George Cybenko's Personal Home Page" (PDF).

86. Hughes, Jeff; Cybenko, George (21 June 2018). "Quantitative Metrics and Risk Assessment: The Three Tenets Model of Cybersecurity". Technology Innovation Management Review. 3 (8).

87. Teplow, Lily. "Are Your Clients Falling for These IT Security Myths? [CHART]". continuum.net.

88. Beckers, K. (2015). Pattern and Security Requirements: Engineering-Based Establishment of Security Standards. Springer. p. 100. ISBN 9783319166643.

89. "Data Privacy and Confidentiality", SpringerReference, Berlin/Heidelberg: Springer-Verlag, 2011, doi:10.1007/springerreference_205286, retrieved 2021-05-29

90. Andress, J. (2014). The Basics of Information Security: Understanding the Fundamentals of InfoSec in Theory and Practice. Syngress. p. 240. ISBN 9780128008126.

91. Boritz, J. Efrim (2005). "IS Practitioners' Views on Core Concepts of Information Integrity". International Journal of Accounting Information Systems. Elsevier. 6 (4): 260–279. doi:10.1016/j.accinf.2005.07.001.

92. Hryshko, I. (2020). "Unauthorized Occupation of Land and Unauthorized Construction: Concepts and Types of Tactical Means of Investigation". International Humanitarian University Herald. Jurisprudence (43): 180–184. doi:10.32841/2307-1745.2020.43.40. ISSN 2307-1745.

93. Kim, Bonn-Oh (2000-09-21), "Referential Integrity for Database Design", High-Performance Web Databases, Auerbach Publications, pp. 427–434, doi:10.1201/9781420031560-34, ISBN 978-0-429-11600-1, retrieved 2021-05-29

94. Pevnev, V. (2018). "Model Threats and Ensure the Integrity of Information". Systems and Technologies. 2 (56): 80–95. doi:10.32836/2521-6643-2018.2-56.6. ISSN 2521-6643.

95. Fan, Lejun; Wang, Yuanzhuo; Cheng, Xueqi; Li, Jinming; Jin, Shuyuan (2013-02-26). "Privacy theft malware multi-process collaboration analysis". Security and Communication Networks. 8 (1): 51–67. doi:10.1002/sec.705. ISSN 1939-0114.

96. "Completeness, Consistency, and Integrity of the Data Model". Measuring Data Quality for Ongoing Improvement. MK Series on Business Intelligence. Elsevier. 2013. pp. e11–e19. doi:10.1016/b978-0-12-397033-6.00030-4. ISBN 978-0-12-397033-6. Retrieved 2021-05-29.

97. "Video from SPIE - the International Society for Optics and Photonics". dx.doi.org. doi:10.1117/12.2266326.5459349132001. Retrieved 2021-05-29.

98. "Communication Skills Used by Information Systems Graduates". Issues in Information Systems. 2005. doi:10.48009/1_iis_2005_311-317. ISSN 1529-7314.

99. "Outages of electric power supply resulting from cable failures Boston Edison Company system". 1980-07-01. doi:10.2172/5083196. OSTI 5083196. Retrieved 18 January 2022.

100. Loukas, G.; Oke, G. (September 2010) [August 2009]. "Protection Against Denial of Service Attacks: A Survey" (PDF). Comput. J. 53 (7): 1020–1037. doi:10.1093/comjnl/bxp078. Archived from the original (PDF) on 2012-03-24. Retrieved 2015-08-28.

101. "Be Able To Perform a Clinical Activity", Definitions, Qeios, 2020-02-02, doi:10.32388/dine5x, S2CID 241238722, retrieved 2021-05-29

102. Ohta, Mai; Fujii, Takeo (May 2011). "Iterative cooperative sensing on shared primary spectrum for improving sensing ability". 2011 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN). IEEE: 623–627. doi:10.1109/dyspan.2011.5936257. ISBN 978-1-4577-0177-1. S2CID 15119653.

103. Information technology. Information security incident management, BSI British Standards, doi:10.3403/30387743, retrieved 2021-05-29

104. Blum, Dan (2020), "Identify and Align Security-Related Roles", Rational Cybersecurity for Business, Berkeley, CA: Apress, pp. 31–60, doi:10.1007/978-1-4842-5952-8_2, ISBN 978-1-4842-5951-1, S2CID 226626983, retrieved 2021-05-29

105. McCarthy, C. (2006). "Digital Libraries: Security and Preservation Considerations". In Bidgoli, H. (ed.). Handbook of Information Security, Threats, Vulnerabilities, Prevention, Detection, and Management. Vol. 3. John Wiley & Sons. pp. 49–76. ISBN 9780470051214.

106. Information technology. Open systems interconnection. Security frameworks for open systems, BSI British Standards, doi:10.3403/01110206u, retrieved 2021-05-29

107. Christofori, Ralf (2014-01-01), "Thus could it have been", Julio Rondo - O.k., Meta Memory, Wilhelm Fink Verlag, doi:10.30965/9783846757673_003, ISBN 978-3-7705-5767-7, retrieved 2021-05-29

108. Atkins, D. (May 2021). "Use of the Walnut Digital Signature Algorithm with CBOR Object Signing and Encryption (COSE)". doi:10.17487/rfc9021. S2CID 182252627. Retrieved 18 January 2022.

109. Le May, I. (2003), "Structural Integrity in the Petrochemical Industry", Comprehensive Structural Integrity, Elsevier, pp. 125–149, doi:10.1016/b0-08-043749-4/01001-6, ISBN 978-0-08-043749-1, retrieved 2021-05-29

110. Sodjahin, Amos; Champagne, Claudia; Coggins, Frank; Gillet, Roland (2017-01-11). "Leading or lagging indicators of risk? The informational content of extra-financial performance scores". Journal of Asset Management. 18 (5): 347–370. doi:10.1057/s41260-016-0039-y. ISSN 1470-8272. S2CID 157485290.

111. Reynolds, E H (1995-07-22). "Folate has potential to cause harm". BMJ. 311 (6999): 257. doi:10.1136/bmj.311.6999.257. ISSN 0959-8138. PMC 2550299. PMID 7503870.

112. Randall, Alan (2011), "Harm, risk, and threat", Risk and Precaution, Cambridge: Cambridge University Press, pp. 31–42, doi:10.1017/cbo9780511974557.003, ISBN 978-0-511-97455-7, retrieved 2021-05-29

113. Grama, J.L. (2014). Legal Issues in Information Security. Jones & Bartlett Learning. p. 550. ISBN 9781284151046.

114. Cannon, David L. (2016-03-04). "Audit Process". CISA: Certified Information Systems Auditor Study Guide (Fourth ed.). pp. 139–214. doi:10.1002/9781119419211.ch3. ISBN 9781119056249.

115. ISACA (2006). CISA Review Manual 2006. Information Systems Audit and Control Association. p. 85. ISBN 978-1-933284-15-6.

116. Kadlec, Jaroslav (2012-11-02). "Two-dimensional process modeling (2DPM)". Business Process Management Journal. 18 (6): 849–875. doi:10.1108/14637151211283320. ISSN 1463-7154.

117. "All Countermeasures Have Some Value, But No Countermeasure Is Perfect", Beyond Fear, New York: Springer-Verlag, pp. 207–232, 2003, doi:10.1007/0-387-21712-6_14, ISBN 0-387-02620-7, retrieved 2021-05-29

118. "Data breaches: Deloitte suffers serious hit while more details emerge about Equifax and Yahoo". Computer Fraud & Security. 2017 (10): 1–3. October 2017. doi:10.1016/s1361-3723(17)30086-6. ISSN 1361-3723.

119. Spagnoletti, Paolo; Resca A. (2008). "The duality of Information Security Management: fighting against predictable and unpredictable threats". Journal of Information System Security. 4 (3): 46–62.

120. Yusoff, Nor Hashim; Yusof, Mohd Radzuan (2009-08-04). "Managing HSE Risk in Harsh Environment". All Days. SPE. doi:10.2118/122545-ms.

121. Baxter, Wesley. Sold out: how Ottawa's downtown business improvement areas have secured and valorized urban space (Thesis). Carleton University. doi:10.22215/etd/2010-09016.

122. de Souza, André; Lynch, Anthony (June 2012). "Does Mutual Fund Performance Vary over the Business Cycle?". Cambridge, MA. doi:10.3386/w18137.

123. Kiountouzis, E.A.; Kokolakis, S.A. (1996-05-31). Information systems security: facing the information society of the 21st century. London: Chapman & Hall, Ltd. ISBN 978-0-412-78120-9.