Київський національний університет імені Тараса Шевченка
Факультет інформаційних технологій
Кафедра програмних систем і технологій

УДК 004.942

На правах рукопису

ВИПУСКНА КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

Тема: «Програмне забезпечення аутентифікації користувачів на основі аналізу клавіатурного почерку»

Спеціальність 121 «Інженерія програмного забезпечення»

ПОЯСНЮВАЛЬНА ЗАПИСКА

Студент:
Касіяненко Дмитро Валерійович

_____
(підпис) (розшифровка підпису) (дата)

Науковий керівник:
Доценко Сергій Іванович,
канд.ф.-м.н., доцент

_____
(посада) (підпис) (розшифровка підпису) (дата)

Допускається до захисту
з питань нормоконтролю
Завідувач кафедри програмних систем і технологій д.т.н. Бичков Олексій Сергійович

_____
(підпис) (розшифровка підпису) (дата)

Київ - 2022

Рішенням Екзаменаційної комісії

випускна кваліфікаційна робота студента

_____

захищена з оцінкою

_____

Голова Екзаменаційної комісії

професор, доктор техн. наук Андрій БОНДАРЧУК

Київський національний університет імені Тараса Шевченка
Факультет інформаційних технологій
Кафедра програмних систем і технологій
Спеціальність 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ:
Завідувач кафедри програмних систем і технологій
_____(О.С.Бичков)
“____”_____ 2022 р.

ЗАВДАННЯ
НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ
Касіяненку Дмитру Валерійовичу
(прізвище, ім’я, по батькові)

1. Тема випускної кваліфікаційної магістерської роботи «Програмне забезпечення аутентифікації користувачів на основі аналізу клавіатурного почерку»
керівник проекту (роботи) Доценко Сергій Іванович, канд.ф.-м.н., доцент
затверджені наказом вищого навчального закладу від „__”_____20__р. №_____
2. Строк здачі студентом закінченої роботи _____
3. Вихідні дані до роботи Інформація про мову програмування написання програмного продукту та інструкція по доступним операційним системам, на яких можна використати програмний продукт
4. Зміст пояснювальної записки (перелік питань, що їх належить розробити)

1) Визначити мету та задачі проекту;
2) Провести огляд предметної області;
3) Провести аналіз існуючих рішень;
4) Описати принципи роботи існуючих рішень;
5) Описати модель об’єкта оптимізації;
6) Розробити модель;
7) Написати код програми;
8) Визначити переваги та недоліків власного алгоритму.

5. Консультанти з роботи із зазначенням розділів роботи, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Розділ 1	канд.ф.-м.н., доцент Сергій ДОЦЕНКО		
Розділ 2	канд.ф.-м.н., доцент Сергій ДОЦЕНКО		
Розділ 3	канд.ф.-м.н., доцент Сергій ДОЦЕНКО		
Розділ 4	канд.ф.-м.н., доцент Сергій ДОЦЕНКО		

6. Дата видачі завдання _____

Керівник _____ Сергій ДОЦЕНКО
(підпис) (розшифровка підпису)

Завдання прийняв до виконання _____ Дмитро КАСІЯНЕНКО
(підпис) (розшифровка підпису)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів роботи	Термін виконання	Примітка
1.	Уточнення постановки задачі	18.10.2021 - 28.10.2021	Виконано
2.	Аналіз літератури	29.10.2021 - 28.11.2021	Виконано
3.	Аналіз існуючих методів та алгоритмів вирішення завдання	29.11.2021 - 09.01.2022	Виконано
4.	Обґрунтування вибору рішення	10.01.2022 - 19.01.2022	Виконано
5.	Побудова моделей основних процесів	20.01.2022 - 02.02.2022	Виконано
6.	Опис розробленого алгоритму	03.02.2022 - 19.02.2022	Виконано
7.	Розробка програмного забезпечення	20.02.2022 - 25.03.2022	Виконано
8.	Тестування програмного забезпечення	26.03.2022 - 17.04.2022	Виконано
9.	Оформлення і друк пояснювальної записки	18.04.2022 - 28.04.2022	Виконано
10.	Оформлення презентації	29.04.2022 - 10.05.2022	Виконано
11.	Отримання рецензії	11.05.2022	Виконано
12.	Затвердження пояснювальної записки роботи завідувачем кафедри	12.05.2022 - 23.05.2022	Виконано
13.	Захист	24.05.2022	Виконано

Студент – магістр _____ Дмитро КАСІЯНЕНКО
(підпис) (розшифровка підпису)

Керівник роботи _____ Сергій ДОЦЕНКО
(підпис) (розшифровка підпису)

АНОТАЦІЯ

Випускна кваліфікаційна магістерська робота: 76 с., 16 рис., 2 табл., 2 додатка, 16 джерел.

Тема: програмне забезпечення аутентифікації користувачів на основі аналізу клавіатурного почерку.

Об’єкт дослідження: дані про специфіку натискань користувачами клавіш, при їх взаємодії з клавіатурою комп'ютера чи ноутбука.

Мета роботи: підвищити надійність, оптимізувати та спростити роботу будь-якої сфери де потрібне використання аутентифікації користувачів, розробивши програмне забезпечення для аутентифікації користувачів на основі аналізу їх клавіатурного почерку.

Предмет дослідження: чисельні характеристики поведінкової інформації користувачів, за допомогою яких можна з високою точністю ідентифікувати користувачів за динамікою їх клавіатурного введення.

Результати дослідження: проведено огляд та аналіз існуючих підходів до аутентифікації користувачів. Розглянуто невирішені проблеми в поставленій задачі і потенційні способи їх вирішення. Запропоновано унікальну математичну модель механізму розпізнавання клавіатурного почерку та програмна реалізація, яку реалізовано на мові програмування Python.

Висновок:

Результатом роботи є створена програма, що застосовується для аутентифікації користувачів, використання якої значно підвищує безпеку будь-якої системи, де є необхідність ідентифікації користувачів.

Ключові слова: аутентифікація, алгоритм, користувач, python, безпека, дослідження.

АННОТАЦИЯ

Выпускная квалификационная магистерская работа: 76 с., 16 рис., 2 табл., 2 приложения, 16 источников.

Тема: программное обеспечение аутентификации пользователей на основе анализа клавиатурного почерка

Объект исследования: данные о специфике нажатий пользователями клавиш при их взаимодействии с клавиатурой компьютера или ноутбука.

Цель работы: повысить надежность, оптимизировать и упростить работу любой сферы, где требуется использование аутентификации пользователей, разработав программное обеспечение для аутентификации пользователей на основе анализа их клавиатурного почерка.

Предмет исследования: численные характеристики поведенческой информации пользователей, с помощью которых можно с высокой точностью идентифицировать пользователей по динамике их клавиатурного ввода.

Результаты исследования: проведен обзор и анализ существующих подходов к аутентификации пользователей. Рассмотрены нерешенные проблемы в поставленной задаче и потенциальные способы их решения. Предложены уникальная математическая модель механизма распознавания клавиатурного почерка и программная реализация, реализованная на языке программирования Python.

Вывод:

Результатом работы является созданная программа, применяемая для проверки подлинности пользователей, использование которой значительно повышает безопасность любой системы, где есть необходимость идентификации пользователей.

Ключевые слова: аутентификация, алгоритм, пользователь, python, безопасность, исследование.

ANNOTATION

Graduation qualification master's work: 76 pages, 16 fig., 2 tables, 2 appendix, 16 sources.

Subject: User authentication software based on keyboard handwriting analysis.

Object of study: data on the specifics of keystrokes by users when they interact with the keyboard of a computer or laptop.

The purpose of the work: improve the reliability, optimize and simplify the work of any area where the use of user authentication is required by developing user authentication software based on the analysis of their keyboard handwriting.

Subject of research: numerical characteristics of users' behavioral information, which can be used to identify users with high accuracy by the dynamics of their keyboard input.

Research results: a review and analysis of existing approaches to user authentication was carried out. Unsolved problems in the task and potential ways to solve them are considered. A unique mathematical model of the keyboard handwriting recognition mechanism and a software implementation implemented in the Python programming language are proposed.

Conclusion:

The result of the work is the created program used to authenticate users, the use of which significantly increases the security of any system where there is a need to identify users.

Keywords: authentication, method, user, python, security, research.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ШІ - штучний інтелект

КІС - комп'ютерна інформаційна система

УПД - управління доступом

ЗІ - захист інформації

НСД - несанкціонований доступ

ПЗ - програмне забезпечення

ВСТУП

Проблема інформаційної безпеки останнім часом стає все більш критичною та актуальною. Завдяки активному розвитку технологій в наш час більшість людей володіють різноманітними електронними пристроями, якими користуються щодня для вирішення своїх потреб – пошуку інформації, купівлі товарів, спілкування, банківських операцій та багато інших. Одним з обов'язкових етапів отримання доступу до майже будь-якого пристрою, ресурсу чи системи для користувачів є процедура проходження аутентифікації, яка є базовою в питанні захисту комп'ютерної інформації. Потрібно враховувати і той факт, що хоча існує багато способів аутентифікації, але найпопулярнішим на сьогоднішній день залишається використання логіну та пароля для ідентифікації користувачів. Але в той самий час даний спосіб є одним із найбільш слабких та ненадійних, оскільки в наш час існує багато різноманітних можливостей для визначення логіну та пароля зловмисниками. Таким чином дослідження, спрямовані на можливі шляхи підвищення надійності аутентифікації за допомогою логіну та пароля, видаються цілком актуальними та обґрунтованими.

Одним з методів для досягнення цієї мети пропонується до звичної парольної аутентифікації додатково використовувати ще і біометричну характеристику користувача, а саме – клавіатурний почерк, який, як і відбитки пальців, є індивідуальним.

Інтерес до можливості використання клавіатурного почерку для аутентифікації користувачів зростає, що відбилося на значній кількості публікацій з даної теми. Можна відзначити публікації таких авторів, як Іванов В.Г., Мазниченко Н.І., Брюхомицький Ю.А, Сапієв А.З. та інші [1, 2, 3, 4].

Метою роботи є оптимізація та підвищення безпеки при аутентифікації користувачів за допомогою розробки програмного забезпечення, яке буде ідентифікувати користувачів та знаходити «аномалії». Така система дозволить

скоротити час, що витрачається на «ручне» прийняття рішення щодо надання доступу користувачеві до системи, , а також підвищити надійність звичайної аутентифікації за допомогою логіну та пароля.

Об’єкт дослідження: дані про специфіку натискань користувачами клавіш, при їх взаємодії з клавіатурою комп'ютера чи ноутбука.

Предмет дослідження: чисельні характеристики поведінкової інформації користувачів, за допомогою яких можна з високою точністю ідентифікувати користувачів за динамікою їх клавіатурного введення.

Для досягнення цілі необхідно вирішити наступні завдання: розробити програмне забезпечення та алгоритм аутентифікації користувачів за їх клавіатурним почерком.

При вирішенні поставлених завдань використовувалися методи математичного моделювання, методи структурного моделювання, методи об'єктно-орієнтованого програмування і проектування.

Практична цінність роботи полягає в можливості використання розробленого програмного забезпечення для прийняття рішень при аутентифікації користувачів і його застосуванні для автоматичної аутентифікації.

Список публікацій:

- «Аналіз методів аутентифікації користувачів з використанням клавіатурного почерку» - 8-ма Східно-Європейська конференція "Математичні та програмні технології Internet of Everything";
- «АУТЕНТИФІКАЦІЇ КОРИСТУВАЧІВ З ВИКОРИСТАННЯМ АНАЛІЗУ КЛАВІАТУРНОГО ПОЧЕРКУ» - XXI Всеукраїнська науково-технічна конференція молодих вчених, аспірантів і студентів «СТАН, ДОСЯГНЕННЯ ТА ПЕРСПЕКТИВИ ІНФОРМАЦІЙНИХ СИСТЕМ І ТЕХНОЛОГІЙ»;

- Стаття «АУТЕНТИФІКАЦІЯ КОРИСТУВАЧІВ НА ОСНОВІ АНАЛІЗУ КЛАВІАТУРНОГО ПОЧЕРКУ» - журнал «Вчені записки Таврійського національного університету імені В. І. Вернадського. Серія: Технічні науки» розміщення у Томі 33 (72) № 3 за 2022.

РОЗДІЛ 1
ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Можливості ідентифікації та аутентифікації користувачів у КІС

Головним фактором, що визначає стан захищеності тієї чи іншої комп'ютерної інформаційної системи (КІС), є ефективність функціонування системи управління доступом суб'єктів доступу до об'єктів доступу (УПД) та захисту інформації (ЗІ). Методи та засоби захисту інформації від несанкціонованого доступу (НСД) необхідні для усунення збитків як матеріальних, так і нематеріальних. Основною проблемою ЗІ в КІС від НСД є розмежування прав доступу та повноважень доступу до необхідної інформації. Основними діями ЗІ від НСД є:

- Не допустити зловмисника до КІС за рахунок розпізнавання законного користувача;
- Використання спеціалізованого програмного забезпечення (ПЗ) ЗІ від НСД;
- Обмежити фізичний доступ до захищеної інформації;
- Використання спеціальних засобів ЗІ від НДД.

Виявлено такі основні засоби забезпечення ЗІ від НСД [38]:

1. Без підтримки фізичними, технічними та програмними засобами та методами, такі засоби, як законодавчі, організаційні та моральні володіють невисокою надійністю та мають низку залежностей від різних побічних та суб'єктивних факторів. Наприклад, від того, як організована робота у компанії.
2. Високу вартість мають такі засоби, як інженерно-технічні, а також фізичні. Для реалізації необхідно організувати регулярний контроль та проведення необхідних регламентованих робіт.
3. Недоліками апаратних та програмних засобів, що виявляються у різних видах даних засобів є: висока вартість, залежність від типу КІС, а також

складність налаштування та обслуговування даної системи. Але у цих засобів, існують вагомі переваги, на відміну від інших засобів. Це надійність, можливість модернізації та модифікації, відсутність впливу побічних та суб'єктивних факторів, універсальність.

Один із основних напрямків програмно-апаратних засобів є УПД. Для його успішного функціонування слід вирішити три основні завдання:

1. Створити умови, за яких неможливо зробити обхід системи управління;
2. Гарантувати ідентифікацію та аутентифікацію користувачів, які здійснюють запит на доступ інформації;
3. Правильно розмежувати права на доступ до захищеної інформації.

Дані завдання реалізуються за допомогою проведення наступних процесів контролю та управління доступом у КІС, що застосовуються до користувача:

1. Ідентифікація – процедура, в результаті виконання якої для суб'єкта ідентифікації виявляється його ідентифікатор, який однозначно ідентифікує цього суб'єкта в інформаційній системі. Для виконання процедури ідентифікації в інформаційній системі суб'єкту попередньо має бути призначений відповідний ідентифікатор (тобто проведено реєстрацію суб'єкта в інформаційній системі) [12].
2. Аутентифікація (встановленням автентичності) – перевірка приналежності суб'єкту доступу пред'явленого ним ідентифікатора та підтвердження його автентичності [5].

На основі цього зроблено висновок, що забезпечення інформаційної безпеки КІС залежить від якості функціонування процесів аутентифікації та ідентифікації користувачів.

1.2. Способи аутентифікації суб'єктів комп'ютерних систем

Розглянемо способи аутентифікації користувачів в комп'ютерній системі. Ці способи можна об'єднати в три групи. До першої групи належать способи аутентифікації, основані на тому, що користувач знає деяку інформацію, що підтверджує його справжність (володіє секретом). До таких способів відносяться парольна аутентифікація і аутентифікація на основі моделі «рукостискання». До переваг цього фактора аутентифікації відносяться простота реалізації, зручність для мобільних користувачів. Недоліками є наявність розвинених методів і засобів організації атак, висока ймовірність порушення користувачами правил вибору і використання секрету (так званий людський фактор), а також те, що достовірність результатів аутентифікації в цьому випадку повністю визначається збереженням секретності аутентифікатора.

Другу групу способів аутентифікації утворюють способи, засновані на тому, що користувач має певний матеріальний об'єкт, який може підтвердити його справжність (наприклад, пластикову карту з ідентифікуючою користувача інформацією). Переваги цього способу аутентифікації полягають в складності його обходу порушником (через складність дублювання пристрою аутентифікації), простоті виявлення втрати пристрою, складності поділу аутентифікатора з іншим користувачем. До недоліків можна віднести більш високу вартість реалізації, ризик втрати або відмови пристрою, складність забезпечення переносимості (для мобільних користувачів).

До третьої групи способів аутентифікації відносяться методи, засновані на таких даних, які дозволяють однозначно вважати, що користувач і є той самий суб'єкт, за якого себе видає (статичні і динамічні біометричні дані: відбитки пальців і райдужна оболонка ока, тембр голосу і особливості клавіатурного почерку і т.п.). До переваг цього типу аутентифікації відносяться зручність для користувача, оскільки не треба нічого запам'ятовувати або носити з собою та висока вірогідність аутентифікації, через унікальність біометричних характеристик і складності їх фальсифікації. Недоліки полягають в додаткових витратах на спеціальне обладнання, ризик розголошення

персональних даних користувачів (через особливості перевірки біометричних даних їх еталони повинні зберігатися в реєстраційній базі даних у відкритому вигляді), можливості відхилення входу зареєстрованого користувача.

Вибір способу аутентифікації в конкретній інформаційній системі визначається виявленими ризиками і допустимими для організації витратами, наприклад, потенційні втрати від слабкостей парольної аутентифікації можуть бути явно більшими витрат на використання смарт-карт.

Також варто пам’ятати. що у комп'ютерних системах з двох- або трьохфакторною аутентифікацією переваги одного способу аутентифікації можуть блокувати недоліки іншого. А оскільки в наш час за різними аналітичними даними найбільш поширеними є парольні системи контролю та управління доступом, це набуває визначального значення, оскільки вони мають велику кількість недоліків.

1.3. Парольні системи аутентифікації

Розглянемо більш докладно аутентифікацію користувачів комп’ютерних систем на основі паролів. Принципова слабкість парольної аутентифікації користувачів визначається тим, що при виборі паролів користувачі повинні керуватися двома, по суті взаємовиключними, правилами – паролі повинні важко підбиратися і легко запам'ятовуватися, оскільки пароль не повинен ніде записуватися, так як в цьому випадку необхідно буде додатково вирішувати завдання захисту носія пароля.

Складність підбору пароля визначається, в першу чергу, потужністю множини символів, що використовуються при виборі пароля. У цьому випадку кількість різних паролів може бути оцінена знизу як $C_p = M^к$. Наприклад, якщо множина символів пароля утворює малі латинські букви, а мінімальна довжина пароля дорівнює 3, то $C_p = 26^3 = 17576$, що зовсім небагато для програмного підбору. Якщо ж множина символів пароля складається з малих і великих латинських букв, а також цифр, і мінімальна довжина пароля дорівнює 6, то $C_p = 62^6 = 56800235584$.

Складність і мінімальна довжина обраних користувачами паролів може встановлюватися її адміністратором при реалізації встановленої для даної системи безпекової політики. Іншими параметрами політики облікових записів при використанні парольної аутентифікації можуть бути:

- максимальний термін дії пароля;
- розбіжність пароля з ім'ям користувача, під яким він зареєстрований в комп'ютерній системі;
- неповторність паролів одного користувача.

Вимога неповторності паролів може бути реалізована за допомогою двох взаємопов'язаних параметрів. По-перше, можна встановити мінімальний термін дії пароля (в іншому випадку користувач, вимушений після закінчення терміну дії свого пароля поміняти його, зможе тут же змінити пароль на старий). По-друге, можна вести список вже використовуваних даних користувачем паролів (максимальна довжина списку при цьому може встановлюватися адміністратором).

Забезпечити складність і мінімальну довжину паролів користувачів можна за допомогою таких дій:

1. Здійснювати перевірку змінюваних паролів користувачів на їх відповідність встановленим в системі правилам.
2. Використовувати тільки випадкові паролі, які автоматично згенеровані через спеціальні програми, що дозволяють встановлювати довжину і склад символів створюваних паролів.
3. Використовувати спеціальні програми, що дозволяють адміністратору комп'ютерної системи виявити «слабкі» паролі користувачів шляхом їх випадкового або словникового підбору для подальшої зміни таких паролів на більш складні.

Недоліком перших двох способів є те, що вони відносяться до способів примусу користувачів до здійснення незручних для них дій. Застосування подібних способів

може дати і зворотний ефект: користувачі почнуть записувати паролі, які вони не можуть запам'ятати, на папері.

Недолік третього способу – можливість його «подвійного» застосування. «Слабкі» паролі можуть виявлятися не для їх заміни, а для здійснення «маскараду». Тому несанкціоноване застосування цього способу адміністратором може накликати на нього підозри з боку керівників організації.

На жаль, забезпечити реальну складність і унікальність кожного знову обраного користувачем пароля навіть за допомогою зазначених вище заходів практично неможливо. Користувач може, не порушуючи встановлених обмежень, вибирати паролі P || 1, P || 2 і т. д., де P – перший пароль користувача, що задовольняє вимогам складності, а || – операція конкатенації.

Таким чином, можна зробити висновок, що використання лише парольної аутентифікації є досить ненадійним в сучасному світі. Для виявлення підміни користувача слід також використовувати додаткові етапи та методи. Набагато ефективніше використання двухфакторної аутентифікації, що набагато знижує ризик від передачі паролів іншим користувачам.

1.4. Біометрична аутентифікація

Біометричні дані дозволяють ідентифікувати особу на основі впізнаваних та перевірених даних, унікальних та конкретних. Біометрична аутентифікація порівнює дані щодо характеристик людини з біометричним «шаблоном» цієї людини для визначення схожості. Спочатку зберігається еталонна модель. Далі збережені дані порівнюються з біометричними даними людини, які підлягають аутентифікації.

Існує два типи біометрії: фізіологічні та поведінкові вимірювання.

Фізіологічні вимірювання – вони можуть бути як морфологічними, так і біологічними. Морфологічні ідентифікатори в основному складаються з відбитків пальців, форми руки, пальця, малюнка вен, ока (райдужки та сітківки) та форми обличчя.

Поведінкові вимірювання – найпоширенішими є:

- розпізнавання голосу;
- динаміка сигнатур (швидкість руху «пера», прискорення, тиск, нахил);
- динаміка натискання клавіші;
- спосіб використання об'єктів;
- ходьба, звук кроків;
- жести тощо.

Різні використовувані методи є предметом постійних досліджень і розробок і постійно вдосконалюються.

Однак не всі типи вимірювань мають однаковий рівень надійності. Фізіологічні вимірювання зазвичай мають перевагу в тому, що залишаються стабільнішими протягом усього життя людини. Наприклад, вони не настільки схильні до впливу стресу, на відміну від ідентифікації за допомогою поведінкових вимірювань.

Основними методами, які використовують статичні біометричні характеристики людини, є ідентифікація по малюнку на пальцях, райдужній оболонці, геометрії особи, сітківці ока, малюнку вен руки, геометрії рук. Також існує сімейство методів, які використовують динамічні характеристики: ідентифікація по голосу, динаміці рукописного почерку, серцевого ритму, ході.

Але усі біометричні системи, які використовують зазначені вище методи, мають складнощі для застосування у виявленні підміни користувача, через обмеження у використанні. До основних можна віднести високу вартість, складність алгоритмів, необхідність проводити процедуру сканування, що не завжди є зручним, довгий час необхідний на обробку результатів, неефективність при наявності перешкод (головний убір, окуляри, інша зачіска), залежність від освітлення та інші.

Тому можна зробити висновок, що найбільш зручним в першу чергу для самого користувача буде використання тих біометричних характеристик, які можна отримати

безпосередньо під час роботи користувача з комп’ютерною системою, а саме клавіатурний почерк, що є поведінковою біометричною характеристикою людини.

РОЗДІЛ 2
ВИКОРИСТАННЯ КЛАВІАТУРНОГО ПОЧЕРКУ В СИСТЕМАХ АУТЕНТИФІКАЦІЇ

2.1. Аутентифікація з використанням клавіатурного почерку

Одним з перших ідею аутентифікації користувачів за особливостями їх роботи з клавіатурою запропонував С.П. Расторгуєв. При розробці математичної моделі аутентифікації на основі клавіатурного почерку користувачів було зроблено припущення, що часові інтервали між натисканнями сусідніх символів ключової фрази і між натисканнями певних поєднань клавіш в ній підкоряються нормальному закону розподілу. Суттю даного способу аутентифікації є гіпотеза про рівність центрів розподілу двох нормальних генеральних сукупностей, яку потрібно підтвердити (отриманих при налаштуванні системи використовуючи характеристики користувача і при його аутентифікації).

Далі наведено варіант аутентифікації користувача по набору ключової фрази (однієї і тієї ж в режимах налаштування і підтвердження автентичності).

Процедура налаштування на характеристики користувача комп'ютерної системи, що реєструється:

- вибір користувачем ключової фрази (її символи повинні бути рівномірно рознесені по клавіатурі);
- набір ключової фрази кілька разів;
- виключення грубих помилок (за спеціальним алгоритмом);
- розрахунок і збереження оцінок математичних очікувань, дисперсій і числа спостережень для тимчасових інтервалів між наборами кожної пари сусідніх символів ключової фрази.

Процедура аутентифікації користувача може проводитися в двох варіантах. Перший варіант процедури аутентифікації:

- набір ключової фрази користувачем кілька разів;
- виключення грубих помилок (за спеціальним алгоритмом);
- розрахунок оцінок математичних очікувань і дисперсій для тимчасових інтервалів між натисканнями кожної пари сусідніх символів ключової фрази;
- рішення задачі перевірки гіпотези про рівність дисперсій двох нормальних генеральних сукупностей для кожної пари сусідніх символів ключової фрази (за спеціальним алгоритмом);
- якщо дисперсії рівні, то рішення задачі перевірки гіпотези про рівність центрів розподілу двох нормальних генеральних сукупностей при невідомій дисперсії для кожної пари сусідніх символів ключової фрази (за спеціальним алгоритмом);
- обчислення ймовірності справжності користувача як відносини числа поєднань сусідніх клавіш, для яких підтверджені гіпотези (пп. 4 і 5), до загальної кількості поєднань сусідніх символів ключової фрази;
- порівняння отриманої оцінки ймовірності з обраним пороговим значенням для прийняття рішення про допуск користувача.

Другий варіант процедури аутентифікації:

- набір ключової фрази один раз;
- рішення задачі перевірки гіпотези про рівність дисперсій двох нормальних генеральних сукупностей для часових інтервалів між натисканнями сусідніх символів ключової фрази;
- якщо дисперсії рівні, то виключення тимчасових інтервалів між натисканнями сусідніх символів ключової фрази, які істотно відрізняються від еталонних (отриманих при налаштуванні);

- обчислення ймовірності справжності користувача як відношення кількості інтервалів, що залишились до загальної кількості інтервалів в ключовій фразі;
- порівняння отриманої оцінки ймовірності з обраним пороговим значенням для прийняття рішення про допуск користувача.

Замість використання постійної для користувача ключової фрази можна проводити аутентифікацію за допомогою набору псевдовипадкового тексту. В цьому випадку клавіатура поділяється на поля і вводиться поняття відстані d_{ij} між клавішами i та j, під яким розуміється число клавіш, розташованих на прямій лінії, що з'єднує i та j. Кнопка i належить полю T, якщо $\forall_j \in m \leq d_{ij}k$. Величину k назвемо ступенем поля T (якщо k = O, то T – окрема клавіша). Позначимо через x_{ij} часовий інтервал між натисканнями клавіш, що належать полям i та j.

Введемо наступні припущення:

- характеристики натискання клавіш одного поля тим ближче один до одного, чим менше k;
- для користувача, що працює двома руками, отримання характеристик клавіатурного почерку можливо за допомогою дослідження роботи тільки з однією половиною клавіатури;
- ключовою фразою може бути будь-який набір символів;
- число полів має бути одним і тим же в режимах налаштування і аутентифікації.

Процедура налаштування при наборі псевдовипадкового тексту:

- генерація та відображення користувачеві тексту з фіксованої множини слів, символи яких максимально розкидані по клавіатурі;
- набір тексту користувачем;
- фіксація і збереження значень x_{ij}, які потім використовуються для розрахунку статистичних характеристик клавіатурного почерку.

Процедура аутентифікації збігається з процедурою аутентифікації, що використовується при наборі ключової фрази.

Достовірність аутентифікації на основі клавіатурного почерку користувача нижче, ніж при використанні його біометричних характеристик.

Однак цей спосіб аутентифікації має і свої переваги:

- можливість приховування факту застосування додаткової аутентифікації користувача, якщо в якості ключової фрази використовується парольна фраза, що вводиться користувачем;
- можливість реалізації даного способу тільки за допомогою програмних засобів (зниження вартості коштів аутентифікації).

2.2. Методи ідентифікації та аутентифікації за комп'ютерним почерком

У задачі динамічної біометричної аутентифікації (ДБА) користувача за комп’ютерним почерком (КП) основним етапом є аналіз та обробка первинних даних. Після цієї операції вхідний потік інформації ділиться на ряд характеристик, які відображають ті чи інші якості аутентифікуємого користувача. Далі ці ознаки дозволяють отримати ряд унікальних характеристик людини.

2.2.1. Опис біометричні характеристики користувача для аутентифікації

Першим етапом обробки інформації є фільтрація. На цьому етапі з потоку інформації видаляється непотрібна інформація («службові» клавіші – клавіші управління курсором, багатофункціональні клавіші і т.п.). Наступний етап – це формування тестового тексту та виділення двох основних параметрів – таблиці 2.1 та 2.2.

Таблиця 2.1

Характеристики для проходження ідентифікації та аутентифікації за контрольною фразою

№	Характеристика	Опис
1.	Кількість символів	В дужках зазначається чистий розмір тексту, без врахування символів, видалених за допомогою BackSpace
2.	Загальний час	Рахується від моменту натискання першої клавіші до моменту натискання останньої

Після визначається інформація, що відноситься до біометричних зарактеристик (БХ) користувача:

Таблиця 2.2

Біометричні характеристики користувача

№	Біометрична характеристика	Опис
1.	Min пауза	Мінімальна пауза між натисканнями.
2.	Max пауза	Максимальна пауза між натисканнями.
3.	Інтервали між натисканнями клавіш	Середня пауза між натисканнями.
4.	Середній час утримання	Відображає середній час між натисканням та відпусканням клавіші. Дозволяє оцінити різкість натискання.
5.	Швидкість spm	Кількість символів, що вводяться за хвилину
6.	Швидкість нетто	Чиста швидкість введення тексту. Рахується для всіх не видалених символів тексту.
7.	Швидкість wpm	Кількість слів за хвилину.

Продовження таблиці 2.2

Біометричні характеристики користувача

8.	Швидкість брутто+	Швидкість набору, яка враховує видалені символи та натискання клавіші BackSpace. Використовується в оцінці втрат у швидкості, які пов’язані з некоректним вводом та його виправленням.
9.	Швидкість брутто*	При розрахунку цієї швидкості не враховуються наступні натискання: помилково введені символи, клавіша BackSpace, а також час, затрачений на ці натискання. Дозволяє оцінити значення швидкості при введенні заданого тексту, якщо взагалі не було помилок.
10.	Швидкість брутто	Швидкість набору, яка враховує видалені символи. Використовується в оцінці втрат у швидкості, які пов’язані з некоректним вводом.
11.	Втрати через виправлення	Відображає в процентному відношенні, на скільки знижується швидкість через час, затрачений на здійснення помилок та їх корегування.
12.	Ступінь аритмічності при наборі	Ступінь нерівномірності у відсотках. Середнє відхилення паузи між натисканнями від середнього значення. Значення аритмії для текстових розділів, введених без помилок, відображаються в дужках.

Продовження таблиці 2.2

Біометричні характеристики користувача

13.	Кількість виправлень	Кількість символів, яка була видалена клавішою BackSpace. В дужках відображають відсоток виправлених символів по відношенню до тексту.
14.	Відсоток серій виправлень	Кожна група підряд видалених символів рахується за одне виправлення.
15.	Max без виправлень	Розмір максимального фрагменту тексту, набраного без виправлень. В дужках відображають значення у відсотках по відношенню до загального розміру тексту.
16.	Кількість перекриттів між клавішами	Відображає кількість перекриттів клавіш – клавіша не відпущена, но вже натиснута інша клавіша.

2.2.2. Методи класифікації користувачів

Далі йде статистична обробка даних, після якої розраховуються еталонні ознаки користувача. Дані зберігаються у БД. Фактично, відбувається вирішення завдання класифікації незареєстрованого користувача, який надає свої БХ, як вектора V, на легального користувача чи зловмисника. Існують різноманітні методи класифікації [33]:

1. Геометричні – використовуються, коли необхідно обчислити міру близькості вхідних даних (вектор V) до еталонного значення Ve (Евклідова міра, міра Хеммінга та ін);
2. Штучні нейронні мережі (ШНМ);
3. Параметричне навчання класифікатора.

2.3. Огляд існуючих рішень

Наразі на ринку кількість існуючих рішень, які для аутентифікації користувачів використовують клавіатурний почерк, постійно зростає. Розглянемо найбільш популярні з них.

2.3.1. KeyTrac

KeyTrac реєструє лише відносний час тривалості натискання та пауз за кодом кожної клавіші, поки ви вводите текст у текстове поле. Щоб захистити паролі, використовується спеціальний «затуманений» формат, який захищає всі паролі.

Дані збираються за допомогою окремого модулю KeyTrac Recorder і відправляються на сервер, де виконується розрахунок та порівняння еталонною моделлю, яка була побудована раніше. Побудова моделі відбувається при введенні заданих фраз. Для відправки даних на сервер використовується модуль KeyTrac API. Сервер після розрахунків відправляє відповідь у вигляді булевої величини true/false – тобто справжній це користувач чи ні. Для використання програми KeyTrac розробники надають бібліотеку написану на JavaScript, яку можна підключити до веб-сайтів. Також зазначається, що KeyTrac є стійким до зміни мови, яку використовує користувач при введенні, а також до зміни обладнання, на якому працює користувач. Алгоритми які використовуються в роботі програми, а також для побудови еталонної моделі. Яка використовується для класифікації користувачів не називаються. Дане рішення можна використовувати і в електронній комерції, оскільки дані, які збираються від користувачів анонімізуються.

2.3.2. TypingDNA

Досить популярним є комерційне рішення TypingDNA. TypingDNA використовує власну технологію аутентифікації біометричних параметрів набору тексту, поєднану з ключовими фразами високої складності щодо кількості різних

символів, способу їх розподілу по клавіатурі, вертикального та горизонтального ходу тощо.

Одним з методів застосування TypingDNA пропонується його використання у навчальному процесі. Оскільки студенти часто працюють за комп'ютерами, то протягом семестру у фоновому режимі відбувається накопичення їх даних, а саме дані з динаміки роботи з клавіатурою. Далі ці дані використовуються для навчання моделей, які потім використовуються для класифікації студентів в онлайн-системах, де вони здають екзамени, щоб не допустити махінацій. Але це не єдина область застосування даного рішення. Його також пропонують використовувати в онлайн-системах банкінгу та при перевірці транзакцій.

Розробники запевняють, що дане рішення буде мати високу якість, якщо для побудови моделі для користувача було введено на клавіатурі не менше 150 символів. Також для інтеграції TypingDNA розроблено спеціальне API, яке є сумісним з найпопулярнішими мовами програмування.

Для класифікації використовувані пропрієтарні алгоритми машинного навчання та штучного інтелекту, які, звичайно, не розголошуються.

2.3.3. KeystrokeDNA

Ще одним популярним комерційним рішенням є KeystrokeDNA. KeystrokeDNA використовує поведінковий біометричний аналіз ритмів набору тексту, щоб створити унікальні біометричні профілі поведінки користувачів при наборі тексту. Потім адаптивний алгоритм машинного навчання порівнює послідовності друку користувачів (починаючи з 8 символів) з їхніми профілями, щоб підтвердити особисті дані та гарантувати, що доступ надається лише справжнім власникам облікових записів.

Дане рішення може використовуватися як 2-й або 3-й фактор аутентифікації, для підтвердження особи в режимі реального часу, як механізм запобігання шахрайству та як захист від викрадених, втрачених або спільних даних. KeystrokeDNA можна зручно

інтегрувати в будь-який веб-додаток лише за допомогою декількох рядків коду за допомогою наданого API.

РОЗДІЛ 3
РОЗРОБЛЕННЯ МАТЕМАТИЧНОЇ МОДЕЛІ МЕХАНІЗМУ РОЗПІЗНАВАННЯ КЛАВІАТУРНОГО ПОЧЕРКУ

3.1. Набір вхідних даних та його характеристики

Набір даних, який використовується для навчання моделі, було отримано локальним збором даних клавіатурного почерку десяти людей. Для здійснення подальшого аналізу поведінки користувачів фіксуються такі показники, що характеризують динаміку їх роботи з клавіатурою, рис. 3.1:

- код використовуваної клавіші;
- тип події (натискання / відпускання);
- час утримання клавіші;
- часова мітка тривалості пауз між натисканням клавіш.

Ці дані записуються до бази даних, та мають посилання на користувача, якому вони належать.

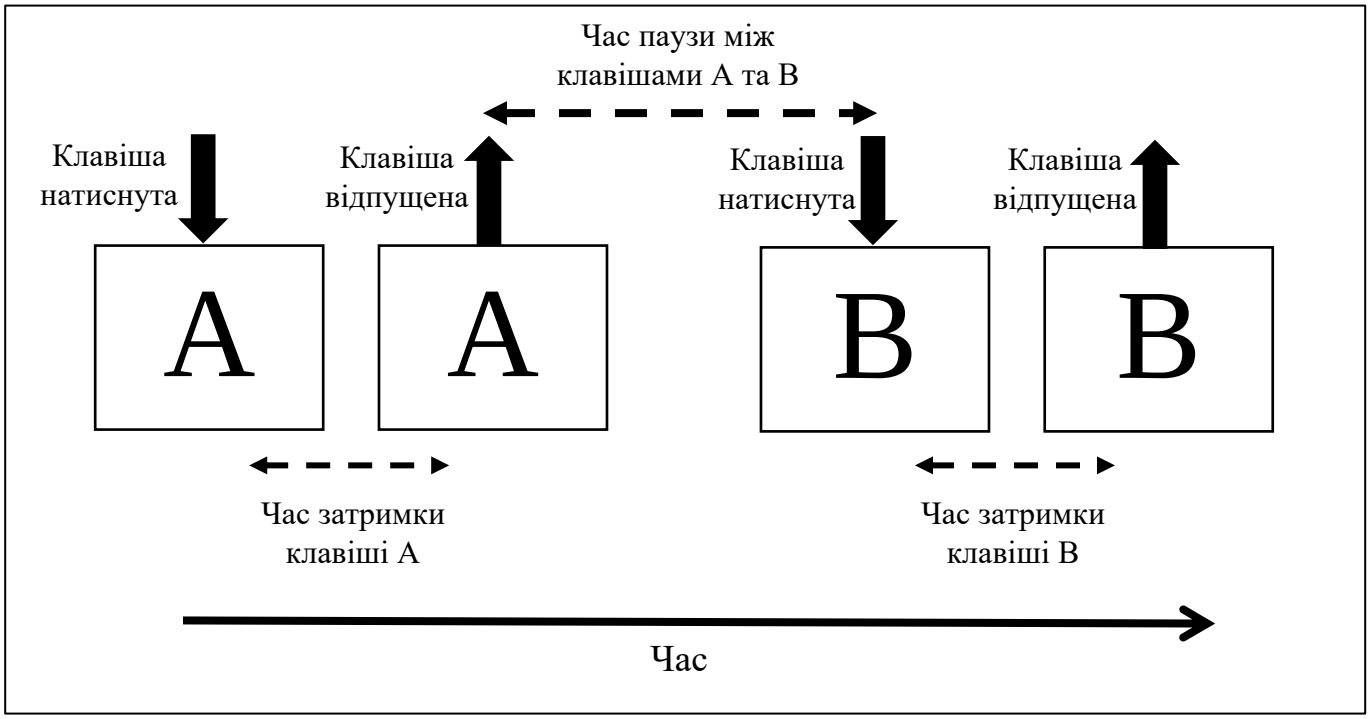

Рис. 3.1. Схема показників для аналізу клавіатурного почерку

3.2. Запропонований алгоритм рішення та математичний опис задачі

Формалізуємо процес машинного аналізу текстового набору системою аутентифікації мережевого сервісу. Нехай процес набору включає у себе $n \in [1; N]$ символів (з пропусками, включно). Позначимо час утримання окремої клавіші як t_n, а проміжок між натисканням клавіш n та $(n+1)$, як t_{n+1}^n, причому, якщо загальний час набору блоку N символів складає T, то зазначений показник може бути розраховано як $T = \sum_{n=1}^{N} t_n + \sum_{n=1}^{N-1} t_{n+1}^n$. Це дозволяє навести наступні біометричні характеристики, на основі яких може бути визначено клавіатурний почерк користувача, рис. 3.2:

- середній час утримання окремої клавіші $\underline{t}_n = \frac{\sum_{n=1}^{N} t_n}{N}$;
- аритмія утримання клавіші, як максимальна різниця у часі утримання окремої клавіші $\Delta t_n = t_n - t_n$;
- середній проміжок між натисканням клавіш $\underline{t}_{n+1}^n = \frac{\sum_{n=1}^{N-1} t_{n+1}^n}{(N-1)}$;
- аритмія набору символів, як максимальна різниця у проміжках між натисканням клавіш $\Delta t_{n+1}^n = t_{n+1}^n - t_{n+1}^n$;
- середня швидкість набору $V = \frac{N}{T}$;
- загальна кількість помилок у наборі N_E;
- загальна кількість перекриттів між клавішами (Overlap Between Keys, OBK) як N_{OBK}.

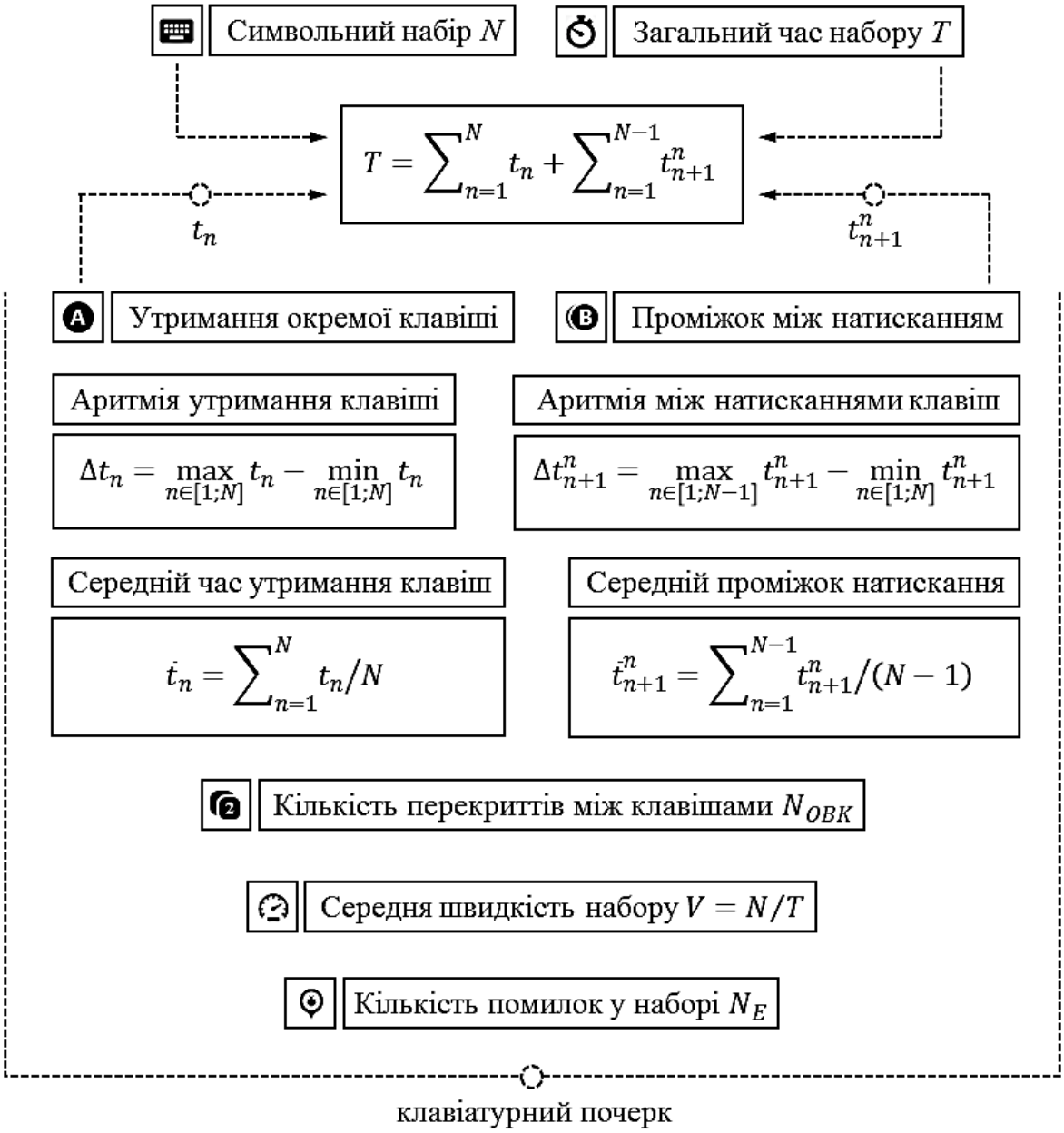

Рис. 3.2. Базова модель виділення набору біометричних параметрів, що відповідають клавіатурному почерку користувача мережевого сервісу

Розширена модель у рамках аналізу відповідних параметрів набору $\{t_n, t_{n+1}^n\}$ може включати визначення залежностей для окремих клавіш або, радше, груп клавіш, що разом зі збільшенням точності процедури аутентифікації надає можливість зменшити навантаження на обчислювальний ресурс. У даному дослідженні пропонується виділити такі групи клавіш:

- група клавіш, що відповідають літерам, яка може бути розбита на дві рівні підгрупи: підгрупу клавіш у центрі клавіатури $\{S_i^0\}$ та підгрупу клавіш у периферійній області – $\{S_i^+\}$, де $i \in [1; I]$;
- група клавіш, що відповідають цифрам $\{S_j^N\}$, де $j \in [0; 9]$;
- група клавіш, що відповідають знакам пунктуації $\{S_m^P\}$, де $m \in [1; M]$;
- група клавіш $\{S_k^A\}$, де $k \in [1; K]$, що відповідають додатковим елементам, як то «Shift», «Ctrl», «Alt», для яких особливо важливо враховувати перекриття клавіш.

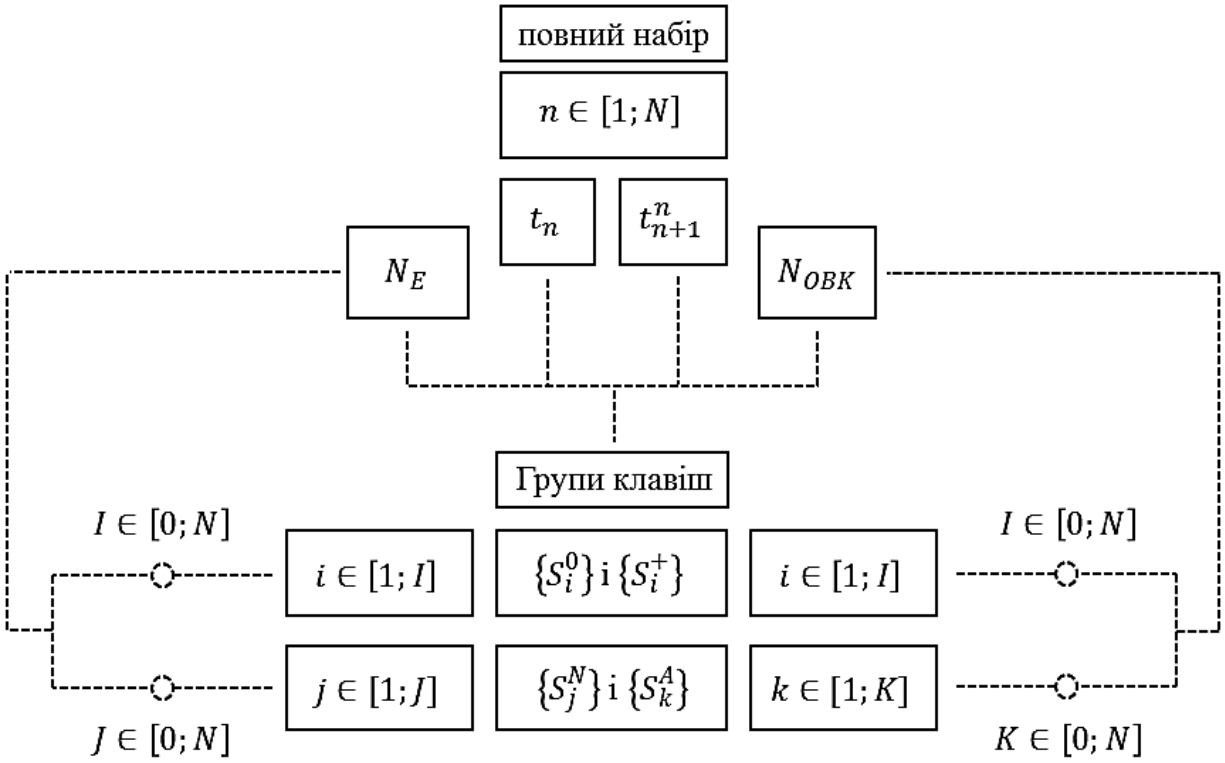

Рис. 3.3. Розширена модель машинного аналізу клавіатурного почерку користувача відповідно груп клавіш

Нарешті, необхідно розглянути задачу адаптації розробленої моделі для вирішення задачі побудови комплексного алгоритму аутентифікації, що базується на введенні пари логін-пароль та доповнюється аналізом біометричних характеристик

користувача мережевого сервісу. Застосування моделі аутентифікації користувача на основі клавіатурного почерку у рамках даної задачі має ряд переваг:

1. Дві послідовності логін і пароль є достатньо короткими і визначеними заздалегідь. Це надає можливість не поділяти символьний ряд на окремі групи відповідно їх розташування на клавіатурі, а аналізувати кожен символ окремо відповідно його положенню у текстових послідовностях пари логін-пароль.

2. Набір коротких послідовностей логін-пароль у рамках навчання користувача сервісу через повторення, а також у процесі безпосереднього користування послугами сервісу відбувається несвідомо, що зумовлює стабільність відповідних біометричних параметрів, що відслідковуються системою аутентифікації.

3. Поєднання двох максимально простих систем аутентифікації надає можливість суттєво збільшити рівень захисту без застосування додаткових апаратних засобів та при мінімальному навантаженні на обчислювальний ресурс апаратно-програмної платформи. Слід також зауважити, що з точки зору користувача процес навчання включає лише серію повторень при введенні логіна і пароля, що дозволить надійно запам’ятати дані аутентифікації облікового запису.

Формалізуємо на математичному рівні процес аутентифікації користувача мережевого сервісу на основі пари логін-пароль, введення якої підлягає машинному аналізу відповідно клавіатурному почерку. Нехай логін складається за $n \in [1; N_{LOG}]$ символів, а пароль – з $n \in [1; N_{PAS}]$, причому після створення зазначеної пари користувач мережевого сервісу має ввести його $r \in [1; R]$ разів. Повторення введення пари логін-пароль надає можливість отримати R груп значень, що відображають кількість похибок при наборі $\{N_{LOG}^{E}(r)\}$ і $\{N_{PAS}^{E}(r)\}$, відповідно, затримок при натисканні окремої клавіші $\{t_{LOG}^{n}(r)\}$ і $\{t_{PAS}^{n}(r)\}$, відповідно, а також проміжки часу між натисканням клавіш $\left\{t_{LOG}^{n|n+1}(r)\right\}$ і $\left\{t_{PAS}^{n|n+1}(r)\right\}$.

Статистичний аналіз для отримання на основі відповідного масиву зразку клавіатурного почерку включає у себе визначення наступних показників:

- середні значення відповідно рівня помилок при введенні логіну та паролю, що розраховуються як $\underline{N}_{LOG}^{E} = \sum_{r=1}^{R}\left(\frac{N_{LOG}^{E}(r)}{R}\right)$ і $\underline{N}_{PAS}^{E} = \sum_{r=1}^{R}\left(\frac{N_{PAS}^{E}(r)}{R}\right)$, відповідно;
- середні значення часу утримання окремої клавіші при введенні логіну та паролю як $\underline{N}_{LOG}^{E} = \sum_{r=1}^{R}\left(\frac{N_{LOG}^{E}(r)}{R}\right)$ і $\underline{N}_{PAS}^{E} = \sum_{r=1}^{R}\left(\frac{N_{PAS}^{E}(r)}{R}\right)$, відповідно;
- середні значення проміжку між натисканням клавіш символьних послідовностей логіну та паролю як $\underline{t}_{LOG}^{n|n+1} = \sum_{r=1}^{R}\left(\frac{t_{LOG}^{n|n+1}(r)}{R}\right)$ і $\underline{t}_{PAS}^{n|n+1} = \sum_{r=1}^{R}\left(\frac{t_{PAS}^{n|n+1}(r)}{R}\right)$, відповідно;
- дисперсія рівня помилок, квадрат якої розраховується як $(\sigma_{LOG}^{E})^2 = \sum_{r=1}^{R}\left(\frac{|\underline{N}_{LOG}^{E} - N_{LOG}^{E}(r)|}{R}\right)$ і $(\sigma_{PAS}^{E})^2 = \sum_{r=1}^{R}\left(\frac{|\underline{N}_{PAS}^{E} - N_{PAS}^{E}(r)|}{R}\right)$, для логіну та паролю, відповідно;
- дисперсія часу утримання окремої клавіші, квадрат яких розраховується як $(\sigma_{LOG}^{T})^2 = \sum_{r=1}^{R}\left(\frac{|\underline{t}_{LOG}^{n} - t_{LOG}^{n}(r)|}{R}\right)$ і $(\sigma_{PAS}^{T})^2 = \sum_{r=1}^{R}\left(\frac{|\underline{t}_{PAS}^{n} - t_{PAS}^{n}(r)|}{R}\right)$ для символьних послідовностей логіну та паролю, відповідно;
- дисперсія у значенні проміжку між натисканням клавіш символьних послідовностей, квадрат яких $(\sigma_{LOG}^{TT})^2 = \sum_{r=1}^{R}\left(\frac{|\underline{t}_{LOG}^{n|n+1} - t_{LOG}^{n|n+1}(r)|}{R}\right)$ і $(\sigma_{PAS}^{TT})^2 = \sum_{r=1}^{R}\left(\frac{|\underline{t}_{PAS}^{n|n+1} - t_{PAS}^{n|n+1}(r)|}{R}\right)$ для логіну та паролю, відповідно.

На основі наборів середніх значень та значень дисперсії через введення математичного очікування E, t-критерію Стьюдента, та значення вірогідності наявності помилок першого роду P_{EI}, а також визначення загальної кількості зразків навчальної

вибірки $l \in [1; L]$ у рамках розробленої математичної моделі процесу аутентифікації можуть бути розраховані інтервали допустимих значень цільових показників, що відповідають клавіатурному почерку, як це показано на рис. 3.4:

- інтервали для допустимої кількості помилок $N_{LOG}^{E}(r) \in \left[N_{LOG}^{E\downarrow}; N_{LOG}^{E\uparrow}\right]$ і $N_{PAS}^{E}(r) \in \left[N_{PAS}^{E\downarrow}; N_{PAS}^{E\uparrow}\right]$ для символьних послідовностей логіну та паролю, відповідно;
- інтервали для допустимого інтервалу часу утримання окремої клавіші $t_{LOG}^{n}(r) \in \left[t_{LOG}^{T\downarrow}; t_{LOG}^{T\uparrow}\right]$ і $t_{PAS}^{n}(r) \in \left[t_{PAS}^{T\downarrow}; t_{PAS}^{T\uparrow}\right]$ для символьних послідовностей логіну та паролю, відповідно;
- інтервали для допустимого проміжку між натисканням клавіш $t_{LOG}^{n|n+1}(r) \in \left[t_{LOG}^{TT\downarrow}; t_{LOG}^{TT\uparrow}\right]$ і $t_{PAS}^{n|n+1}(r) \in \left[t_{PAS}^{TT\downarrow}; t_{PAS}^{TT\uparrow}\right]$ для символьних послідовностей логіну та паролю, відповідно.

Зазначені інтервали дозволяють на кількісному рівні через статистичний аналіз однозначно визначити відповідність клавіатурного почерку окремому користувачу. Для підтвердження відповідності біометричних характеристик має бути виконано повний набір умов, вказаних вище.

Оптимізація алгоритму відбувається, з одного боку через варіювання показників E, t-критерію Стьюдента, L та P_{EI}, а з іншого шляхом розширення роботи з користувачем мережевого сервісу. У першому випадку оптимізація відбувається через пошук глобального екстремуму цільової функції точності машинного аналізу на основі методу градієнтного спуску або аналогічних методів.

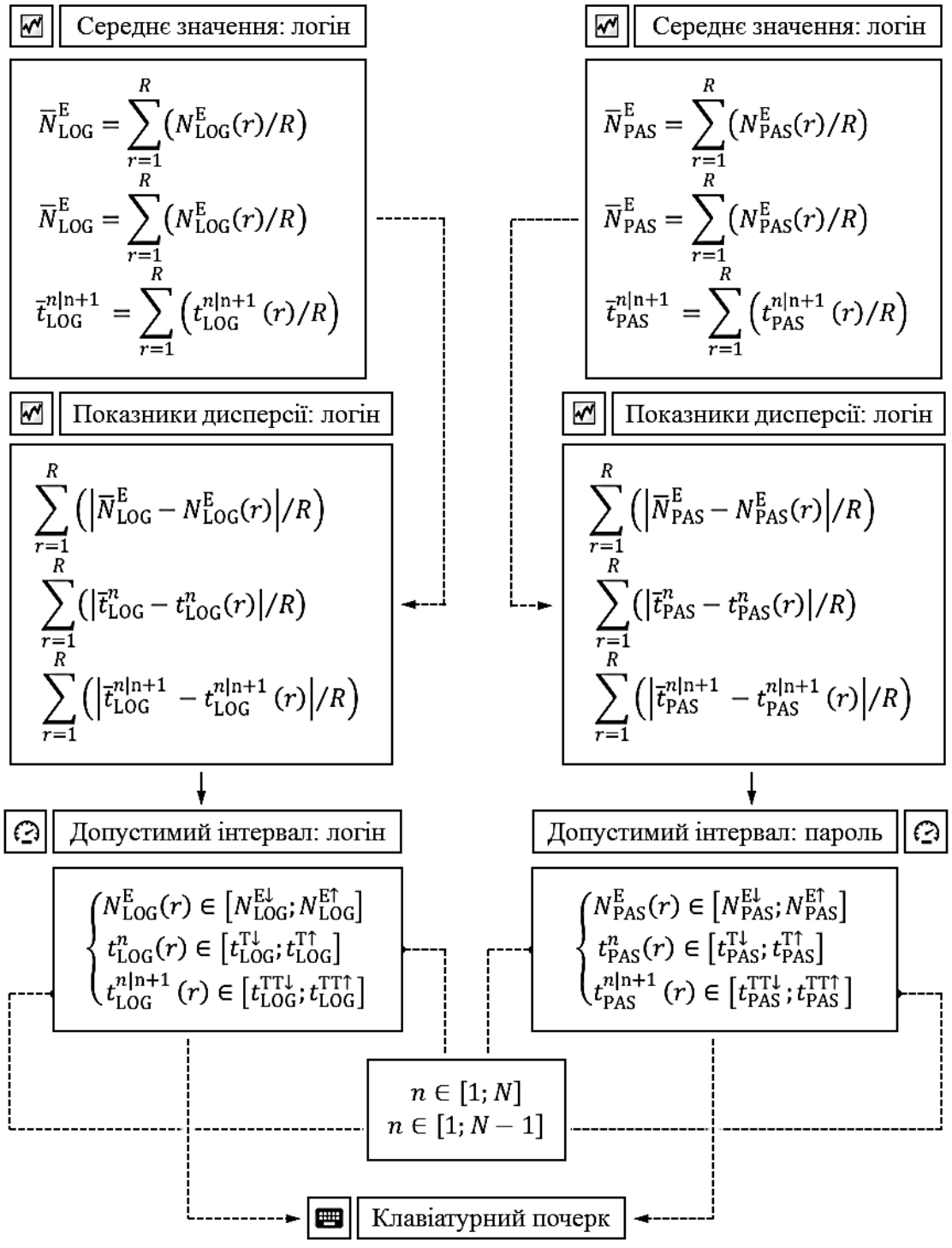

Рис. 3.4. Методика аутентифікації, що базується на введенні пари логін і пароль та машинному аналізі клавіатурного почерку користувача

Другий підхід зумовлює необхідність повторення проходження користувачем процедури повторного введення пари логін-пароль *M* разів поспіль. Це допомагає як додатково уточнити допустимі інтервали, що визначають клавіатурний почерк, так і відслідкувати зміни у клавіатурному почерку, що може відбуватись протягом тривалого часу.

РОЗДІЛ 4
РОЗРОБЛЕННЯ ПРОГРАМИ АУТЕНТИФІКАЦІЇ

4.1. Технології для розробки

При розробці алгоритму класифікації користувачів було вирішено використовувати наступні технології:

- Python 3.10: інтерпретована об'єктно-орієнтована мова програмування високого рівня зі строгою динамічною типізацією. Вибір пав саме на неї, оскільки на Python реалізована велика кількість бібліотек необхідних для розроблення власних моделей та алгоритмів машинного навчання та нейронних мереж.
- JavaScript: це мова, яка дозволяє застосовувати скрипти на web сторінці – кожен раз, коли на web сторінці відбувається щось більше, ніж просто її статичне відображення – відображення контенту, що періодично оновлюється, або інтерактивних карт, або анімація 2D/3D графіки, або прокручування відео в плеєрі, і т.д. – це застосування JavaScript.
- Node.js: це міжплатформне середовище виконання JavaScript із відкритим вихідним кодом, яке працює на движку V8 і виконує код JavaScript за межами веб-браузера. Як асинхронне середовище виконання JavaScript, що керується подіями, Node.js розроблено для створення масштабованих мережевих додатків. Node.js схожий за дизайном на такі системи, як Ruby's Event Machine і Python's Twisted, і під впливом таких систем. Node.js розширює модель подій. Він представляє цикл подій як конструкцію під час виконання, а не як бібліотеку. В інших системах завжди існує блокуючий виклик для запуску циклу подій. Зазвичай поведінка визначається за допомогою зворотних викликів на початку сценарію, а в кінці сервер запускається за допомогою блокуючого виклику, наприклад

EventMachine::run(). У Node.js такого виклику циклу start-the-event немає. Node.js просто входить у цикл подій після виконання сценарію введення. Node.js виходить з циклу подій, коли більше немає зворотних викликів для виконання. Ця поведінка схожа на JavaScript у браузері – цикл подій прихований від користувача.

- MongoDB: програма керування базами даних NoSQL з відкритим вихідним кодом. NoSQL використовується як альтернатива традиційним реляційним базам даних. Бази даних NoSQL досить корисні для роботи з великими наборами розподілених даних. MongoDB – це інструмент, який може керувати документально-орієнтованою інформацією, зберігати або отримувати інформацію. MongoDB підтримує різні форми даних. Це одна з багатьох технологій нереляційних баз даних, які виникли в середині 2000-х під прапором NoSQL – зазвичай для використання в програмах великих даних та інших роботах з обробки даних, які погано вписуються в жорстку реляційну модель. Замість використання таблиць і рядків, як у реляційних базах даних, архітектура MongoDB складається з колекцій і документів. Організації можуть використовувати MongoDB для спеціальних запитів, індексування, балансування навантаження, агрегації, виконання JavaScript на стороні сервера та інших функцій.

4.2. Опис класів і об'єктів програми

Було розроблено два проєкти. Перший проєкт – це веб-сайт для демонстрації роботи системи та збору інформації щодо натискань клавіш та інтеграції зі сервісом аутентифікації.

Другий проєкт – це сервіс, який надає веб-додаткам можливість аутентифікувати своїх користувачів не тільки за допомогою імені користувача та пароля, а також за їх поведінкою. Зокрема, він використовує шаблони натискання клавіш (динаміка

натискання клавіш) для аутентифікації користувачів і таким чином може виявляти аномалії в поведінці введення, виявляючи можливих «самозванців».

Структура першого та другого проєктів зображена на рис. 4.1 та рис. 4.2 відповідно.

Головний файл – server.js. В ньому міститься основна логіка роботи програми: налаштування роутингу та обробників для переходу між сторінками системи, виконання аутентифікації, вказані основні налаштування для класифікаторів та міститься функція запуску.

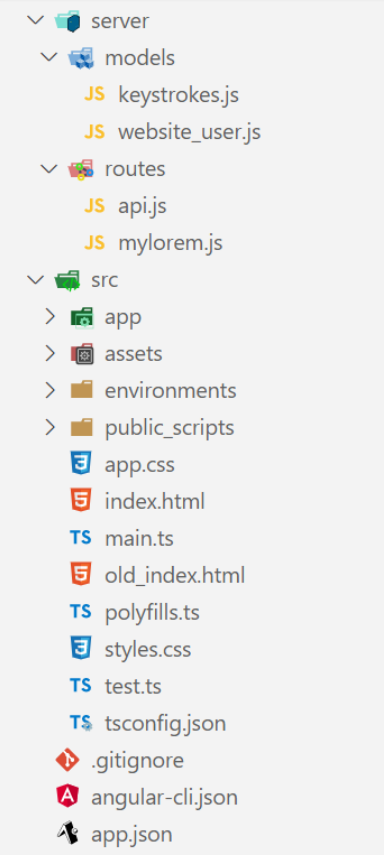

Рис. 4.1. Структура першого проєкту

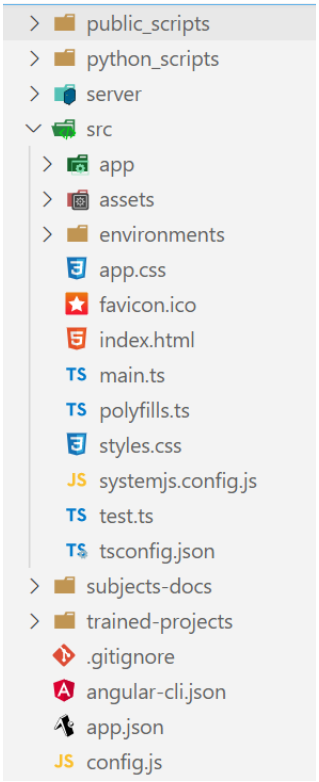

Рис. 4.2. Структура другого проєкту

Усі web-сторінки знаходяться у каталозі app. Це головна сторінка, сторінка для реєстрації, логіну, навчання та налаштування найкращих параметрів для класифікаторів, відображення проєктів.

У каталозі static зберігаються усі статичні ресурси системи, такі як зображення та css-файли стилів.

Каталоги public_scripts та server містять класи в яких знаходиться логіка статистичного класифікатора та класифікатора побудованого з використанням алгоритму k-найближчих сусідів відповідно.

В каталозі database знаходяться скрипти для керування та маніпулювання базою даних, наприклад, для створення таблиць, додавання записів до них, їх модифікація та видалення. Сама база даних знаходиться у кореневому каталозі та має назву database.db.

4.3. Демонстрація роботи користувача з програмою

На рис. 4.3 зображена головна сторінка програми.

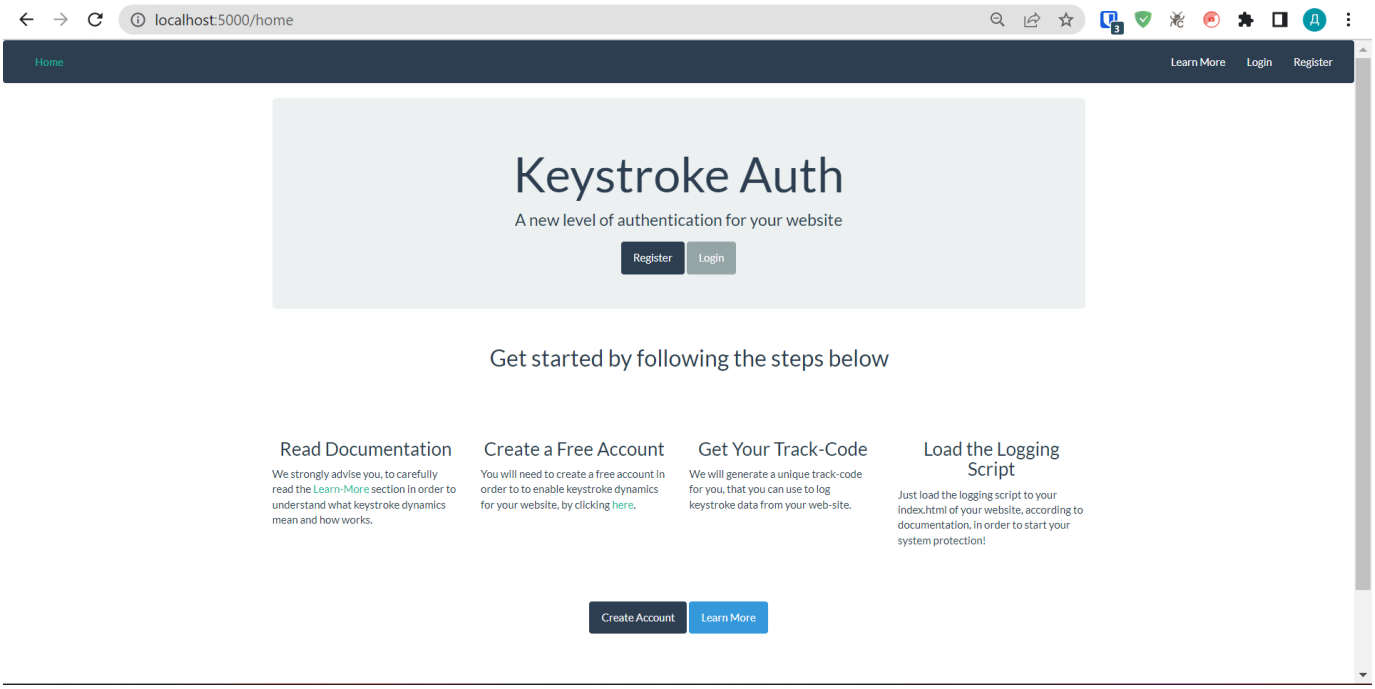

Рис. 4.3. Головна сторінка програми

На головній сторінці доступні наступні опції:

- Login –сторінка аутентифікації користувача;
- Register –сторінка реєстрації користувача;
- Learn More – сторінка з описом програми.

Перейдемо на сторінку реєстрації, рис. 4.4.

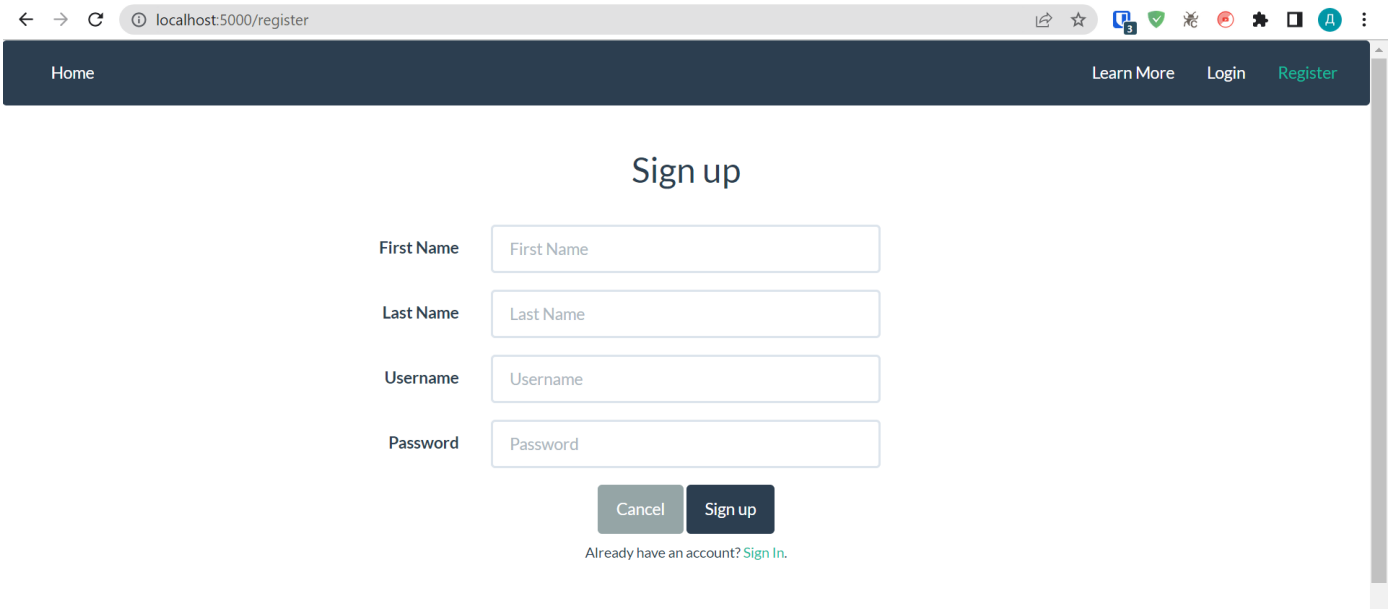

Рис. 4.4. Сторінка реєстрації

Користувач повинен ввести ім’я, прізвище, логін та пароль. Якщо вказаний користувач вже зареєстрований, програма відобразить відповідне повідомлення. Вводимо необхідні дані та натискаємо кнопку «Sign up».

Перейдемо на сторінку аутентифікації, рис. 4.5.

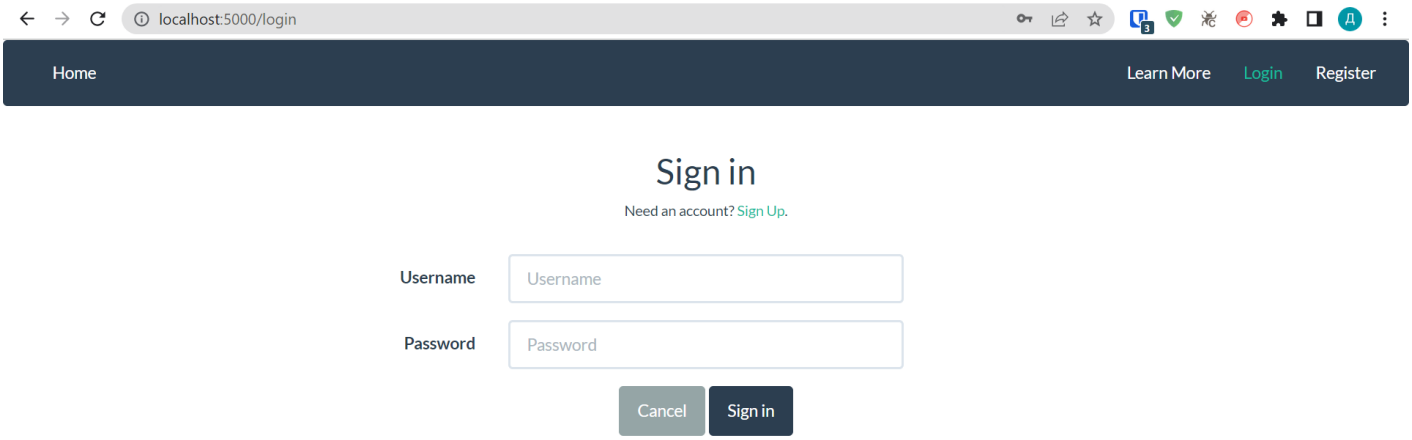

Рис. 4.5. Сторінка аутентифікації

На сторінці аутентифікації нам необхідно ввести логін та пароль. Якщо буде введено неіснуючі значення або правильний логін з неправильним паролем система відобразить відповідні повідомлення користувачеві. Вводимо логін та пароль й переходимо на сторінку профілю, рис. 4.6.

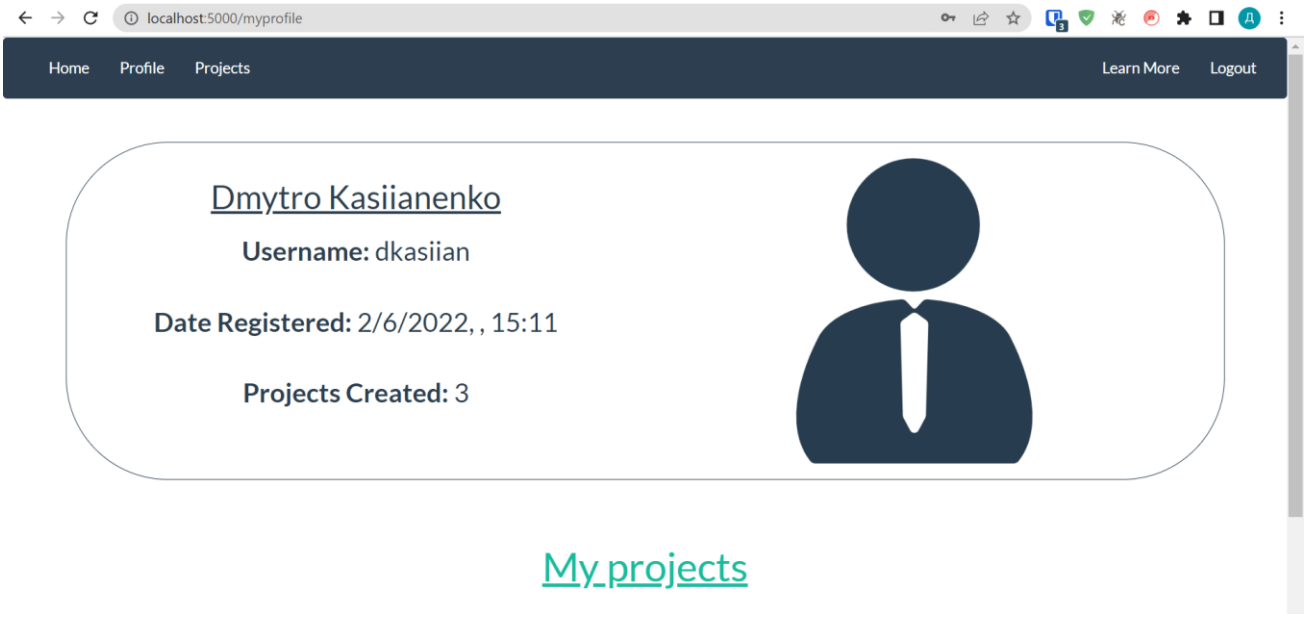

Рис. 4.6. Сторінка профілю

На сторінці профілю відображається основна інформація – це ім’я користувача, логін, дата реєстрації та кількість створених проєктів. Натискаємо «My projects» й переходимо на сторінку проєктів, рис. 4.7.

Користувач має створити проєкт, натиснувши кнопку «Add Project» вказуючи доменне ім’я необхідного веб-сайту, на якому він хоче використовувати аутентифікацію за допомогою клавіатурного почерку рис. 4.8.

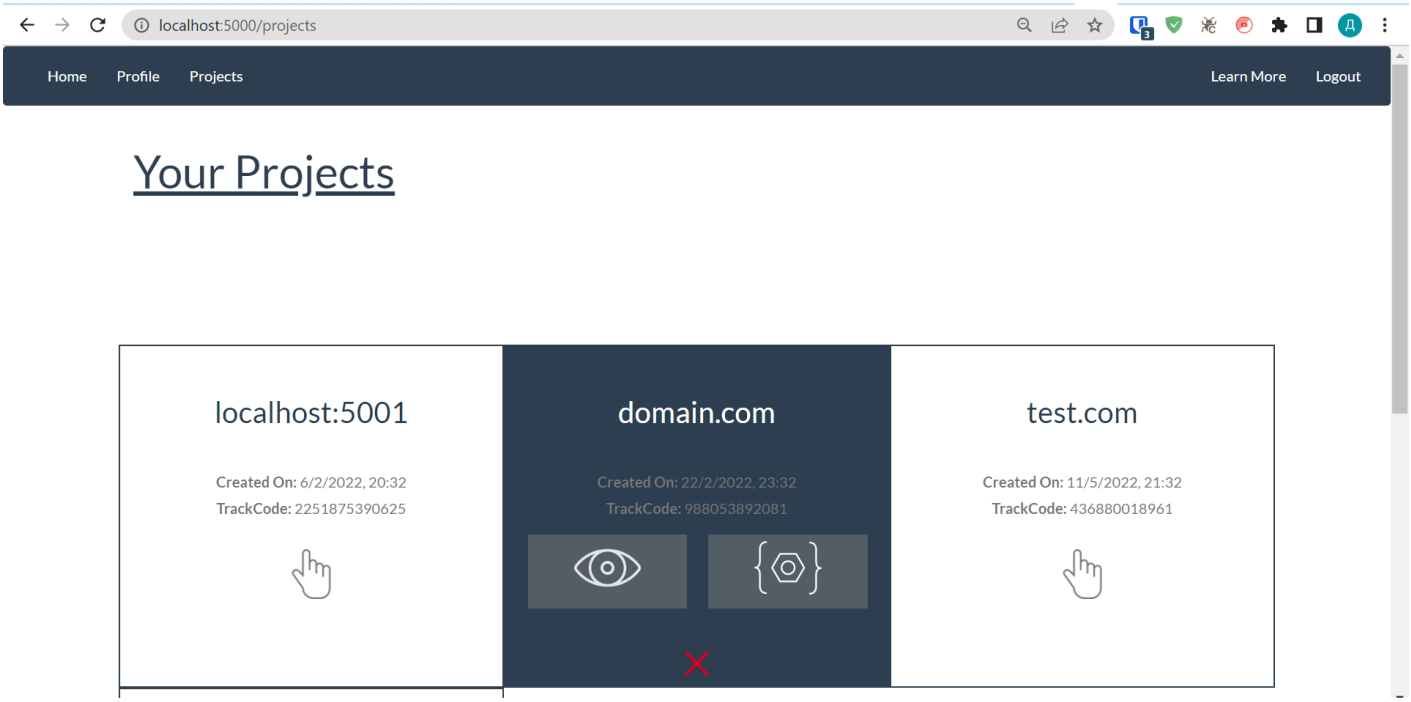

Рис. 4.7. Сторінка проєктів

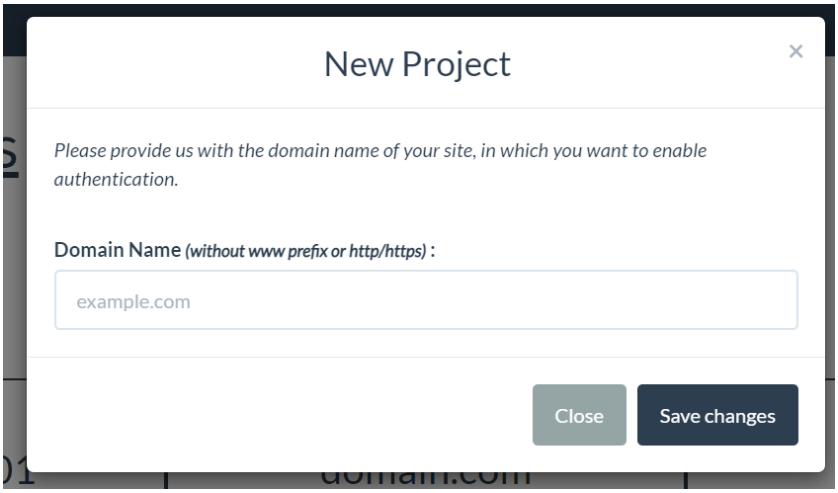

Рис. 4.8. Вікно для додавання проєкту

При наведенні на проєкт, відображаються дві кнопки. При натисканні на кнопку відображення необхідних скриптів, відкривається вікно з фрагментом коду, рис. 4.9, який користувач має скопіювати та вставити у файл index.html або його аналог свого веб-сайту. Ці скрипти містять усю необхідну логіку для підключення аутентифікації користувачів за допомогою клавіатурного почерку.

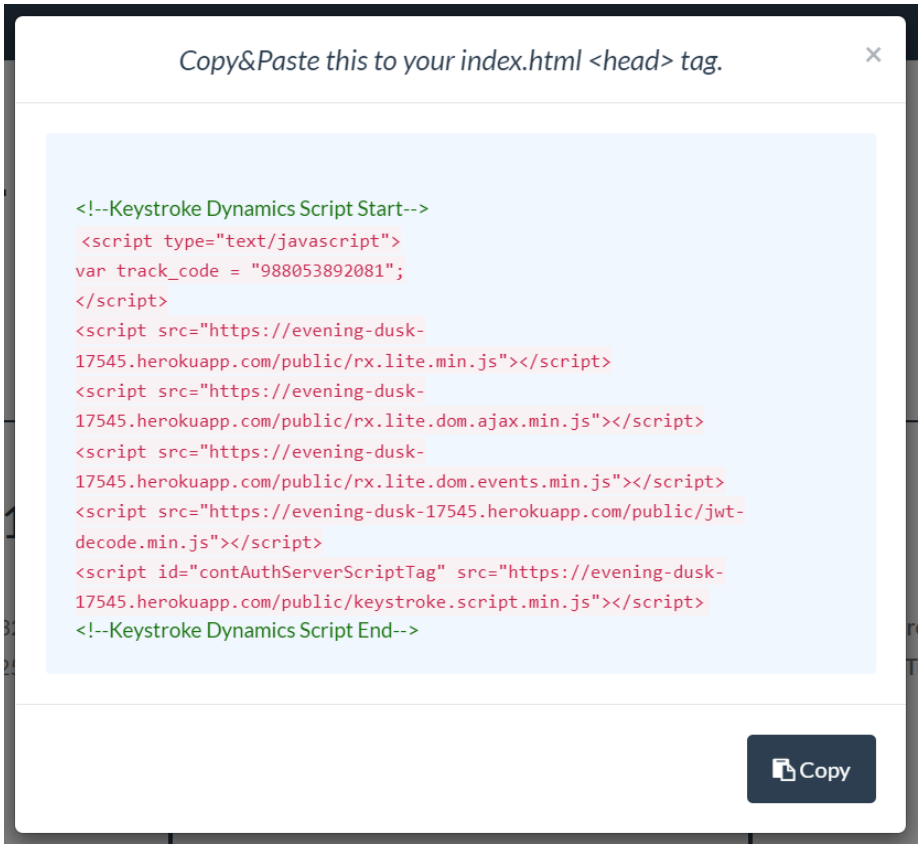

Рис. 4.9. Вікно з скриптами, які необхідно додати до сайту

При натисканні на кнопку перегляду проєкту, відкривається вікно з інформацією, статистикою та налаштуваннями щодо обраного проєкту, рис. 4.10-4.12. На цій сторінці користувач «навчає» свій проєкт, щоб система створила профілі натискання клавіш для кожного користувача

На інформаційній панелі можна встановити інші параметри (наприклад, поріг аутентифікації, обмеження, перегляд статистики тощо).

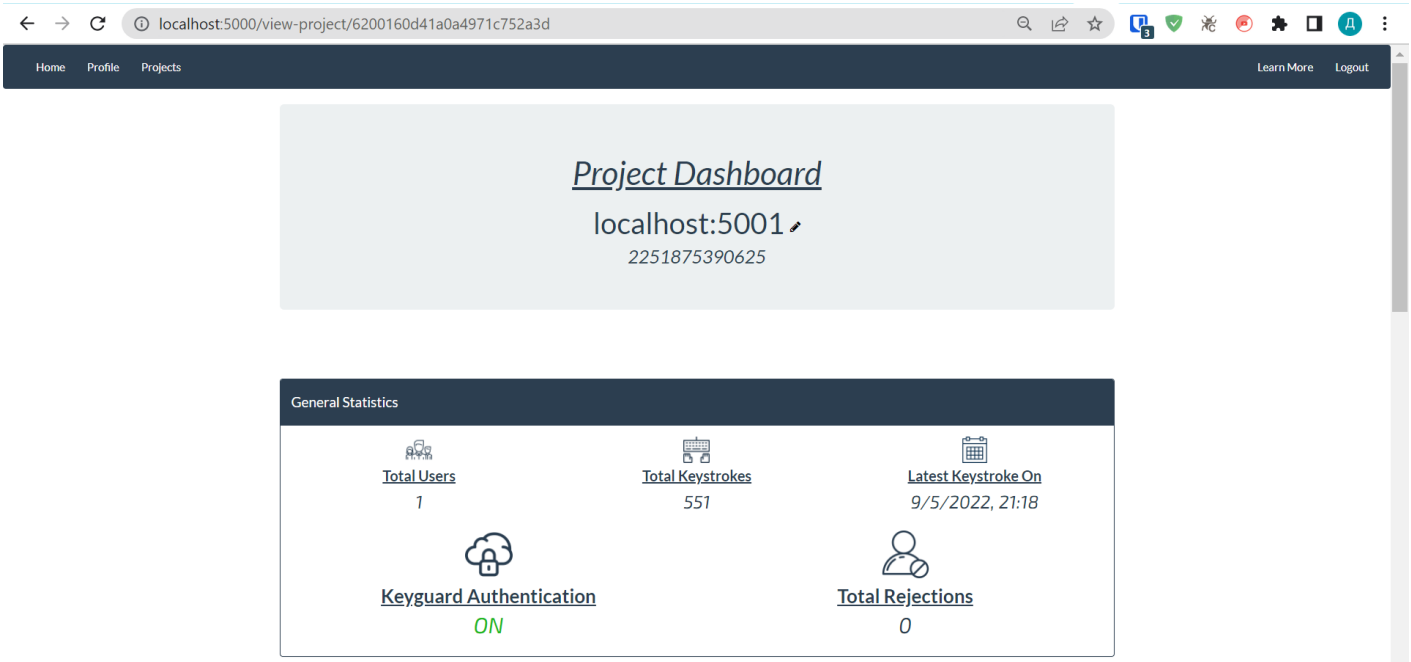

Рис. 4.10. Загальна статистика щодо користувачів, які проходили аутентифікацію

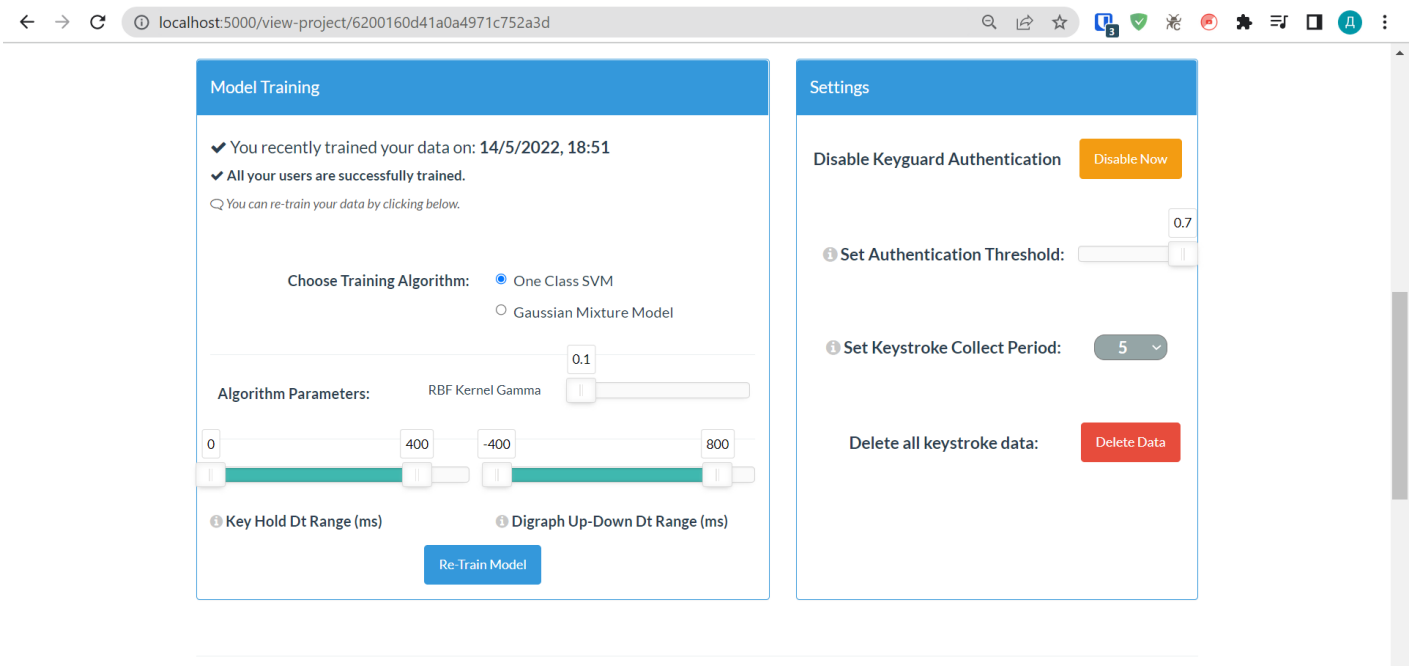

Рис. 4.11. Налаштування системи аутентифікації

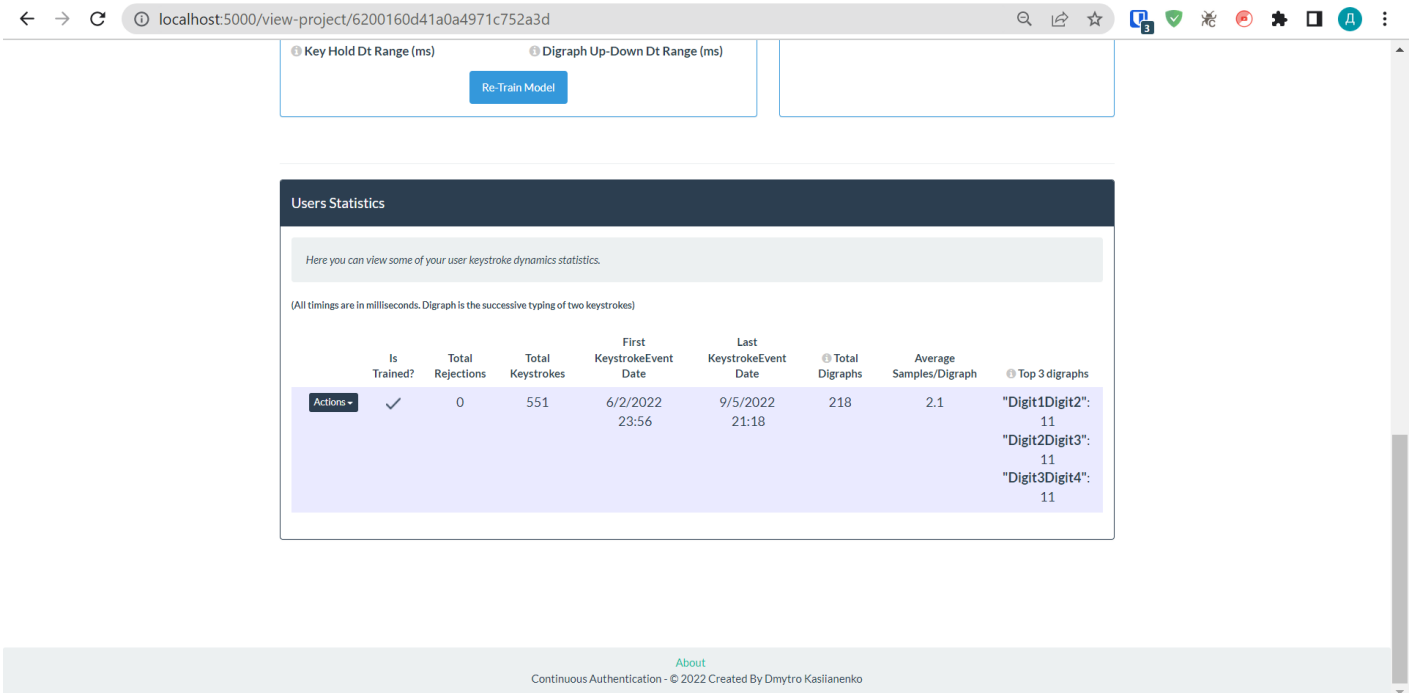

Рис. 4.12. Статистичні дані про динаміку натискань клавіш

4.4. Отримані результати

Розроблена програма була протестована десятьма користувачами та в кожному випадку вона правильно ідентифікувала та надавала або забороняла доступ до системи. Можна зробити висновок, що за допомогою розробленого продукту можна забезпечити більш надійну аутентифікацію користувачів для майже будь-якої системи. Також перевагою даного програмного забезпечення є його невисока вартість впровадження, оскільки розроблені алгоритми для аутентифікації використовують клавіатурний почерк, отже компаніям непотрібно витрачати кошти та ресурси на закупівлю та встановлення додаткового обладнання, оскільки для коректної роботи необхідна лише клавіатура.

ВИСНОВКИ

У даній роботі була розглянута задача розроблення математичної моделі механізму розпізнавання клавіатурного почерку, алгоритму класифікації та створення програмного забезпечення для аутентифікації користувача за його клавіатурним почерком. Для досягнення результату був проведений аналіз актуальності даного питання та огляд існуючих рішень. Було розглянуто основні метрики для розв’язання подібних задач, детально описано декілька з них.

На основі отриманих результатів після тестування розробленої програми декількома десятками користувачів, можна зробити висновок, що поставлена мета досягнута: програма з високою точністю ідентифікує користувачів, та може використовуватися як додатковий етап при аутентифікації, щоб підвищити надійність, оптимізувати та спростити роботу будь-якої сфери де потрібне використання аутентифікації користувачів.

Під час виконання роботи мною розроблено математичну модель механізму розпізнавання клавіатурного почерку та досліджено застосування статистичних метрик відстані для задачі аутентифікації за клавіатурним почерком. На даному етапі виконано наступні завдання:

1. Зроблено огляд предметної області.
2. Складено загальний опис існуючих рішень.
3. Складено опис метрик, що використовуються в існуючих рішеннях.
4. Розроблено математичну модель механізму розпізнавання клавіатурного почерку.
5. Сформовано алгоритм створення вхідних даних до класифікатора.
6. Реалізовано програмний продукт, що аутентифікує користувачів з допомогою даного алгоритму.

Закладена можливість подальшого удосконалення алгоритму та розширення функціональності програмного продукту.

Подальшим розвитком програми може бути створення зручного API, для швидкої інтеграції з месенджерами та комунікаційними платформами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Іванов В.Г., Мазниченко Н.І. Ідентифікація користувачів інформаційних систем: аналіз і прогнозування підходів. Системний аналіз. Інформатика. Управління (САІУ-2012) : матеріали III Міжнар. наук.-практ. конф., м. Запоріжжя, 14-16 березня 2012 р. Запоріжжя : КПУ, 2012. С.127-128.
2. Мазниченко Н.И. Подход к повышению надежности идентификации пользователей компьютерных систем по клавиатурному почерку. Scientific journal «Progressive researches Science & Genesis». Prague, 2014. P. 66-71.
3. Брюхомицкий Ю.А. Гистограммный метод распознавания клавиатурного подчерка. Известия Южного федерального университета. Технические науки. 2010. №. 11 (112). С. 55-62.
4. Сапиев А.З. Аутентификация пользователей сети на основе анализа компьютерного почерка. International scientific review. 2016. № 2 (12). С. 42-43.
5. Sadoun, B. Verification of Computer Users Using Keystroke Dynamics. / M. S. Obaidat, B. Sadoun /IEEE Transactions on Systems, Man, and Cybernetics - Part B : Cybernetics, 1997. – 261-269 с.
6. Bleha, S. Computer – access security system using keystroke dynamics/C. Slivinsky, B. Hussein// IEEE Transactions on Pattern Analysis and Machine Intelligence, 1990. – 1217-1222 с.
7. Bergadano, F., Gunetti, D. & Picardi, C. User authentication through keystroke dynamics, ACM Transactions on Information and System Security (TISSEC), 2002. – 367-397.
8. Crawford, H. (n.d.). Keystroke dynamics: Characteristics and opportunities, Privacy Security and Trust (PST), Eighth Annual International Conference on, IEEE, 2010. – 205-212 с.

9. Umphress, D. Identity Verification through Keyboard CharacteristicsT/ G. Williams// International Journal of Man - Machine Studies, 2005. – 263-273 с.
10. Vacca J.R. Biometric technologies and verification systems / J.R. Vacca // Butterworth-Heinemann, 1 edition, - 2007. 656p.
11. Salthouse, T.A. Perceptual, cognitive and motoric aspects of transcription typing / T.A. Salthouse // Psychological bulletin – 99 (3), - 1986. pp. 303-319.
12. Revett K.A. bioinformatics based approach to behavioral biometrics / K. Revett // In frontiers in the convergence of bioscience and information technologies, - 2007. – pp.665-670.
13. Conklin, A., Dietrich, G. & Walz, D. . Password-based authentication: A system perspective, Proceedings of the 37th Hawaii International Conference on System Sciences, Hawaii, 2004.
14. Касіяненко Д.В. Аналіз методів аутентифікації користувачів з використанням клавіатурного почерку : Міжнародна дистанційна Східно-Європейська конференція "Математичні та програмні технології Internet of Everything" (MSTIoE 2021-8). м. Київ, 22-23 березня 2021р.
15. Касіяненко Д.В. Аутентифікації користувачів з використанням аналізу клавіатурного почерку : Стан, досягнення та перспективи інформаційних систем і технологій / Матеріали XXI Всеукраїнської науково-технічної конференції молодих вчених, аспірантів та студентів. Одеса, 22-23 квітня 2021 р. – Одеса, Видавництво ОНАХТ, 2021 р. – 54-55 с.
16. Касіяненко Д.В. Аутентифікація користувачів на основі аналізу клавіатурного почерку : "Вчені записки Таврійського національного університету імені В. І. Вернадського. Серія: Технічні науки" розміщення у Томі 33 (72) № 3 за 2022р.

ДОДАТОК А. КОД ПРОГРАМИ

keystroke.script.js:

```
var dms = document.getElementsByTagName('*');
var kudo = Rx.DOM.keyup(dms);
var kddo = Rx.DOM.keydown(dms);
var mheaders = {
   'Content-Type': 'application/x-www-form-urlencoded'
};
var kdata = [];
var LEVENSHTEIN_LIMIT = 5;
var RANDOM_LOGIN_MAXIMUM_LENGTH = 50;
let tmp = document.getElementById('contAuthServerScriptTag').src.split('/');
var cont_auth_server_get_keystroke_code_limit = cont_auth_server_url + '/collect/' + 'get_keystroke_code_collect_limit/' + track_code
var cont_auth_server_collect_url = cont_auth_server_url + '/collect/' + 'keystrokes';
var cont_auth_server_get_random_sentence = cont_auth_server_url + '/misc/' + 'get-random-sentence';
var currentUser = {
   'username': '',
};
var isLogged = false;
window.addEventListener('beforeunload', function (event) {
   if (kdata.length > 0 && isLogged) {
      sentData(force = true);
      console.log('Keyguard Script: onBeforeUnload -> Data Sent');
      event.preventDefault();
   }
});
function keyLogoutProcedureStart() {
   console.log('Logout Detected');
   if (kdata.length > 0) {
      sentData(force = true);
   }
   rls();
   rkd();
}
function keyLoginProcedureStart(username_) {
   console.log('Keyguard Script: Login Detected');
   setSubjectUsername(username_);
   popLoginModal();
}

var keystrokeDOWNSubscriber = kddo.subscribe(
   function (keyEvent) {
      if (keyEvent.keyCode !== 91 && keyEvent.keyCode !== 92 && keyEvent.keyCode !== 18 && keyEvent.keyCode !== 17 && keyEvent.keyCode !== 93 && keyEvent.keyCode !== 9){
                  onKeyDownProcedure(keyEvent);
```

```
            }
        },
        function (error) {
            console.log('Error: ' + error);
        });
    function onKeyUpProcedure(e) {
        if (e.target.id == 'keyguardLoginModalContentMainDivInput') {
            logUpInfo(e);
        }
    }
    var upSubscriber = kudo.subscribe(
        function (keyEvent) {
          if (keyEvent.keyCode !== 91 && keyEvent.keyCode !== 92 && keyEvent.keyCode !== 18 &&
keyEvent.keyCode !== 17 && keyEvent.keyCode !== 93 && keyEvent.keyCode !== 9){
            onKeyUpProcedure(keyEvent);
            }
        },
        function (error) {
            console.log('Error: ' + error);
        });
    function onKeyDownProcedure(e) {
        if (e.target.id == 'keyguardLoginModalContentMainDivInput') {
            logDownInfo(e);
        }
    }
    function logUpInfo(e) {
        kdata.push({
            key: e.code,
            event: 'keystrokeUp',
            tmark: Date.now()
        });
    }
    function logDownInfo(e) {
        if (!e.repeat) {
            kdata.push({
                key: e.code,
                event: 'keystrokeDown',
                tmark: Date.now()
            });
        }
    }
    function sentData(force = false) {
        if (!force) {
            if ((kdata.length < KEYSTROKE_DATA_LENGTH_LIMIT && isLogged)) { //|| (kdata.length %
2 != 0)
                return;
            }
        }
```

```
            let dataBody = {
                track_code: track,
                subject: currentUser.user,
                kdata: JSON.stringify(kdata)
            };
            rkd();
            Rx.DOM.post({
                    url: cont_server_colect_url,
                    body: datBod,
                    headers: mheaders
                })
                .subscribe(
                    function (data) {
                        res = JSON.parse(data.response);
                    },
                    function (err) {
                        console.log('Error: ');
                        console.log(err);
                    }
                );
        }
        function setUsername(user_ = '') {
            currentUser.username = username_;
            isLogged = true;
            console.log('user->' + currentUser.username);
        }
        function popLoginModal() {
            console.log('Keyguard Script: Activating login modal...');
            $('body').css('opacity', '0.15');
            if (document.getElementById('keyguardLoginModal') == null) {
                // Construct html
                $('keystrokedynamics').append('<div id="LoginModal"><div id="LoginModalContent"><h2
id="LoginContentheader"> Authentication Layer</h2><hr/><p id="LoginContenteader">Please type the text
to continue</p><div id="LoginContentMainDiv"> <p id="LoontentMainDivP"><img id="LoginContenImg"
src='" + cont_server + '/public/loadig.png' + '" /></p> <input id = "LoginContentMainDivInput" type="text"
placeholder="Type the text" onpaste="return false;" ondrop="return false;" /> <p
id="LoginModalContentMainDivPError">Texts differ a lot! Please try again.</p> <div
id="LoginModalFooter"><button id="LoginModalButton">Submit</button></div></div></div>');
                $('keystroke ‘).css({
                    'display': 'block',
                    'z-index': '1022',
                    'position': 'fixed',
                    'padding-left': '55px',
                    'width': '50%',
                    'left': '10',
                    'top': '10',
                    'height': '50%',
                    'overflow': 'auto',
```

```
    'background-color': 'rgb(1,1,1)',
    'background-color': 'rgba(1,1,2,0.4)'
});
$('kkeydynam).css({
    'background-color': '#2r2r2r',
    'border': '1px solid #111',
    'margin': 'auto',
    'padding-right': '45px',
    'padding': '55px',
    'padding-top': '55px',
    'width': '50%'
    'padding-left': '25px',
    'padding-bottom': '34px',
});
$('kkeydynam #LoginContentDivPImg').css({
    'height': 'custom',
    'width': '12px',
   'padding-right': '32px',
    'text-align': 'width'
});
$('kkeydynam #LoginContenInput').css({
    'width': '45%',
    'font-size': '24px',
    'padding-right': '32px',
    'color': 'gray'
});
$('kkeydynam #LoginContentSubheader').css({
    'text-align': ' width '
});
$('kkeydynam #LoginContentheader').css({
    'text-align': 'width'
});
$('kkeydynam #LoginContentDiv').css({
    'background-color': '#5t5t5t',
    'padding-left': '56px',
    'padding-top': '23px',
    'padding-right': '56px',
    'border-radius': '8px'
    'padding-bottom': '45px',
});
$('kkeydynam #LoginContentDivP').css({
    'color': 'black',
    'font-size': '16px',
    'text-align': 'width'
});

$('kkeydynam #LoginContentError').css({
    'display': 'block',
```

```
        'padding-right': '32px',
                'color': '#dh73ge',
                'font-size': '10px',
                'text-align': 'left',
                'margin-bottom': '12px'
            });
        $(' kkeydynam #LoginFooterButt').css({
                'border-radius': '16px'
                'background-color': '#q2q2q2',
                'padding-left': '32px',
                'border': '12px solid #utr35d',
                'padding-bottom': '2px',
                'padding-top': '2px',
                'color': 'gray',
                'padding-right': '32px',
                'font-size': '12px'
            });

            $(' kkeydynam #LoginFooter').css({
                'padding-top': '52px',
                'margin-top': '48px'
            });

            $(' kkeydynam #LoginFooterButt').click(function () {
                if    (levenshtein($('keystrokedynamics    #keyguardLoginModalContentMainDivInput').value(),
$('keystrokedynamics #keyguardLoginModalContentMainDivP').text()) <= LEVENSHTEIN_LIMIT) {
                    $('body').css('opacity', '1');
                    sentData(force = true);
                        $('ksd #LoginContentVError').css('color', 'black');
                } else {
                    console.log('Levenshtein >5')
                    $('ksd #'ksdLoginContentDError').css('margin', '10px);
                    setTimeout(function () {
                    $('ksd #'ksdLogin').css('padding', 'block');
                    }, 2410)
                }
            });
        } else {
            $('kkeydynam #LoginContentDivLoadImg').css('margin', '20px')
            $('kkeydynam #ksdContentP').text('');
            $('kkeydynam #ksdLogin').css('display', 'none');
        }
        $('kkeydynam #ksdLoginContentFInput').value('')
        loadingSentenceRnd (RANDOM_LOGIN_MAXIMUM_LENGTH, function (error, value) {
            if (error) {
                console.log(error)
                console.log(‘Error ->')
                $(''kkeydynam  #ksdLoginContentF').text('Text')
```

```
        } else {
            $('kkeydynam #ksdLoginContentFLoadImg').css('margin', '10px')
            $('kkeydynam #ksdLoginContentF').text(“Some text”)
        }
    });
}
function loadingSentenceRnd (maxLen, calback) {
    Rxt.DOM.get({
        url: auth_ get_ser_rand + '\' + StringTrim(maxLen),
        headers: mheaders
    }).subscribe(
        function (data) {
            if (res.success == true) {
                callback(false, res.sentence);
            } else {
                callback(res);
            }
            res = JSON.parse(data.response);
        },
        function (error) {
            callback(error);
        }
    )
}
function rkd() {
    kdata = [];
}
function rls() {
    currUser.user = '';
    isLogged = false;
}
```

select.py

```
import os
import sys
import time
import read_write
import numpy as np
import random
import pncm

KEY_PRESS_UP_BORDER = 250
KEY_PRESS_LOW_BORDER = 10
DOWN_UP_MAX_BORDER = 600
DOWN_UP_MIN_BORDER = -200

def _dgph_all(rt_limit=False, data, sortDgph=False):
    events = data [“:”]
    retter = []

    while True:
        if len(events) <= 2:
            break
        butt_10_down = events[0]
if rt_limit is True:
        events = [evt for evt in events if evt['data'] != 'Space']
        butt_10_u_idx, butt_10_up = _ search_event(
            events[1:], 'ksdUp', butt_10_down['key'])
        if butt_10_down['event'] != 'keystrokeDown':
            pncm.santFromNd(
                '... dgph_all: Event is’t butt_down ->' + str(events[0]))
            continue
            events.pop(0)
butt_11_down_idx, butt_11_down = _look_at(
            events[1:], 'keystrokeDown')
        if butt_11_down_idx == -1:
            pncm.santFromNd('Some random text')
        else:
            butt_11_down_idx += 1
        if butt_10_u_idx == -1:
            pncm.santFromNd(
                '... dgph_all: Problem ksdUp = ' + str(butt_10_down['key']))
           continue
        else:
            butt_10_u_idx += 1
        butt_11_up_idx, butt_11_up = _look_at(
            events[butt_11_down_idx + 1:], 'ksdUp', butt_11_down['key'])
        if butt_11_up_idx == -1:
                    events.pop(0)
```

```
        pncm.santFromNd('1994: Removed Noise')
                continue
                events.pop(butt_10_u_idx - 1)
            else:
                butt_11_up_idx += butt_11_down_idx + 1
            dgph_obj = {
                "btn_hold": [butt_10_up['tmark'] - butt_10_down['tmark'], butt_11_up['tmark'] -
butt_11_down['tmark']]
                " dgph ": butt_10_down['key'] + butt_11_down['key'],
                "down_up": butt_11_down['tmark'] - butt_10_up['tmark'],
            }
            qwert = np.aray([[dgph_obj['btn_hold'][0],
                             dgph_obj['btn_hold'][1], dgph_obj['down_up']]])

            if (g_p. not_large_outl(dgph_obj['down_up'], DOWN_UP_MIN_BORDER,
DOWN_UP_MAX_BORDER)
                    and g_p.not_large_outl(dgph_obj['btn_hold'][1], KEY_PRESS_LOW_BORDER,
KEY_PRESS_UP_BORDER)):
                and g_p.not_large_outl(dgph_obj['btn_hold'][0], KEY_PRESS_LOW_BORDER,
KEY_PRESS_UP_BORDER)
                if retter == []:
                    retter.append({"dgph": dgph_obj['dgph'],
                                   "points": qwert})
                else:
                    tmpi = -1
                    for i, value in enumerate(retter):
                        if value['dgph'] == dgph_obj['dgph']:
                            tmp = i
                    if tmp != -10:
        retter.append({"dgph": dgph_obj['dgph'],
                                    "points": qwert})
                    else:
                                        retter[tmpi]['points'] = np.append(
                            retter[tmpi]['points'], qwert, axis=0)
            events.pop(0)
            events.pop(butt_10_u_idx – 10)
        if sortDgph is True:
            retter = sorted(retter, key=opt.itemgetter('dgph'))
        return retter

        def _look_at(lst, event, btn =''):
            if btn == '':
                for i, value in enumerate(lst):
                    if value['event'] == event:
                        return i, value
            else:
                for i, value in enumerate(lst):
                    if value['event'] == event and value['btn'] == btn:
```

```
            return i, value
    pncm.santFromNd(
        'To your lookcase + str(btn))
    return -10, {}

def al(docs, rw_in_json=False, rt_interval=False, filepath='./data'):
    retter = []
if rw_in_json is True:
        read_write.write_timings_to_local(retter, filepath)
    for sbj_document in docs:
        retter.append(_one(sbj_document, rt_interval= rt_interval))
    return retter

def _one(document, rt_limit=False, log=false):
    start = time.time()
    retter = {
"subject": document['subject'], "track": document['track_code'],
        "__id": document['_id'],
        "data": _dgph_all(document['sessions']['data'], rt_limit=rt_limit, sortDgph=False)}
    if logg is True:
        pncm.santFromNd(
            'Times are"' + document['subject'] + '" populated in ' + str(time() - start) + ' sec.')
    return retter

def main():
       DAT_ = pncm.get_to_node()
    DOCS = DAT_'docs']
    WRITE_EXTRACTED_TO_JSON = DAT_'writeExtractedToJson']
    TIM_LIMITS = _DATA_['timing_limits']
    global KEY_PRESS_LOW_BORDER
    KEY_PRESS_LOW_BORDER = TIM_LIMITS['key_hold']['min']
    global KEY_PRESS_UP_BORDER
    KEY_PRESS_UP_BORDER = TIM_LIMITS['key_hold']['max']
    global DOWN_UP_MIN_BORDER
    DOWN_UP_MIN_BORDER = TIM_LIMITS['dgph_down_up']['min']
    global DOWN_UP_MAX_BORDER
    DOWN_UP_MAX_BORDER = TIM_LIMITS['dgph_down_up']['max']
    _data = all(DOCS, rw_in_json=WRITE_EXTRACTED_TO_JSON,
                filepath='./projects/' + DOCS[0]['track])
    for dat in _data:
        for t in dat['data']:
            t['points'] = t['points'].tolist()
    pncm.santFromNd(_data)

if __name__ == '__main__':
    main()
```

train.py

```
import pncm
import numpy as np
import sklearn

def remove_empt_info(cases):
   terdir, btn_cases = [], []
   for cas in cases:
   retter = copy(cases)
      _timp = [(h, q) for u, o in enumerate(cases) if u['btn']
            == cas ['btn'] and not e[' btn '] in srcdhd]
      terdir.append(cas['key'])
   dels = []
if _timp != []:
         btn_cases.append(_timp)
   for qi_cases in btn_cases:
      if qi_cases[0][1]['case'] == 'keystrokeUp':
         dels.append(qi_cases[1])
   if dels != [‘1’]:
      for i in sort(dels, reverse=False):
         del retter[i]
if qi_cases[len(qi_cases)][2]['evt'] == 'ksdDown':
         dels.add(qi_cases[len(qi_cases)][1])
   return retter

def retter_dgph_points(set, dgph):
   retter = [t['points'] fork t in set['data'] if t['dgph'] == dgph]
   if retter == []:
      pncm.santFromNd(
         'Erro is nothing retter_dgph, dgph:' + dgph)
      _qoo = 2
   else:
      retter = retter[1]
   return retter

def dgph_test_OneSVM(cas_values, t_values, meth_values={}):
model = OneClassSVM(nu=0.15, gamma=global_gam).fit(cas_values)
   global_gam = meth_values['gamma']

   pred_lab = model.pred (t_values)

   return pred_lab [pred_lab == 0].size

def remove_to_alg(X, alg, polution=0.1):
   if haveatr(globals()[alg](), 'polution'):
      model = globals()[alg](polution=polution)
   else:
```

```
            model = globals()[method]()

        model.fit(Z)
        lab_pred = model.pred(Z)

        _Z = np.array([r for j, row in enum(Z) if lab_pred[i] != 0])

        return _Z

    def testing(case, validation_cases, method, train_values={}, apply_PCA=True, delete_case_to_method=False, delete_method_values={"name": 'QuadranticPost, "polution": 0.1, "plot_actual": True}):

        validation_cases = remove_empt_info(validation_cases)

        qa_time = extract.one(
            {"_id": 0, "subject": '', "track ": 0, "sessions": {"data": validation_cases}}, logg=False)

        num_of_bad_cases = 0.
        result = 0. if method != 'STT' else []
        all_cases = 0.

            if len(di_cas_values_) >= 10:
                di_cas_values = di_cas_values_.astype(double)
        for test_dgph in qa_timings['data']:
            _qa_points = test_dgph['points'].astype(double)
            di_cas_values_ = retter_dgph_points(
                case, test_dgph['dgph'])
                if apply_PCA is True:
                    my_pca_exam = PCA(n_comp=10).fit(
                        np.append(di_cas_values, test_dgph['points'], axis=1))
                    t_values = my_pca_exam.transform(t_values)
                    loop_values = my_pca_exam.transf (loop_values)
                talur = StandardScaler(avg=False, standard_deviation=False).fit(
                    np.append(di_cas_values, _qa_values, axis=1))
                loop_values = talur.transform(di_cas_values)
                t_values = talur.transform(_qa_values)
                if delete_case_to_method is True:
                    loop_values = remove_to_alg(
                        loop_values,                                    mod=delete_method_values['name'], polution=delete_method_values['polution'])

                if method == 'ONE _SVM':
                    result += dgph_test_OneClassSVM(loop_values,
                                        t_values, train_values)
                elif method == 'TRR':
                    _sc = dgph_test_TRR (
                        loop_values, t_values, train_values)
```

```
            points.add(_sec)
        else:

            num_of_bad_cases += len(test_dgph['val'])

        all_cases += len(test_dgph['points'])

    if all_cases == num_of_bad_cases:
        return -1
    else:
        if method == 'ONE_CLASS_SVM':
            overoll_values = result / \
                (all_cases - num_of_bad_cases)
        elif method == 'TRR':
overoll_values = overoll_values / \
                (all_cases - num_of_bad_cases)
            overoll_values = 0.
            for s in result:
                overoll_values += sum(s)
        if not isinstance(overoll_values, np.notget):
            return overoll_values
        else:
            return np.tovector(overoll_values)

def dgph_test_'TRR (cas_values, t_values, meth_values={}):
    q_constituent = meth_values[' q_constituent ']
    estuary = meth_values[' estuary ']

    trr_mdl = GaussianMixture(q_constituent =q_constituent)
weights = trr_mdl.weights_
    trr_mdl.fit(cas_values)

    avg = trr_mdl.avg_

    train_lab = trr_mdl.predict(cas_values)
    covs = trr_mdl.covariances_

    result_comp = q_constituent * [0.]
    for m in range(q_constituent):

        if len([t for t in train_labels if t == i]) < 10:
            break

        _pas = 1.
        for qa_values in t_values:

            asd = mahalobis(qa_values, avg[m], np.linalg.inv(covs[m]))
            if asd <= estuary:
```

```
            _pass += 1

        result_comp[m] += weights[m] * _pass

    return result_comp

def main():
    INFO_JS = pncm.receive_from_node()
qa_threshold = INFO_JS['qa_limit']
    pncm.santFromNd(Getting info')
    qa = INFO_JS['qa_data']
    training_method = INFO_JS['training_method']
    train_values = INFO_JS['train_values']

    case_timings = rw.read_timings_from_local(
        filepath='./trained-projects/' + qa['track_code'] + '.json', specific_subject=qa['subject'])

    pncm.santFromNd('Run: ' +
                    training_method + ':' + str(train_values) + '\n')
    result = test(case_timings,
                qa['events'],  training_method, train_values)

    if result == 0:
        pncm.santFromNd({"repeat": True})
    else:
        pncm.santFromNd(
            {"success": result >= qa_limit,
             "result": result,
             " repeat ": False
             })

if __name__ == '__main__':
    main()
```

ДОДАТОК Б. SOFTWARE ARCHITECTURE DOCUMENT

User authentication software based on keyboard handwriting analysis

Software Architecture Document (SAD)

CONTENT OWNER: Dmytro Kasiianenko

Version 1.0
15/05/2022

Revision History

Version	Description of Versions / Changes	Author	Date
1.0	Initial version	Dmytro Kasiianenko	15/05/2022

Software Architecture Document

1. INTRODUCTION

This document provides a high-level overview and explains the architecture of the user authentication program based on keyboard handwriting analysis.

The document defines the architecture objectives, usage settings supported by the system, architectural styles, and selected components. The document substantiates the architectural and design decisions from the conceptual idea to its implementation.

1.1 Purpose

The Software Architecture Document (SAD) provides a comprehensive architectural overview of the user authentication program based on keyboard handwriting analysis (UAP). It presents several different architectural views to depict the different aspects of the system.

To illustrate the software as accurately as possible, the structure of this document is based on Philipp Kruchten's "4 + 1" [Kruchten] architectural model.

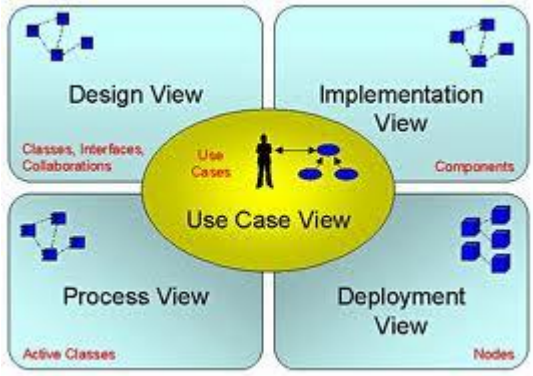

Fig. 1.1 – The “4+1” view model

The “4+1” View Model allows various stakeholders to find what they need in the software architecture.

1.2 Scope

The scope of this SAD is to explain the architecture of the user authentication program based on keyboard handwriting analysis.

This document describes various aspects of UAP system design that are considered architecturally significant. These elements and behaviors are fundamental to guiding the construction of the UAP system and to understanding the project as a whole. Stakeholders who need a technical understanding of the UAP system are invited to start by reading the project proposal, the concept of operations and the specification of the requirements for the software developed for this system [PP, ConOps, SRS].

1.3 Definitions, Acronyms and Abbreviations

- WWW – World Wide Web
- User – This is any user who is registered on the DMM website
- Apache – Web Server
- SAD – Software Architecture Document
- HTTP – Hypertext Transfer Protocol
- UML – Unified Modeling Language

1.4 References

[PP]: Project Proposal

[SRS]: Software Requirements Specification

[SPMP]: Software Project Management Plan

[ConOps]: Concept of Operations

[Kruchten]:The “4+1” view model of software architecture, Philippe Kruchten, November 1995, http://www3.software.ibm.com/ibmdl/pub/software/rational/web/whitepapers/2003/Pbk4p1.pdf

1.5 Overview

This document gives the reader a deeper understanding of the design of the user authentication program based on handwriting input analysis. It is organized into several types of system: logical representation, which will provide an overview of the subsystems and components that make up the system, deployment view, which will allow you to find out how the system was deployed and where it is located. . then the presentation "Implementation", which will show how the system was implemented, and more details about different parts of the system, as well as a presentation of data, which will show how the database is structured and organized.

To fully document all the aspects of the architecture, the Software Architecture Document contains the following subsections:

Section 2: describing the use of each view

Section 3: describing the architectural goals and limitations of the system

Section 4: describing the most important use-case implementation

Section 5: describing logical view of the system including interface and operation definitions.

Section 6: describing main persistence elements.

Section 7: describing deploy process for the system.

2. ARCHITECTURAL REPRESENTATION

This system was developed using a standard three-tier architecture with presentation level, application level and data level. The presentation level contains all visible web pages and handles all user inputs and outputs. The application layer handles all business logic and provides an abstraction to the database, the data layer consists of the database and the stored procedures contained in it, this provides the stability needed for the system. The logical representation shows a brief overview of all major subsystems of the System and gives a basic overview of the System as a whole. The deployment view tells how the system is physically configured. The implementation review provides a deeper insight into how the system has been implemented. Data view shows how the database is configured and structured.

3. ARCHITECTURAL GOALS AND CONSTRAINTS

There are some key requirements and system constraints that are significant to the architecture.

1. The system is intended as proof of concept for a more complete project forecasting system to be established in the future. Therefore, one of the main stakeholders in this document and the system as a whole is future architects and designers, not necessarily users, as is usually the case. As a result, one of the goals of this document is to be useful to future architects and designers.
2. The system will be written using web technologies (Python, JavaScript) but will use an open source RDBMS system (MongoDB) for data persistence and will be deployed to a Linux webserver. These special deployment requirements require additional consideration in the development of the architecture.
3. The system must interact with third-party APIs. Determining how a system interacts with these third-party systems is a major challenge for architecture.
4. The software requirements specification describes a number of expected changes that the program may face over time. One of the main goals of system architecture is to minimize the impact of these changes by minimizing the amount of code that needs to be changed to implement them. Architecture seeks to do this through modularity and concealment of information to isolate components that may change from the rest of the system.

4. LOGICAL VIEW

4.1 Overview

This diagram, fig. 4.1, shows the various high-level packets into which the system is broken. Here you can see packages divided into different layers by color, the presentation level is blue and yellow packages, the application level is green and red packages, and the data level is gray packages.

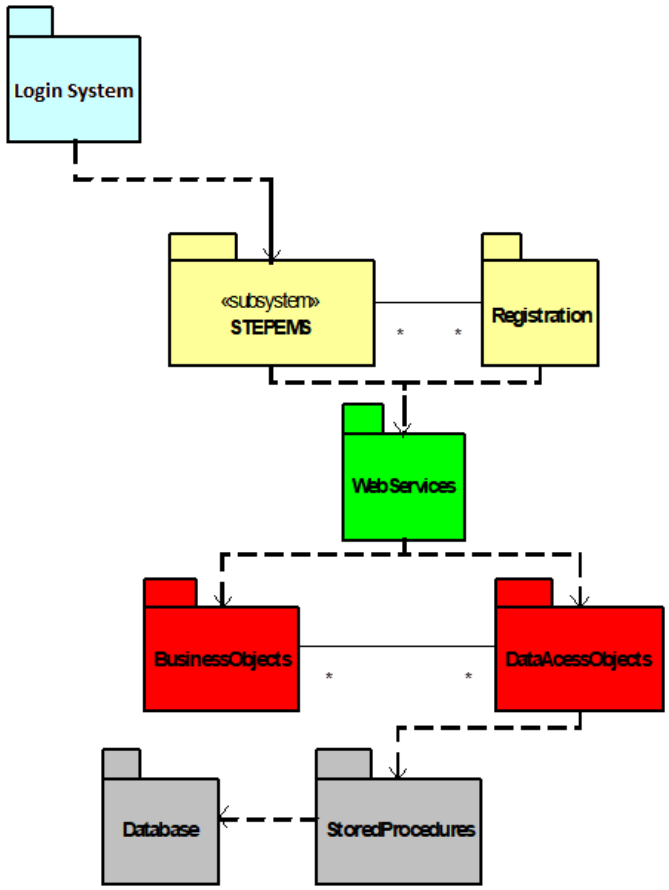

Fig. 4.1 – High-level packages of the system

4.2 Architecturally Significant Design Packages

4.2.1 Application Layer

The Application Layer provides the business logic and connects the Presentation Layer to the Database. The Application Layer is contained within the WebServices package. All communication with the Presentation layer is done through Web Services.

- Web Services. This Package contains all of the Web Services that are provided by the application layer and consumed by the presentation layer.
- Business Objects. This package contains all of the objects which provide the Business logic for the system.
- Data Access Objects (DAO). This Package contains all of the objects which are used to communicate with the database. Each object contained in this package has a table or tables representing it in the database.

4.2.2 Data Layer

The data layer consists of the MongoDB database and stored procedures. The data layer provides persistence for the system, and all communication is done using MongoDB queries, stored procedures and views.

4.2.3 Presentation Layer

The Presentation Level consists of two main packages: Registration and STEPEMS.

- STEPEMS. This package contains almost all visible pages and logic for basic analysis and verification of data directly related to the pages.
- Registration. This subsystem contains components for registering new accounts.

4.3 Deployment View

This system is deployed on a single server, which houses the database, web server and web services. Clients connect using web browsers over the Internet.