Київський національний торговельно-економічний університет
Кафедра інженерії програмного забезпечення та кібербезпеки

ВИПУСКНИЙ КВАЛІФІКАЦІЙНИЙ ПРОЕКТ

на тему:

«Розробка біометричної системи аутентифікації обліку відвідування студентів»

Студента 2м курсу, 6 групи,
спеціальності 121
«Інженеріяпрограмного
забезпечення»
спеціалізації
«Інженеріяпрограмного
забезпечення»
Моспан Олександр
Вікторович

_____
підпис студента

Науковий керівник
доктор технічних наук,
професор кафедри інженерії
програмного забезпечення та
кібербезпеки
Пашорін Валерій
Іванович

_____
підпис керівника

Гарант освітньої програми
доктор технічних наук,
професор кафедри інженерії
програмного забезпечення та
кібербезпеки
Криворучко Олена
Володимирівна

_____
підпис гаранта

КИЇВ – 2020

Технічне завдання на розробку ПЗ

1. **Введення:**

1.1. Призначення.

ПЗ призначене для ведення електронного журналу відвідувань студентів по біометричним даним.

1.2. Мета створення.

Метою створення програмного продукту є спрощення процесів ведення обліку студентів.

1.3. Область дії.

Область дії програмного продукту – університет.

1.4. Короткий огляд.

Система біометричної аутентифікації.

2. **Загальний опис:**

2.1. Повне найменування інформаційної системи.

«Біометрична система аутентифікації обліку відвідування студентів».

2.2. Скорочене найменування інформаційної системи.

«Біометрична система аутентифікації студентів»,

2.3. Функції продукт (короткий опис).

При розробці системи біометричної аутентифікації потрібно реалізувати наступні функції:

а) Перегляд, додавання та редагування карток студентів.

б) Зберігання основної інформації про студента.

в) Додавання та зберігання біометричних даних студента.

2.4. Головний бенефіціар та потенційні користувачі системи.

Головним бенефіціаром виступає університет. Потенційними користувачами системи є викладачі та студенти навчального закладу.

2.5. Характеристики користувача.

Потенційний користувач повинен розуміти систему роботи. Мати рівень користувача ПК від середнього і вище.

3. **Вимоги:**

3.1. Вимоги до інтерфейсу.

Інтерфейс програмного продукту повинен бути зручним та інтуїтивно зрозумілим.

Вимоги до розміщення об’єктів на екрані:

а) Об’єкти, що по своїй ролі можна віднести до основних, потрібно групувати

в центрі екрану;

б) Другорядні об’єкти – по периферії;

в) Кнопки групувати відповідно до їх функціонального призначення, не розташовувати кнопки (що відносяться до різних груп) поруч один з одним.

Вимоги до виведення інформації на дисплей:

Будь-яка взаємодія користувача з програмним забезпеченням повинна відбуватися в окремому вікні. Тобто, якщо користувачеві необхідно здійснити дію, що не зв’язана напряму з його поточною роботою – повинно відкриватися нове вікно.

Вимоги до мови взаємодії\спілкування з користувачем:

а) Мова повинна відповідати професійним характеристикам користувачів;

б) Відповідати потребам і задачам користувачів;

в) Відповідати призначенню і особливостям програмного продукту;

г) Бути зручною и легкою у використанні, ефективною в діяльності;

д) Відповідати нормам предметної області.

3.2. Функціональні вимоги.

3.2.1 Інформаційне середовище.

Інформація, з якою працюватимуть користувачі програмного продукту, зберігатиметься в базі даних.

Вхідними даними для роботи з програмним продуктом є дані описаної вище бази даних..

Вихідними даними слугуватимуть заповнена інформація про відвідування студентів.

3.2.2 Функції програмного забезпечення.

При розробці біометричної системи аутентифікації студентів необхідно реалізувати наступне:

Створити сторінку головного меню додатку, на якій будуть відображатися доступні операції – додавання обличчя, навчання системи , розпізнавання.

3.3. Нефункціональні вимоги (надійність, доступність, безпека та ін.).

3.3.1 Параметри, що характеризують ступінь відповідності програмного забезпечення призначенням.

Основним параметром, що характеризує ступінь відповідності програмного забезпечення призначенню є виконання функцій, що описані в п. 3.2.2.

3.3.2 Вимоги до надійності.

Основними показниками надійності програмного забезпечення є:

а) Безвідмовність програмного забезпечення;

б) Відновлюваність програмного забезпечення.

Необхідно проводити тестування функціональності програмного продукту після завершення написання програмного коду для кожного функціонального блоку. Також, при завершенні написання програмного коду потрібно провести тестування додатку, врахувавши всі можливі варіанти використання.

Позитивним результатом оцінки та контролю показників надійності буде вважатися відсутність помилок та відмов при тестуванні.

Київський національний торговельно-економічний університет

Факультет інформаційних технологій

Кафедра інженерії програмного забезпечення та кібербезпеки

Освітній ступінь магістр

Спеціальність 121 «Інженерія програмного забезпечення»

Затверджую
Зав. кафедри інженерії програмного забезпечення та кібербезпеки
Криворучко О. В.
8 листопада 2019 р.

Завдання
на випускний кваліфікаційний проект студентові

Моспан Олександр Вікторович
(прізвище, ім'я, по батькові)

1. Тема випускного кваліфікаційного проекту «Розробка біометричної системи аутентифікації обліку відвідування студентів»

Затверджена наказом ректора від "13"грудня 2019 р. № 4304

2. Строк здачі студентом закінченого проекту 01 грудня 2020

3. Цільова установка та вихідні дані до проекту

Мета проекту створення системи, яка зможе аутентифікувати студента та автоматично вести журнал його відвідування занять.

Об'єкт дослідження процес розробки програмного забезпечення

Предмет дослідження розробка системи для аутентифікації студентів

4. Консультанти проекту із зазначенням розділів, які консультують:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв

5. Зміст випускного кваліфікаційного проекту (перелік питань за кожним розділом)

ВСТУП

РОЗДІЛ 1 АНАЛІТИЧНА ТА ТЕОРЕТИЧНА ЧАСТИНА РОЗРОБКИ ПРОЕКТУ

1.1. Характеристика біометричної аутентифікації

1.2. Аналіз методів біометричної аутентифікації

1.3. Переваги та недоліки біометричної аутентифікації

1.4. Висновок до розділу 1

РОЗДІЛ 2. ВИБІР ТЕХНОЛОГІЙ ТА ПРОГРАМНИХ ЗАСОБІВ ДЛЯ РОЗРОБКИ ПРОЕКТУ

2.1. Вибір мови програмування

2.2. Інтерпретована об'єктно-орієнтована мова програмування Python

2.3. Django WEB Framework

2.4. Висновок до розділу 2

РОЗДІЛ 3. РЕАЛІЗАЦІЯ СИСТЕМИ БІОМЕТРИЧНОЇ АУТЕНТИФІКАЦІЇ

3.1. Перелік бібліотек та програмних модулів

3.2. Створення скелетону проекту та старт застосунку

3.3 Створення дизайну та оформлення сайту

ВИСНОВКИ ТА ПРОПОЗИЦІЇ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

ДОДАТКИ

6. Календарний план виконання проекту

№ пор.	Назва етапів випускного кваліфікаційного проекту	Строк виконання етапів проекту	
		за планом	фактично
1	2	3	4
1.	*Вибір теми випускної кваліфікаційної роботи*	*20.09.2019*	*20.09.2019*
2.	*Розробка та затвердження завдання на проект магістра*	*08.11.2019*	*08.11.2019*
3.	*Вступ та перелік літературних джерел*	*24.01.2020*	*24.01.2020*
4.	*Наукова стаття*	*01.09.2020*	*01.09.2020*
5.	*Технічне завдання*	*28.02.2020*	*28.02.2020*
6.	*Розділ 1.Аналіз предметної області застосування мобільного додатку*	*25.06.2020*	*25.06.2020*
7.	*Розділ 2. Вибір програмних засобів та проектування мобільного додатку*	*07.09.2020*	*07.09.2020*
8.	*Розділ 3. Реалізація мобільного додатку*	*19.10.2020*	*19.10.2020*
9.	*Програма та методика тестування*	*21.10.2020*	*21.10.2020*
10.	*Керівництво користувача*	*23.10.2020*	*23.10.2020*
11.	*Висновки та пропозиції*	*30.10.2020*	*30.10.2020*
12.	*Здача випускного кваліфікаційного проекту на кафедру (перша перевірка)*	*05.11.2020*	*05.11.2020*
13.	*Підготовка автореферату та презентації доповіді*	*05.11.2020*	*05.11.2020*
14.	*Попередній захист випускного кваліфікаційного проекту*	*25.11.2020-27.11.2020*	*25.11.2020-27.11.2020*
15.	*Зовнішнє рецензування випускної кваліфікаційної роботи*	*01.12.2020*	*01.12.2020*
16.	*Підготовка до публічного захисту випускної кваліфікаційної роботи*	*10.12.2020-11.12.2020*	

7. Дата видачі завдання «8» листопада 2019 р.

8. Науковий керівник випускного кваліфікаційного проекту Пашорін В.І.
(прізвище, ініціали, підпис)

9. Гарант освітньої програми Криворучко О.В.
(прізвище, ініціали, підпис)

10. Завдання прийняв до виконання студент Моспан О.В.
(прізвище, ініціали, підпис)

11. Відгук керівника випускного кваліфікаційного проекту

____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____
____

Науковий керівник випускного кваліфікаційного проекту
____

(підпис, дата)

Відмітка про попередній захист _____________Пашорін В.І.___________________

(ПІБ, підпис, дата)

12. Висновок про випускний кваліфікаційний проект

Випускний кваліфікаційний проект студента ____________________________

(прізвище, ініціали)

може бути допущена до захисту екзаменаційній комісії.

Гарант освітньої програми ______________________________ Криворучко О. В.

(прізвище, ініціали, підпис)

Завідувач кафедри ______________________________________ Криворучко О. В.

(підпис, прізвище, ініціали)

«_____»__________________20 ___ р.

KNUTE KHTEY

Реферат

Дипломна робота містить: 36 стоірнок, 9 рисунків, 10 посилань на літературні джерела.

Мета дослідження: створення системи, яка зможе аутентифікувати студента та автоматично вести журнал відвідування занять.

Об'єкт дослідження: процес розробки програмного забезпечення.

*Предмет дослідження***:** розробка системи для аутентифікації студентів.

У відповідності з метою дослідження поставлені наступні *завдання*:

1) визначити слабкі місця методів біометричної аутентифікації;

2) обґрунтувати вибір методу біометричної аутентифікації який буде використовуватись у додатку;

3) проаналізувати та обґрунтувати вибір програмного забезпечення та технологій, що використовуватимуться у процесі розробки;

4) провести тестування готового рішення на реальних пристроях та розробити інструкцію користувача.

Методи дослідження, що були використані у роботі: аналіз, порівняння, графічний метод і моделювання.*Наукова новизна дослідження*полягає в удосконаленні та використанні сучасних технологій та методів біометричної аутентифікації людини.

*Практичне значення дослідження***:** готове альтернативне рішення у виглядівеб-додатку, який зможе аутентифікувати студента університету, та автоматично вести журнал відвідування.

АНОТАЦІЯ

Відповідно до мети дослідження робота присвячена розробці біометричної системи аутентифікації студентів.

В результаті порівняльного аналізу методів біометричної аутентифікації були визначені недоліки та переваги даного підходу.

Розробка серверної частини виконана у середовищі розробки Pycharm,. Обрана мова програмування – Python. Окремі елементи системи розроблені із використанням HTML, CSS, відповідно до стандартів розробки.

Готовий програмний комплекс FaceRecognitionSystemбуло успішно протестовано відповідно до функціональних вимог.

Ключові слова: Python, Django, аутентифікація, біометрія.

ABSTRACT

According to the purpose the research is is devoted to the development of a biometric system for student authentication.

The comparative analysis of biometric authentication methods identified the disadvantages and advantages of this approach.

The development of the server part is performed in the Pycharm development environment. The chosen programming language is Python. Some elements of the system are designed using HTML, CSS, in accordance with development standards.

The ready-made software package FaceRecognitionSystem was successfully tested in accordance with the functional requirements.

Keywords: Python, Django, authentication, biometrics.

ЗМІСТ

ВСТУП 3
РОЗДІЛ 1 АНАЛІТИЧНА ТА ТЕОРЕТИЧНА ЧАСТИНА РОЗРОБКИ ПРОЕКТУ 5
1.1. Характеристика біометричної аутентифікації 5
1.2 Аналіз методів біометричної аутентифікації 7
1.3 Переваги та недоліки біометричної аутентифікації 9
Висновки до розділу 1 11
РОЗДІЛ 2 ВИБІР ТЕХНОЛОГІЙ ТА ПРОГРАМНИХ ЗАСОБІВ ДЛЯ РОЗРОБКИ ПРОЕКТУ .. 12
2.1. Вибір мови комп’ютерного програмування 12
2.2 Інтерпретована об'єктно-орієнтована мова програмування Python 14
2.3 Django Web Framework 16
2.4 Висновки до розділу 2 20
РОЗДІЛ 3 РЕАЛІЗАЦІЯ СИСТЕМИ БІОМЕТРИЧНОЇ АУТЕНТИФІКАЦІЇ 21
3.1. Перелік бібліотек та програмних модулів 21
3.2 Створення скелетону проекту та старт застосунку 24
3.3 Створення дизайну та оформлення сайту 29
ВИСНОВКИ ТА ПРОПОЗИЦІЇ 33
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 34
ТЕХНІЧНЕ ЗАВДАННЯ 35
ДОДАТКИ 37

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ПЗ – програмне забезпечення

ОС – операційна система

IDE (англ. Integrated Development Environment) – інтегроване середовище розробки

HTML (англ. HyperText Markup Language) - мова розмітки гіпертексту

CSS (англ. Cascading Style Sheets) — каскадні таблиці стилів

ВСТУП

Актуальність. У сучасному суспільстві точна ідентифікація стає дуже важливою і дедалі важчою. Ряд ситуацій вимагає ідентифікації особи: чи я бачив цього заявника раніше? Ця людина є працівником цієї компанії? Чи є ця особа громадянином цієї країни? Загроза тероризму все ж підняла важливість точної та бездоганної ідентифікації особи. Тому, хоча ідентифікація людини є проблемою, її можна трактувати як чудову можливість. Точне встановлення особи може зупинити злочин та шахрайство, впорядкувати бізнес-процеси та заощадити критичні ресурси.

Аутентифікація користувачів є найважливішим компонентом більшості комп'ютерних систем, забезпечує контроль доступу та розмежування прав для користувачів. Як ми знаємо, на ринку існує багато типів систем аутентифікації користувачів, але буквено-цифрові імена/ паролі є найпоширенішим типом аутентифікації користувачів. Їх багато і легко реалізувати та використовувати. Через обмеження людської пам’яті більшість користувачів зазвичай вибирають короткі або прості паролі, які легко запам’ятати. Опитування показують, що частими паролями є особисті імена членів сім'ї, дата народження або слова зі словника. У більшості випадків ці паролі легко вгадати і вразливі до атаки за словником. Сьогодні користувачі мають багато паролів для персональних комп’ютерів, соціальних мереж, електронної пошти тощо. Вони можуть вирішити використовувати один пароль для всіх систем, щоб зменшити навантаженняна пам'ять, що зменшує безпеку. Тому все більшої популярності набуває біометрична аутентифікація. Біометрична аутентифікація - це одна із

форм захисту, яка вимірює та відповідає біометричним характеристикам користувача, щоб перевірити, чи особа, яка намагається отримати доступ, має на це дозвіл. Біометричні ознаки - це фізичні та біологічні характеристики, які є унікальними для окремої людини і їх можна легко порівняти з дозволеними ознаками, збереженими в базі даних.

Мета дослідження: створення системи, яка зможе аутентифікувати студента та автоматично вести журнал відвідування занять.

Об'єкт дослідження: процес розробки програмного забезпечення.

*Предмет дослідження***:** розробка системи для аутентифікації студентів.

У відповідності з метою дослідження поставлені наступні *завдання*:

1) визначити слабкі місця методів біометричної аутентифікації;

2) обґрунтувати вибір методу біометричної аутентифікації який буде використовуватись у додатку;

3) проаналізувати та обґрунтувати вибір програмного забезпечення та технологій, що використовуватимуться у процесі розробки;

4) провести тестування готового рішення на реальних пристроях та розробити інструкцію користувача.

Методи дослідження, що були використані у роботі: аналіз, порівняння, графічний метод і моделювання.*Наукова новизна дослідження*полягає в удосконаленні та використанні сучасних технологій та методів біометричної аутентифікації людини.

*Практичне значення дослідження***:** готове альтернативне рішення у виглядівеб-додатку, який зможе аутентифікувати студента університету, та автоматично вести журнал відвідування.

РОЗДІЛ 1
АНАЛІТИЧНА ТА ТЕОРЕТИЧНА ЧАСТИНА РОЗРОБКИ ПРОЕКТУ

1.1. Характеристика біометричної аутентифікації

Біометрія - це фізичні або поведінкові характеристики людини, які можуть бути використані для цифрової ідентифікації людини для надання доступу до систем, пристроїв або даних.

Прикладами цих біометричних ідентифікаторів є відбитки пальців, візерунки обличчя, каденція голосу або друку. Кожен із цих ідентифікаторів вважається унікальним для особистості, і вони можуть використовуватися в поєднанні для забезпечення більшої точності ідентифікації.

Оскільки біометрія може забезпечити розумний рівень впевненості в автентичності людини з меншим тертям для користувача, вона може потенційно значно покращити безпеку підприємства. Комп'ютери та пристрої можуть розблоковуватися автоматично, коли виявляють відбитки пальців схваленого користувача. Двері серверної кімнати можуть відкритися, коли вони розпізнають обличчя надійних системних адміністраторів. Системи довідкової служби можуть автоматично витягувати всю відповідну інформацію, коли вони розпізнають голос працівника на лінії підтримки.

Згідно з недавнім опитуванням Ping Identity, 92 відсотки підприємств оцінюють біометричну автентифікацію як "ефективну" або "дуже ефективну" для захисту особистих даних, що зберігаються в приміщеннях, а 86 відсотків вважають, що вона є ефективною для захисту даних, щозберігаються в загальнодоступній хмарі. Ще одне опитування, опубліковане минулого

рокукомпанією Spiceworks, повідомляє, що 62 відсотки компаній вже використовують біометричну аутентифікацію, а ще 24 відсотки планують впровадити її протягом найближчих двох років.

Однак компаніям слід бути обережними щодо того, як вони впроваджують свої біометричні системи автентифікації, щоб уникнути порушення конфіденційності співробітників чи клієнтів або неналежного викриття конфіденційної інформації. Зрештою, хоча легко видати новий пароль, коли старий був скомпрометований, ви не можете видати комусь нове очне яблуко.

Згідно з опитуванням Spiceworks, 48 відсотків називають ризики викрадених біометричних даних головним ризиком для безпеки цієї технології. Інші бар'єри для усиновлення включають витрати, на які посилаються 67 відсотків респондентів.

За даними опитування Ping Identity, для компаній, які спеціально використовують біометричні дані для захисту IT-інфраструктури у хмарі, SaaS, попередніх та гібридних середовищах, рівень прийняття ще нижчий. Лише 28 відсотків компаній використовують біометричні дані в приміщеннях, а ще менше, 22 відсотки, використовують їх для хмарних додатків.

1.2 Аналіз методів біометричної аутентифікації

Біометричний ідентифікатор - це той, який пов'язаний із внутрішніми характеристиками людини. Вони приблизно поділяються на дві категорії: фізичні ідентифікатори та поведінкові ідентифікатори. Фізичні ідентифікатори, здебільшого, незмінні та незалежні від пристрою:

Сканери відбитків пальців: сканери відбитків пальців, цифрова версія старомодних відбитків пальців із чорнила та паперу, покладаються на запис унікальних візерунків завихрень та гребенів, що складають відбитки пальців людини. Сканери відбитків пальців - це один із найпоширеніших і доступних режимів біометричної аутентифікації, хоча версії споживчого класу, ті, що знаходяться на смартфонах, все ще мають можливість помилкових спрацьовувань. Незважаючи на випадкові неточності, сканери відбитків пальців є одними з найпопулярніших та найвищих біометричних технологій для повсякденних споживачів.

Розпізнавання обличчя: Технологія розпізнавання обличчя спирається на поєднання десятків різних вимірювань від затвердженого обличчя до обличчя користувача, який намагається отримати доступ, створюючи так звані відбитки обличчя. Як і сканери відбитків пальців, якщо достатня кількість вимірювань від користувача відповідає затвердженому обличчю, доступ надається. Розпізнавання обличчя було додано до ряду смартфонів та іншихпопулярних пристроїв, хоча воно може виявлятися непослідовним при порівнянні облич при розгляді з різних сторін або при спробі розрізнити людей, схожих на людей, наприклад, близьких родичів.

Ідентифікація голосу: Технології розпізнавання голосу вимірюють вокальні характеристики, щоб розрізнити людей. Як і сканери обличчя,вони поєднують ряд точок даних і створюють профіль голосового відбитку для

порівняння з базою даних. Замість того, щоб «слухати» голос, технології ідентифікації голосу зосереджуються на вимірюванні та дослідженні рота та горла мовця для формування певних форм та звукових якостей. Цей процес дозволяє уникнути проблем із безпекою, які можуть бути спричинені спробами замаскувати або імітувати голос, або загальними станами, такими як хвороба або час доби, які можуть змінити звукові якості голосу на людське вухо. Слова, які користувач промовляє для доступу до захищеного голосом пристрою, також можуть бути дещо стандартизованими, слугуючи своєрідним паролем та полегшуючи порівняння затверджених голосових відбитків до унікального голосового відбитка користувача, а також заважаючи певним способам обійти порівняння голосових відбитків, наприклад як запис уповноваженого користувача, який каже щось не пов'язане.

Сканери для очей:сканери для очей працюють, проектуючи яскраве світло на око, яке робить видимими схеми кровоносних судин, які потім можна прочитати сканером і порівняти із схваленоюінформацією, збереженою в базі даних. Сканери райдужної оболонки працюють подібно, цього разу шукаючи унікальні візерунки в кольоровому кільці навколо зіниці ока. Обидва типи сканерів очей корисні як параметри перевірки у режимі “вільні руки”, але все одно можуть зазнати неточностей, якщо суб’єкти носять контактні лінзи або окуляри. Фотографії також використовувались для обману сканерів очей, хоча цей метод, ймовірно, стане менш життєздатним, оскільки сканери стають все більш досконалими та включають такі фактори, як рух очей, у свої схеми перевірки.

1.3 Переваги та недоліки біометричної аутентифікації

Хоча біометричні дані, як правило, більш безпечні, вони не є надійними. Наприклад, сканери відбитків пальців на смартфонах часто покладаються на часткові збіги, і дослідники виявили, що можна створити "основні відбитки", які збігаються з частковими частками, досить добре, щоб надати доступ до великої кількості облікових записів користувачів.

Дослідники також продемонстрували здатність створювати фальшиві відбитки пальців із високоякісних відбитків, що залишилися. Інші знайшли способи використовувати фотографії або 3D-принти, щоб обдурити сканери райдужної оболонки ока або системи розпізнавання обличчя. Іноді проблема полягає в тому, що систему можна зламати настільки, наскільки вона занадто часто не розпізнає дійсного користувача: хтось у різному макіяжі чи нових окулярах, голос користувача, який захворів або щойно прокинувся.

Тож не дивно, що якісні біометричні рішення коштують дорожче. Насправді 67 відсотків IT-спеціалістів називають вартість найбільшою причиною відмови від біометричної аутентифікації. Також існують приховані витрати, оскільки 47% опитаних повідомляють про необхідність оновлення систем, щоб підтримати перехід до біометрії.

Проаналізувавши найпоширеніші методи біометричної аутентифікації, можемо зробити певні висновки. Переваги біометричних систем:

- Швидкість аутентифікації: за відбитком пальців або формі обиччя, електронно-аналітичні пристрої пізнають людини протягом однієї-двох секунд;
- Високий рівень безпеки: біометричні ознаки людини неповторні, що зводить до мінімуму кількість помилок при впізнанні;Дані біометричних характеристик неможливо втратити або забути;
- Пристрої біометричної аутентифікації зручні в користуванні і бюджетних в експлуатації.

Недоліки біометричних систем:

- Біометричні характеристики можна змінити в поточній базі даних - на відміну від паролів, вони пов'язані з конкретною особистістю протягом усього її життя;
- Через вікові зміни, травми, ампутації, потрібне постійне оновлення еталонних моделей порівняння, які вносяться в пам'ять електронно-обчислювальних пристроїв;
- Для створення зразків біометрії потрібні спеціальні зчитувальні пристрої;
- Біометричні характеристики неможливо зберегти в секреті, тому досвідчені зловмисники можуть підробити зразки відбитків пальців або долонь.

Висновки до розділу 1

В дослідженні теоретичних відомостей було розглянуто і описано найпоширеніші методи біометричної аутентифікації, проаналізовано їх переваги та недоліки. Аналізуючи результати проведеного вище дослідження, для реалізації аутентифікації в додатку було обрано метод розпізнавання обличчя.

РОЗДІЛ 2
ВИБІР ТЕХНОЛОГІЙ ТА ПРОГРАМНИХ ЗАСОБІВ ДЛЯ РОЗРОБКИ ПРОЕКТУ

2.1. Вибір мови комп’ютерного програмування

Мови комп’ютерного програмування дозволяють давати вказівки комп’ютеру мовою, яку він розуміє. Так само, як існує багато мов, заснованих на людині, існує безліч мов комп'ютерного програмування, які програмісти можуть використовувати для спілкування з комп'ютером. Частина мови, яку може зрозуміти комп’ютер, називається „двійковою”. Переклад мови програмування у бінарний файл відомий як "компіляція". Кожна мова, від мови C до Python, має свої особливі особливості, хоча багато разів між мовами програмування є спільні риси.

Ці мови дозволяють комп’ютерам швидко та ефективно обробляти великі та складні масиви інформації. Наприклад, якщо людині надається список рандомізованих чисел від однієї до десяти тисяч і його просять розмістити у порядку зростання, швидше за все, це займе значну кількість часу і включатиме деякі помилки.

На сьогоднішній день існує багато різних мов програмування багатьох типів: мови низького та високого рівня, мови функціональні, структурні, об‘єктно орієнтовані та навіть езотеричні мови програмування. Кожна з цих мов має своє призначення, переваги та недоліки в порівнянні з іншими. Проаналізувавши літературні та інтернет-джерела можна з впевненістю сказати, що найпопулярнішими на даний час є такі мови:C# – це мова, яка не має

великого базового функціоналу, але може виконувати поставлені задачі швидше, за інші мови;

3 Java – мова програмування, що дозволяє створювати універсальні програми для різник операційних систем (далі - ОС). І немає різниці, що за ОС встановлена на пристрої – програма буде виконуватись у віртуальній машині Java Virtual Machine;

4 PHP, JavaScript та мови розмітки гіпертекстових документів HTML та CSS – мови, що використовуються для створення веб-сторінок та веб-додатків для інтернет-браузерів.

5 Python — інтерпретована об’єктно-орієнтована мова програмування високого рівня зі строгою динамічною типізацією.

Всі говорять про Python, і недарма. Завдяки експоненціально зростаючому співтовариству навколо науки про дані, машинного навчання, штучного інтелекту, веб-розробників тощо, Python - це мова, яка відкриває доступ до програмування у всьому світі.

Серед серверних програмних мов Python вважається простим для читання, написання та вивчення. Крім того, це надзвичайно масштабовано. На ньому працюють найбільші світові компанії, і це також чудово підходить для швидких і простих прототипів.

2.2 Інтерпретована об'єктно-орієнтована мова програмування Python

Python — інтерпретована об'єктно-орієнтована мова програмування високого рівня зі строгою динамічною типізацією. Розроблена в 1990 році Гвідо ван Россумом. Структури даних високого рівня разом із динамічною семантикою та динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також як засіб поєднування наявних компонентів. Python підтримує модулі та пакети модулів, що сприяє модульності та повторному використанню коду. Інтерпретатор Python та стандартні бібліотеки доступні як у скомпільованій, так і у вихідній формі на всіх основних платформах.

В мові програмування Python підтримується кілька парадигм програмування, зокрема: об'єктно-орієнтована, процедурна, функціональна та аспектно-орієнтована.

Для чого використовується Python? Python використовується в Data Mining, Data Science, AI, Machine Learning, Web Development, Web Frameworks, Embedded Systems, Graphic Design applications, Gaming, Network development, development product, Rapid Application Application, Testing, Automation, тощо. Python використовується як простіша та ефективніша написана альтернатива таким мовам, як C, R та Java, і зростає у популярності як основна мова для багатьох додатків.

Python використовується вData Science.Історично мова програмування R найчастіше використовується для науки про дані, з багатою екосистемою бібліотек для статистичного аналізу. Оскільки Python-код вважається простішим у обслуговуванні та більш масштабованим, ніж R, популярність Python зросла в галузі науки про дані - особливо серед професіоналів, які не мають вищої освіти у галузі статистики та математичних областей.

Які компанії використовують Python?

Python чудово підходить для швидкого прототипування, тому широко використовується стартапами для створення свого першого мінімально життєздатного продукту (MVP). Як дуже масштабована мова, Python також використовується у великих компаніях. У своєму недавньому дописі в блозі Netflix обговорив використання Python у всьому - від їх Мережі доставки вмісту (CDN) до своїх систем моніторингу.

Переваги:

- Простий у вивченні, особливо - на початкових етапах.
- Відкрита розробка.
- Синтаксис з його особливостями стимулюють писати легко читаємий код.
- Наявність великої спільноти, яка добре налаштована по відношенню до новачків.Величезна кількість корисних бібліотек і розширень мови з легким використанням в своїх проектах – гранично уніфікований механізм імпорту.
- Абсолютно все є об'єктами в сенсі ООП, при цьому програмісту не нав'язується об'єктний підхід.

Недоліки:

- Не дуже зручна підтримка багатопоточності.
- Не така велика кількість якісних програмних проектів по порівняно з іншими універсальними мовами програмування, наприклад, з Java.
- Початкова обмеженість інструментів по роботі з базами даних.

2.3 Django Web Framework

Для написання серверної частини системи буде використовуватися Django Web Framework. Django - це фреймворк з відкритим кодом для веб-додатків, що базуються на основі Python - однієї з найкращих мов веб-розробки. Основними цілями є простота, гнучкість, надійність та масштабованість.

Django має власну систему іменування всіх функцій та компонентів (наприклад, відповіді HTTP називаються "views"). Він також має адміністративну панель, з якою легше працювати, ніж у Lavarel або Yii, та інші технічні особливості, зокрема:

- Простий синтаксис;
- Власний веб-сервер;
- Архітектура ядра MVC (Model-View-Controller);
- «Батареї в комплекті» (постачається з усім необхідним для вирішення загальних справ);
- ORM (Object Relational Mapper);
- Бібліотеки HTTP;
- Підтримка проміжного програмного забезпечення;

Крім того, Django надає динамічний CRUD-інтерфейс (create, read, update, delete) інтерфейс, налаштований на моделіадміністратора та генерований за допомогою самоаналізу. CRUD використовується для опису основних команд бази даних, що означає, що інтерфейс полегшує перегляд, зміну та пошук інформації. Чому саме фреймворк Django? Це швидко і просто. Однією з головних цілей Django є спрощення роботи для розробників. Для цього фреймворк Django використовує:

- Принципи швидкого розвитку, що означає, що розробники можуть робити кілька ітерацій за раз, не починаючи весь графік з нуля;
- DRY філософія – Don’t Repeat Yourself- це означає, що розробники можуть повторно використовувати наявний код і зосередитись на унікальному.

• Як результат, для виведення проекту на ринок потрібно набагато менше часу.

• швидкий розвиток

• Це безпечно

Безпека також є пріоритетом для Django. У ньому є одна з найкращих нестандартних систем безпеки, і вона допомагає розробникам уникати загальних проблем безпеки, зокрема

• clickjacking,

• міжсайтові сценарії та

SQL-ін’екції.Django регулярно отримує оновлення безпеки.

• Це підходить для будь-якого проекту веб-додатків

За допомогою Django ви можете вирішувати проекти будь-якого розміру та потужності, будь то простий веб-сайт або веб-додаток із великим навантаженням. Навіщо використовувати Django для свого проекту?

• Він повністю завантажений додатковими функціями та масштабований, тому ви можете створювати додатки, які обробляють інтенсивний трафік та великий обсяг інформації;

• Це крос-платформний, що означає, що ваш проект може базуватися на Mac, Linux або PC;

• Він працює з більшістю основних баз даних і дозволяє використовувати базу даних, яка більше підходить для конкретного проекту, або навіть кілька баз даних одночасно.

- • Django перевірено часом і натовпом. У ньому є велика підтримка спільнота, до якої можна отримати доступ через численні форуми, канали та спеціальні веб-сайти. Легко знайти допомогу, коли в коді є проблемна функція, і знайти розробників, якщо ваша компанія хоче створювати наступний проект на Django.
- Django почав із чудової документації, найкращої з будь-яких інших фреймворків з відкритим кодом. І він все ще підтримується

нависокому рівні, оновлюється разом із новими функціями та виправленнями, так що ви можете легко адаптуватися до змін.

- • Ви можете довіряти, що будь-які проблеми з фреймворком будуть вирішені, як тільки вони виникнуть. Програмне забезпечення постійно оновлюється та випускаються нові пакети, що роблять роботу з Django зручнішою, ніж вона є.
- • Структура Django добре налагоджена
- Що ви можете зробити з Django?
- • Фінансові платформи з функціями для аналізу та обчислення приблизних результатів на основі особистих даних, толерантності до ризику, ймовірності досягнення цілей;
- • Вбудовані власні CRM-системи для внутрішніх даних;
- • B2B CRM-системи для обміну інформацією між підприємствами;
- • Платформи, що полегшують спілкування між двома сторонами, такими як бізнес та споживач;
- • Двигуни бронювання з великим навантаженням або торгові майданчики;

• Мобільні програми для Android та iOS, які підтримують веб-програми;

• Системи оцінки майна нерухомості;

Системи управління документами;Платформи для вирішення правових питань, таких як перевірка умов іпотеки або статусу оренди.

• Деякі компанії вирішують базувати свої проекти на більш ніж одній структурі. Django також можна використовувати для створення окремих функцій, таких як:

• Система електронної пошти для надсилання сповіщень користувачам;

• Система фільтрації з вдосконаленою логікою та динамічно мінливими правилами;

• Генератори на основі алгоритму;

• Інструменти аналізу даних;

• Інтерфейси для управління інвестиційними фондами;

• Інформаційні панелі адміністратора;

• Системи перевірки на основі фотографій;

А також інші функції, що полегшують розвиток CRM та B2B платформ, онлайнових ринків, систем бронювання тощо.

2.4 Висновки до розділу 2

Отже, проаналізувавши сучасні методи та технлогії розробки, їх переваги та недоліки, для написання системи біометричної аутентифікації була обрана мова програмування Python, та веб-фреймворк Django.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ СИСТЕМИ БІОМЕТРИЧНОЇ АУТЕНТИФІКАЦІЇ
3.1. Перелік бібліотек та програмних модулів

Для коректної роботи біометричної системи аутентифікації потрібен пристрій, за допомогою якого можна отримати та додати зображення користувача, та необхідне програмне забезпечення для роботи з ним. З цим з легкістю впорається USB-Web камера Для створення зв'язку між вхідним зображенням та програмним кодом використовується бібліотека OpenCV Python. OpenCV (Open Source Computer Vision Library) - це бібліотека програмного забезпечення для комп'ютерного зору та машинного навчання. OpenCV був побудований для забезпечення загальної інфраструктури для програм комп'ютерного зору та прискорення використання машинного сприйняття в комерційних продуктах. Як продукт із ліцензією BSD, OpenCV полегшує бізнесу використання та модифікацію коду.

Бібліотека має понад 2500 оптимізованих алгоритмів, що включає в себе повний набір як класичних, так і найсучасніших алгоритмів комп'ютерного зору та машинного навчання. Ці алгоритми можна використовувати для виявлення та розпізнавання облич, ідентифікації об'єктів, класифікації людських дій у відео, відстеження рухів камери, відстеження рухомих об'єктів, вилучення 3D-моделей об'єктів, створення 3D-хмар точок зі стереокамер, зшивання зображень для отримання високої роздільної здатності зображення цілої сцени, знайти подібні зображення з бази даних зображень, видалити червоні очі із зображень, зроблених за допомогою спалаху, слідкувати зарухами очей, розпізнавати

декорації та встановлювати маркери, щоб накласти їх на доповнену реальність тощо. OpenCV має понад 47 тис. користувачів спільноти та передбачає кількість завантажень, що перевищує 18 млн. Бібліотека широко використовується у компаніях, дослідницьких групах та державних органах

OpenCV має інтерфейси C ++, Python, Java та MATLAB та підтримує Windows, Linux, Android та Mac OS. OpenCV схиляється здебільшого до програм зору в режимі реального часу та використовує переваги інструкцій MMX та SSE, коли вони доступні. Зараз розробляються повнофункціональні інтерфейси CUDA та OpenCL. Існує понад 500 алгоритмів і приблизно в 10 разів більше функцій, які складають або підтримують ці алгоритми. OpenCV написаний власною мовою на C ++ і має шаблонний інтерфейс, який безперебійно працює з контейнерами STL.

Що ж таке Computer Vision?

Наші очі посилають сигнали нашому мозку, і він аналізує те, що ми бачимо; ми можемо розпізнати обличчя, предмети, рухи та визначити, чи є щось хорошим чи поганим за певним сценарієм. Таким чином, комп'ютерний зір (CV) хоче досягти того, що роблять наші очі. Він має справу з комп’ютерами та іншим електронним обладнанням для отримання інформації за допомогою цифрових зображень чи відео. Він також може аналізувати складні зображення, виконувати порівняння та встановлювати відмінності.

Комп’ютерний зір дуже корисний для зростання технологій у нашому сучасному суспільстві, оскільки вони еволюціонували від теорії до реальності.Важливість комп'ютерного зору була корисною для багатьох підприємств і життєво важливою для деяких. Компанії використовують комп'ютерний зір для OCR, біометрію зору, розпізнавання об'єктів, спецефекти, 3D-друк та захоплення зображень, спорт, розумні машини, медичну візуалізацію та багато інших.

З точки зору IT-спеціаліста, вони прагнуть автоматизувати завдання, що включають візуалізацію. Таким чином, це спричинило величезні події, що призвели до величезного інтересу як з боку підприємців, так і з боку постачальників програмного забезпечення. Це люди з усіх напрямків життя з різними рівнями кваліфікації, які співпрацюють, щоб згладити тертя між продуктивністю та комерційністю. Крім того, вони прагнуть знайти баланс складності комп'ютерного зору та ідеально подати його до світу.

Комп’ютерний зір має на меті полегшити наше життя, оскільки його можна застосовувати майже у всіх сферах. Ці програми створюють героїв у світі обчислювальної техніки - таких героїв, як клієнти та розробники програмного забезпечення. Клієнти, які хочуть вирішити проблеми CV із низьким рівнем, використовуючи перспективні та сміливі підходи. Плюс, фантастично талановиті розробники згуртовано розробляють системи CV над комп’ютерами та іншим обладнанням.

Ще один не менш важливий та сучасний інструмент розробки — dlib, що містить алгоритми машинного навчання та інструменти для створення складного програмного забезпечення на C ++ для вирішення реальних проблем. Він використовується як у промисловості, так і в наукових колах у широкомудіапазоні доменів, включаючи робототехніку, вбудовані пристрої, мобільні телефони та великі високопродуктивні обчислювальні середовища. Ліцензування відкритого коду Dlib дозволяє вам використовувати його в будь-якому додатку безкоштовно.

3.2 Створення скелетону проекту та старт застосунку

Для створення проекту використовується професійне інтегроване середовище розробки PyCharm Professional Edition від JetBrains. Основні переваги IDE Pycharm:

- **Редактор коду.** Інтелектуальний редактор коду, який надає PyCharm, дозволяє програмістам писати високоякісний код Python. Редактор дозволяє програмістам легко читати код за допомогою колірних схем, автоматично вставляти відступи в нові рядки, вибирати відповідний стиль кодування та користуватися контекстними пропозиціями щодо заповнення коду.
- **Навігація коду.** Параметри інтелектуальної навігації кодом, що надаються PyCharm, допомагають програмістам редагувати та вдосконалювати код, не докладаючи зайвого часу та зусиль. IDE полегшує програмістам перехід до класу, файлу та символів, а також до оголошень переходу до посилань.
- **Рефакторинг**. PyCharm полегшує розробникам швидке та ефективне впровадження як локальних, так і глобальних змін. Розробники можуть навіть скористатися перевагами параметрів рефакторингу, передбачених IDE, під час написання простого коду Python та роботи з фреймворками Python.
- **Підтримка популярних веб-фреймворків Python.** На додаток до підтримки часто використовуваних веб-технологій, PyCharm також забезпечує першокласну підтримку надійної веб-платформи Python, такої як Django.**Інструменти бази даних**. На додаток до підтримки різних бібліотек та фреймворків Python, PyCharm дозволяє розробникам працювати з низкою реляційних баз даних, включаючи Oracle, SQL Server, MySQL та PostgreSQL.
- **Тестування програмного забезпечення**. Як і інші IDE, PyCharm також має функції та інструменти для спрощення тестування додатків Python. Це

дозволяє розробникам проводити модульне тестування за допомогою таких популярних фреймворків для тестування Python, як Nose, Attest та Doctests.

Для початку створимо новий Python проект faceRecognitionSystem(Рис 3.1)

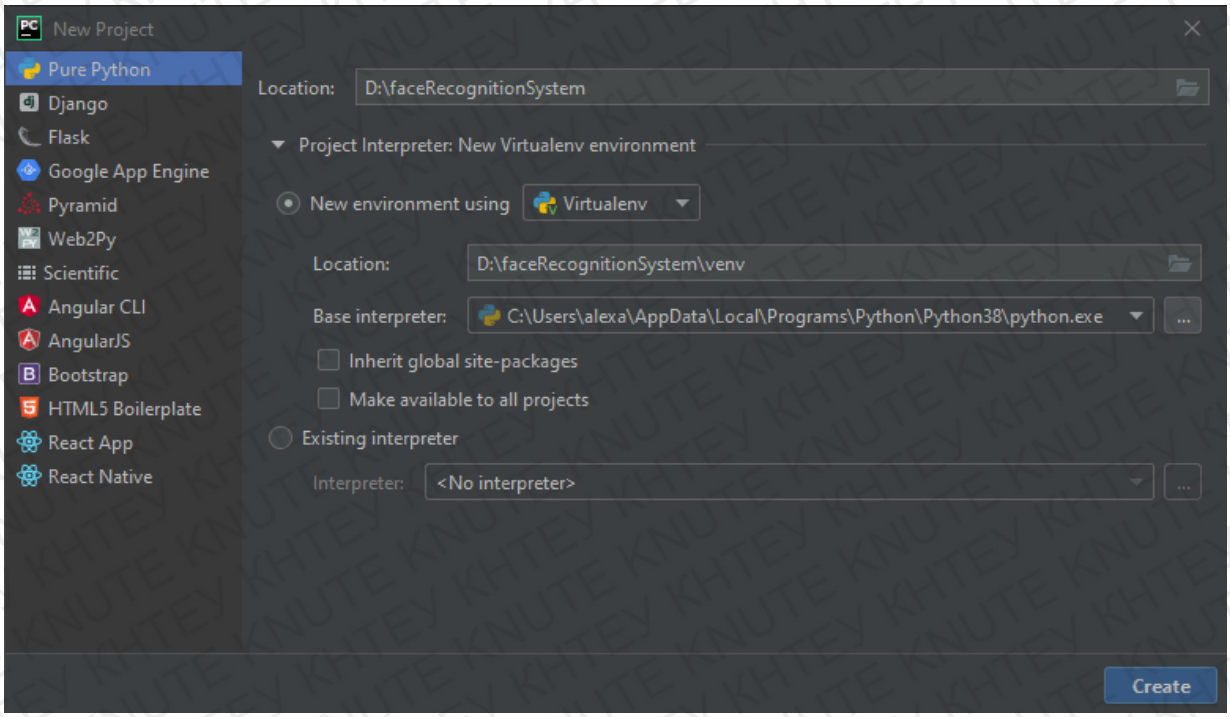

Рис. 3.1 Вікно створення нового проекту

В якості інтерпретатора використовується Python версії 3.6. В результаті отримуємо порожню директорію проекту та встановлюємо необхідні пакети за допомогою системи управління пакетами Python – Pipm, попередньо переконавшись в тому, що віртуально середовище створено та активовано. Спочатку встановимо основний пакет — django командою:

```
$pip install django
```

Після чого сформуємо скелетон нашого додатку, створивши новий проект (Рис. 3.2)

```
$ django-admin startproject faceRecognitionSystem
```

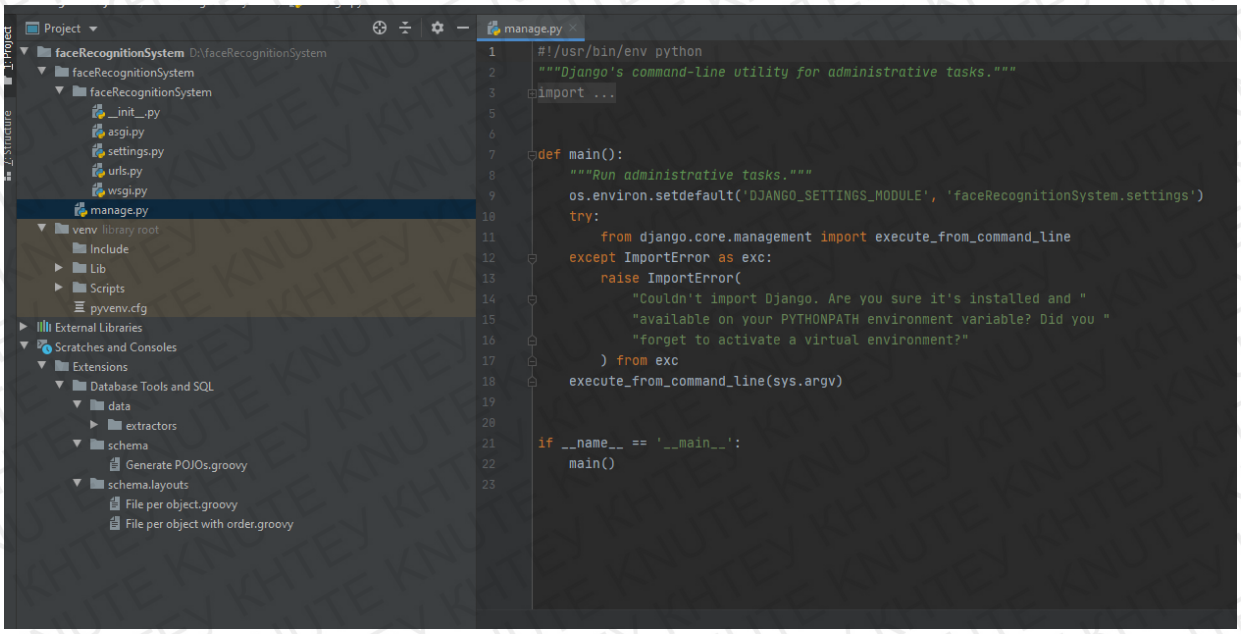

Рис. 3.2 скелетон веб-сайту

Зовнішній каталог **mysite / root** є контейнером для вашого проекту. Його назва не має значення для Django; Ви можете перейменувати його на все, що вам подобається.**manage.py**: Утиліта командного рядка, яка дозволяє взаємодіяти з цим проектом Django різними способами.

Внутрішній каталог **mysite /** є фактичним пакетом Python для вашого проекту. Його ім'я - ім'я пакету Python, яке вам потрібно буде використовувати для імпорту будь-чого всередині нього (наприклад, mysite.urls).

- **mysite / __ init__.py**: порожній файл, який повідомляє Python, що цей каталог слід вважати пакетом Python.
- **mysite / settings.py**: Налаштування / конфігурація для цього проекту Django.
- **mysite / urls.py**: Оголошення URL-адрес для цього проекту Django; "зміст" вашого веб-сайту, що працює на Django.
- **mysite / asgi.py**: точка входу для веб-серверів, сумісних з ASGI, для обслуговування вашого проекту.
- **mysite / wsgi.py**: Точка входу для веб-серверів, сумісних з WSGI, для обслуговування вашого проекту.

Створимо додаток, який буде виконувати основну функцію проекта.

$python manage.py startapp face_rec

Після чого відкриємо файл views.py, та запишемо туди програмний код, який буде відповідати за підключення камери, створення директорій для збереження зображень та виконувати функції аутентифікації(Рис. 3.3)

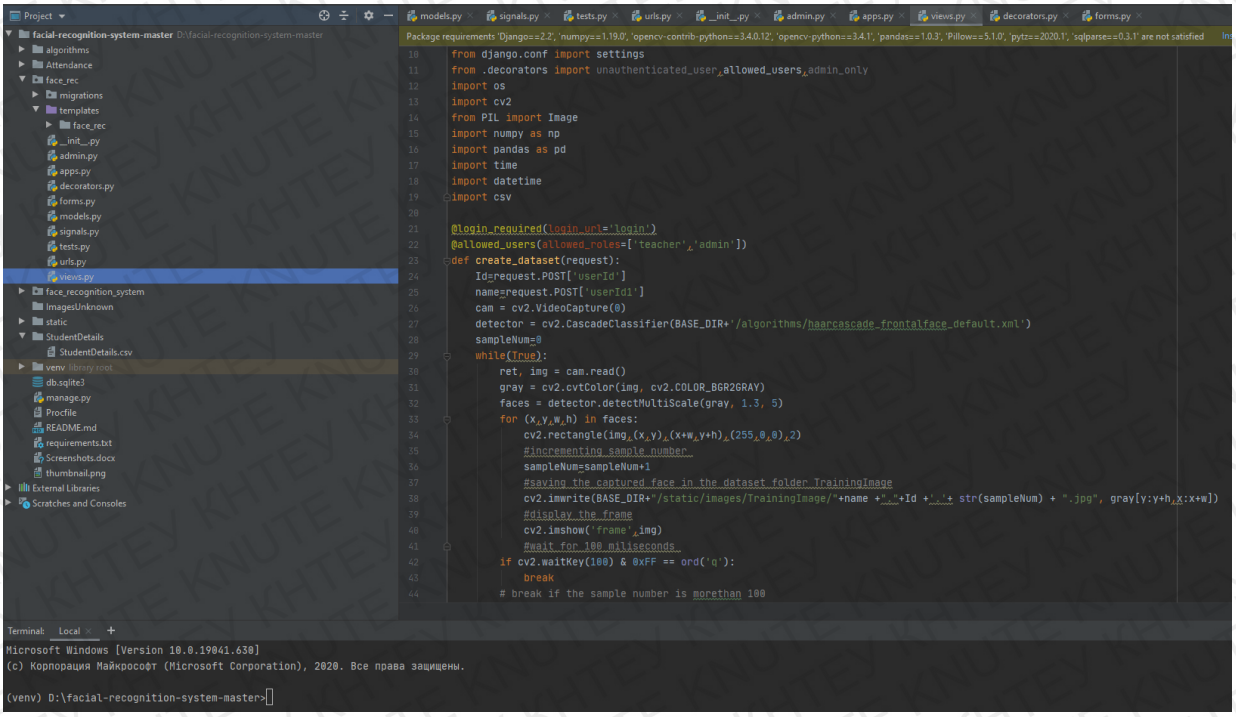

Рис. 3.3 файл views.py додатку face_rec

Перехід між посиланнями сайту налаштовується у файлі urls.py, який потрібно попередньо створити. Заповнимо його, для відслідковування всіх посилань, по яким може переходити користувач(Рис. 3.4).

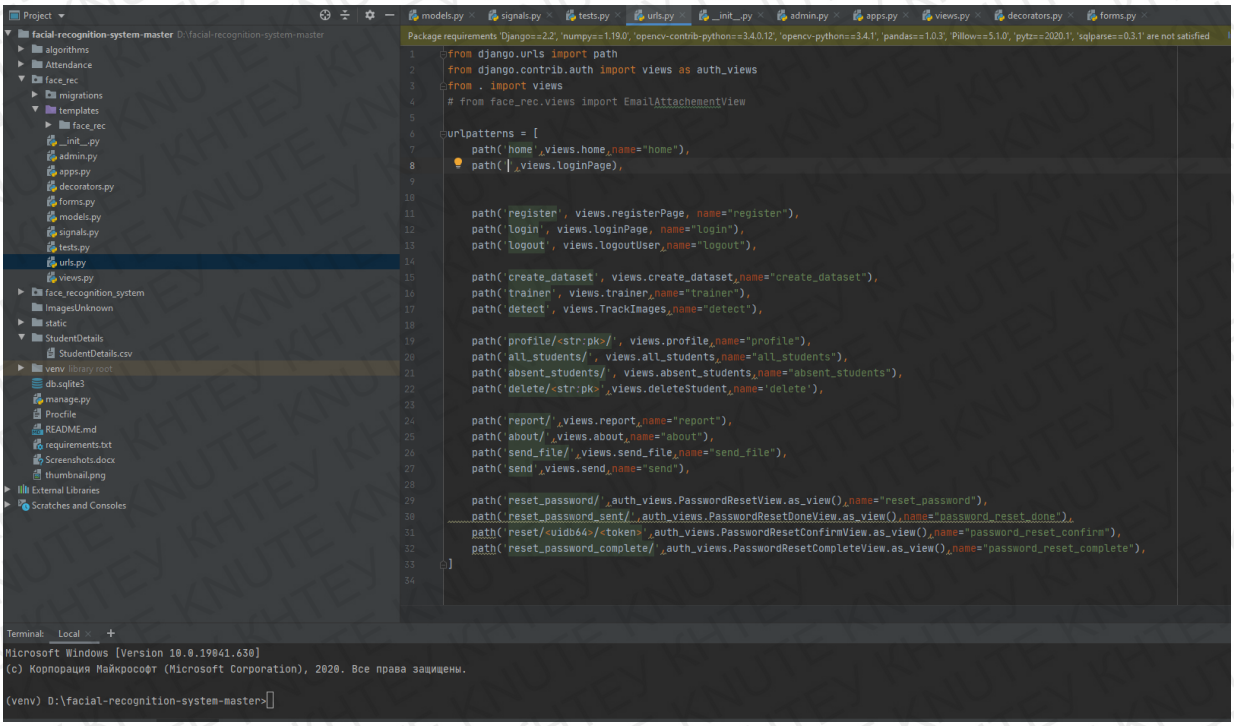

Рис 3.4 urls.py – файл, що відповідає за зміст сайту.

Для збереження інформації про студентів та їх відвідування, створимо дві моделі в файлі models.py: class student_profile(models.Model) та class student_attendance(models.Model)(Рис 3.5),Рис 3.5 моделі для створення таблиць в Django

3.3 Створення дизайну та оформлення сайту

Після написання лоігки сайту, потрібно створитиFront-End частину сайту за допомогою мови гіпертекстової розмітки HMTL та каскадної таблиці стилів CSS. Для цього в директорії додатку face_rec створимо папку templates, куди будемо зберігати файли для створення дизайну.

Приклад оформлення сторінки авторизації /login (Рис 3.6):

Рис 3.6 Сторінка авторизації користувача

Сторінка реєстрації для нового користувача /register (Рис 3.7)

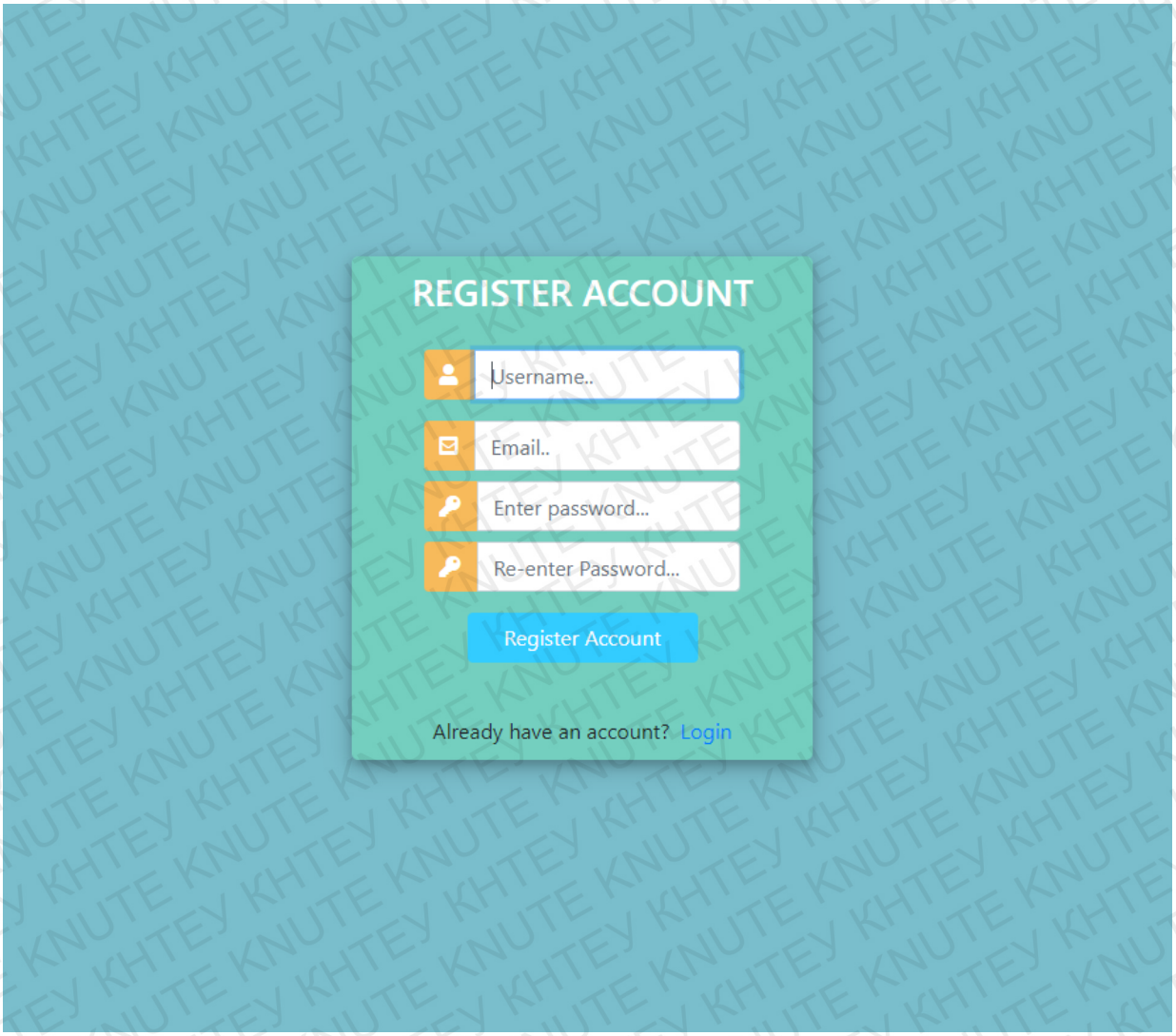

Рис 3.7 форма реєстрації

Головна сторінка сайту /home/ (Рис 3.8), де можна додати нове обличчя до системи та навчити розпізнавати та кнопка аутентифікації.

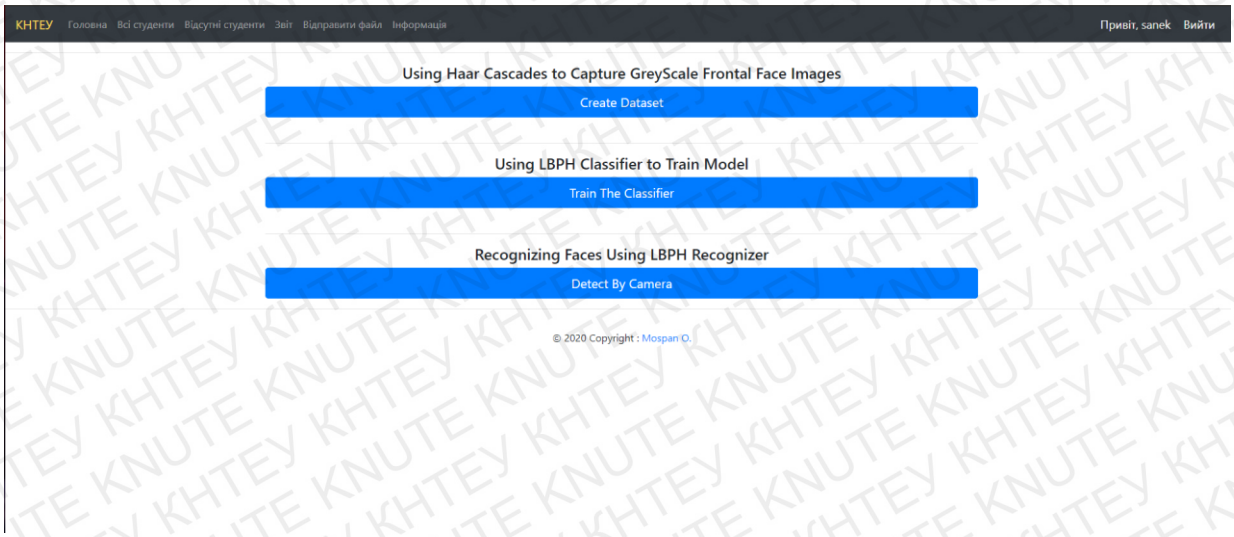

Рис 3.8 Головна сторінка

Сторінка за адресою /all_students/, де можна побачити список всіх студентів(Рис 3.9),

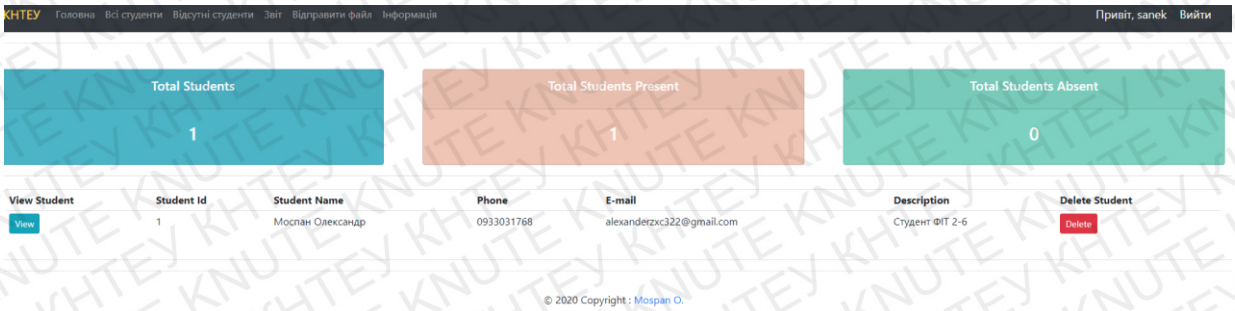

Рис 3.9 Список всіх студентів

Детальна інформація про студента /profile/id/. Де можна знайти дані для зв’язку на персональну статистику відвідувань(Рис 3.10).

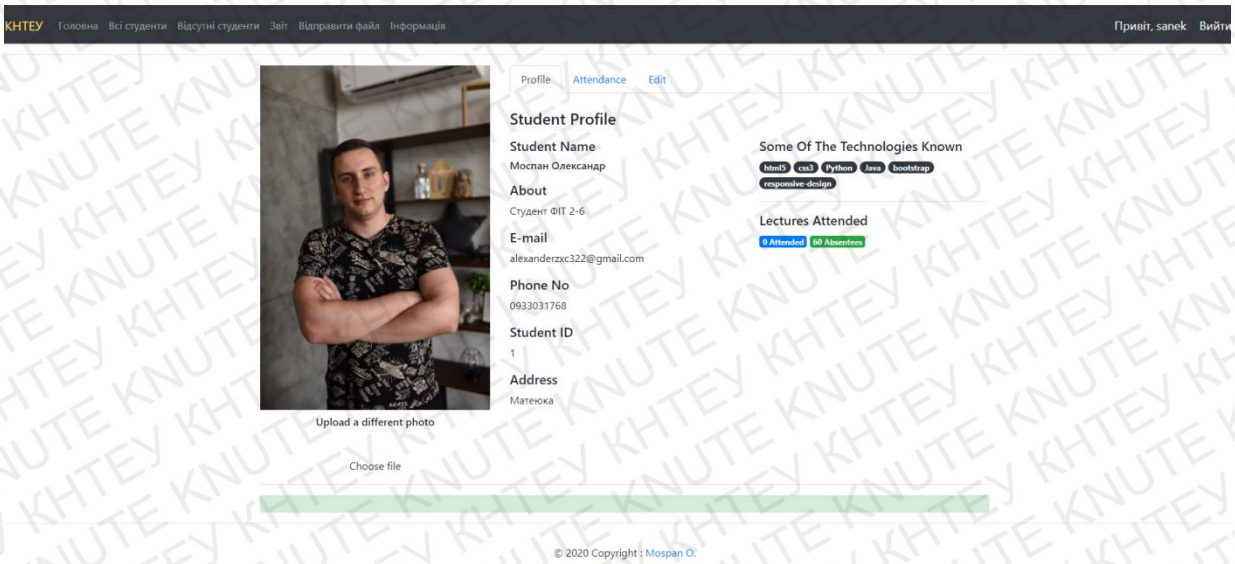

Рис 3.10 особиста сторінка студента

ВИСНОВКИ ТА ПРОПОЗИЦІЇ

З проведених досліджень можна зробити наступні висновки:

1. Проведено аналіз роботи методів біометричної аутентифікації, виявлено їх слабкі місця та переваги. Також було порівняно різні методи аутентифікації та обрано оптимальний.Для реалізації поставлених задач була вибрана клієнт-серверна архітектура. Серверна частина, на відміну від альтернативних рішень, встановлюється локально і не вимагає доступу до Інтернет.

2. Для розробки проекту була обрана мова програмування Python та в якості Back-End частини - Web Framework Django.

3. Проведений аналіз засобів розробки показав, що найзручнішим для реалізації поставлених завдань є IDE Pycharm.

4. В процесі виконання магістерської роботи розроблено систему біометричної аутентифікації. В якій реалізовано функції додавання студентів до бази, їх аутентифікації та ведення статистики відвідувань.

Розроблена система повністю задовольняє поставлені функціональні вимоги. Тестування продемонструвало повну відповідність функціональним вимогам та надійну роботу за стосунку.

					КНТЕУ 121– 06-15.МР			
					Розробка біометричної системи аутентифікації обліку відвідування студентів	Стадія	Аркуш	Аркушів
Зм.	Аркуш	№ докум.	Підпис	Дата		ВП	33	36
Зав. каф.		Криворучко О.В.						
Керівник		Пашорін В.І.						
Гарант		Криворучко О.В.			Висновки та пропозиції	Факультет інформаційних технологій, 2 курс, 6 група		
Розробив		Моспан О.В.						

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Олексіївський П.І. Навчання програмуванню студентів на основі методології уніфікованого процесу розробки програмного забезпечення // Педагогічна освіта в Росії. 2014. №8. С. 151.
2. Al Sweigart Automate the Boring Stuff with Python: Practical Programming formTotal Beginners. - 1st Edition No Starch Press, 2015
3. Eric Matthes Python Crash Course: A Hands-On, Project-Based Introduction to Programming. - 1st edition No Starch Press, 2015

Інтернет ресурси:

4. Документація Django [Електронний ресурс] – Режим доступу: https://docs.django.io/django/docs/current/django-web-framework.html
5. Система управління версіями [Електронний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/Система_управления_версиями
6. Документація Hibernate [Електронний ресурс] – Режим доступу: https://hibernate.org/orm/documentation/5.4/
7. Learn Version Control with Git [Електронний ресурс] – Режим доступу: https://www.gittower.com/learn/git/ebook/en/command-line/introduction
8. Spring Security Taglibs [Електронний ресурс] – Режим доступу: https://www.baeldung.com/spring-security-tagli

ТЕХНІЧНЕ ЗАВДАННЯ

Однією з найголовніших запорук відповідності вимогам та високої якості розробленого програмного рішення є грамотно спроектоване до початку розробки технічне завдання. Отже, далі розкриємо невід'ємні положення технічного завдання до розробки біометричної системи аутентифікації.

1. Загальні відомості

Назва проекту: FaceRecognitionSystem.

Планові строки розробки: серпень 2019 – вересень 2019.

Потенційні користувачі: користувачі стаціонарних ПК з встановленою ОС Windows.

Призначення програмного рішення: надати можливість користувачу ідентифікувати студента, та відмітити його в електронному журналі

Мета створення програмного рішення: опанування сучасних технологій розробки веб-додатків та їх закріплення на практиці.

2. Структура проекту

Оскільки процес управління передбачає наявність суб'єкта управління (смартфон) і об'єкта управління (курсор миші на екрані ПК), програмне рішення складатиметься з двох частин: серверної і клієнтської.

Серверна частина виконуватиметься на ПК, оскільки він надає доступ до своєї системи.

Клієнтська частина буде реалізована у вигляді веб-додатку, що виконуватиметься на ПК.

3. Функціональні вимоги до додатку

					КНТЕУ 121– 06-15.МР			
					Розробка біометричної системи аутентифікації обліку відвідування студентів	Стадія	Аркуш	Аркушів
Зм.	Аркуш	№ докум.	Підпис	Дата		ТЗ	35	36
Зав. каф.		Криворучко О.В.		19.10.20				
Керівник		Пашорін В.І.		19.10.20				
Гарант		Криворучко О.В.		19.10.20	Технічне завдання	Факультет інформаційних технологій, 2 курс, 6з група		
Розробив		Моспан О.В.		19.10.20				

Перелік функціональних можливостей визначений базовими діями, що виконує система:

- Реєстрація нового користувачами
- Створення знімків обличчя для аутентифікації
- Навчання системи для виявлення потрібного користувача
- Аутентифікація студента та внесення його до електронного журналу
- Створення загальної статистики відвідувань

ДОДАТКИ

ДОДАТОК А

Код файлу views.py додатку face_rec

```
from django.shortcuts import render,redirect
from django.contrib.auth import authenticate,login,logout
from django.contrib import messages
from .forms import CreateUserForm
from django.contrib.auth.decorators import login_required
from face_recognition_system.settings import BASE_DIR
from .models import student_profile,student_attendance
from django.views import View
from django.core.mail import EmailMessage
from django.conf import settings
from .decorators import unauthenticated_user,allowed_users,admin_only
import os
import cv2
from PIL import Image
import numpy as np
import pandas as pd
import time
import datetime
import csv

@login_required(login_url='login')
@allowed_users(allowed_roles=['teacher','admin'])
def create_dataset(request):
Id=request.POST['userId']
name=request.POST['userId1']
cam = cv2.VideoCapture(0)
```

Продовження дод. А

```
detector = cv2.CascadeClassifier(BASE_DIR+'/algorithms/haarcascade_frontalface_default.xml')
sampleNum=0
while(True):
ret, img = cam.read()
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
faces = detector.detectMultiScale(gray, 1.3, 5)
for (x,y,w,h) in faces:
cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
#incrementing sample number
sampleNum=sampleNum+1
#saving the captured face in the dataset folder TrainingImage
cv2.imwrite(BASE_DIR+"/static/images/TrainingImage/"+name +"."+Id +'.'+ str(sampleNum) + ".jpg", gray[y:y+h,x:x+w])
#display the frame
cv2.imshow('frame',img)
#wait for 100 miliseconds
if cv2.waitKey(100) & 0xFF == ord('q'):
break
# break if the sample number is morethan 100
elif sampleNum>60:
break
cam.release()
cv2.destroyAllWindows()
# res = "Images Saved for ID : " + Id +" Name : "+ name
row = [Id , name]
with open('StudentDetails/StudentDetails.csv','a+') as csvFile:
```

Продовження дод. А

```
writer = csv.writer(csvFile)
writer.writerow(row)
csvFile.close()
return redirect('home')

@allowed_users(allowed_roles=['teacher','admin'])
@login_required(login_url='login')
def trainer(request):
recognizer = cv2.face.LBPHFaceRecognizer_create()
detector =
cv2.CascadeClassifier(BASE_DIR+'/algorithms/haarcascade_frontalface_default.xml')
faces,Id = getImagesAndLabels(BASE_DIR+"/static/images/TrainingImage/")
recognizer.train(faces, np.array(Id))
recognizer.save(BASE_DIR+"/algorithms/TrainingImageLabel/Trainner.yml")
cv2.destroyAllWindows()
return redirect('home')

def getImagesAndLabels(path):
#get the path of all the files in the folder
imagePaths=[os.path.join(path,f) for f in os.listdir(path)]
#print(imagePaths)

#create empth face list
faces=[]
#create empty ID list
Ids=[]
```

#now looping through all the image paths and loading the Ids and the images
for imagePath in imagePaths:

Продовження дод. А

#loading the image and converting it to gray scale
pilImage=Image.open(imagePath).convert('L')
#Now we are converting the PIL image into numpy array
imageNp=np.array(pilImage,'uint8')
#getting the Id from the image
Id=int(os.path.split(imagePath)[-1].split(".")[1])
# extract the face from the training image sample
faces.append(imageNp)
Ids.append(Id)
cv2.imshow("Training", imageNp)
cv2.waitKey(10)
return faces,Ids
@login_required(login_url='login')
@allowed_users(allowed_roles=['teacher','admin'])
def TrackImages(request):
recognizer =
cv2.face.LBPHFaceRecognizer_create()#cv2.createLBPHFaceRecognizer()
recognizer.read(BASE_DIR+'/algorithms/TrainingImageLabel/Trainner.yml')
harcascadePath = "algorithms/haarcascade_frontalface_default.xml"
faceCascade = cv2.CascadeClassifier(harcascadePath);
df=pd.read_csv(BASE_DIR+"/StudentDetails/StudentDetails.csv")
cam = cv2.VideoCapture(0)
font = cv2.FONT_HERSHEY_SIMPLEX
col_names = ['Id','Name','Date','Time']
attendance = pd.DataFrame(columns = col_names)
minW = 0.1*cam.get(3)

Продовження дод. А

```
minH = 0.1*cam.get(4)

while True:
ret, im =cam.read()
gray=cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
faces = faceCascade.detectMultiScale(
gray,
scaleFactor = 1.2,
minNeighbors = 5,
minSize = (int(minW), int(minH)),
)
for(x,y,w,h) in faces:
cv2.rectangle(im,(x,y),(x+w,y+h),(225,0,0),2)
Id, conf = recognizer.predict(gray[y:y+h,x:x+w])
if(conf < 100):
ts = time.time()
date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
aa=df.loc[df['Id'] == Id]['Name'].values
name=("".join(aa))
tt=str(Id)+"-"+aa
attendance.loc[len(attendance)] = [Id,name,date,timeStamp]
student=student_profile.objects.get(student_id=Id)
data=student_attendance.objects.create(roll=student,name=name,date=date,time=tim
eStamp)
data.save()
```

```
user=student_profile.objects.filter(student_id=Id).update(attendance='Present')
if user:
```

Продовження дод. А

```
messages.success(request, 'Attendance Saved For ' + str(Id))

cam.release()
cv2.destroyAllWindows()
return redirect('/profile/'+ str(Id))
else:
Id='Unknown'
tt=str(Id)
if(conf > 75):
noOfFile=len(BASE_DIR+("ImagesUnknown"))+1
cv2.imwrite("ImagesUnknown/Image"+str(noOfFile) + ".jpg", im[y:y+h,x:x+w])
cv2.putText(im,str(tt),(x,y+h), font, 1,(255,255,255),2)
attendance=attendance.drop_duplicates(subset=['Id'],keep='first')
cv2.imshow('im',im)
if (cv2.waitKey(1)==ord('q')):
break
ts = time.time()
date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
Hour,Minute,Second=timeStamp.split(":")
fileName="Attendance/Attendance_"+date+"_"+Hour+"-"+Minute+"-
"+Second+".csv"
attendance.to_csv(fileName,index=False)
cam.release()
```

```
cv2.destroyAllWindows()
return redirect('home')
```

Продовження дод. А

```
def addStudent(request):
if request.method=='POST':

id = request.POST.get('rollno', '')
name = request.POST.get('username','')
address=request.POST.get('address', '')
mob= request.POST.get('mobileno', '')
email=request.POST.get('email', '')
desc=request.POST.get('desc','')
user=student_profile(student_id=id,name=name,address=address,phone=mob,email
=email,description=desc)
user.save()
if user:
messages.success(request, 'Account was created sfor ' + name)
return redirect('home')
else:
return messages.success(request,'Internal Server Error')
return render(request,'face_rec/add_student.html')
return render(request, 'face_rec/add_student.html')

def registerPage(request):
form=CreateUserForm()
if request.method=='POST':
form=CreateUserForm(request.POST)
if form.is_valid():
user=form.save()
```

username=form.cleaned_data.get('username')
messages.success(request,'Account Was Created For '+username)

Продовження дод. А

return redirect('login')
context={'form':form}
return render(request,'face_rec/add_teacher.html',context)

def loginPage(request):if request.method=='POST':
username=request.POST.get('username','')
password=request.POST.get('password','')
user=authenticate(request,username=username,password=password)
if user is not None:
login(request,user)
return redirect('home')
else:
messages.info(request,'Username Or Password is Incorrect')
context={}
return render(request,'face_rec/login.html',context)

@login_required(login_url='login')
@allowed_users(allowed_roles=['teacher','admin'])
def deleteStudent(request,pk):
student=student_profile.objects.get(student_id=pk)
if request.method=='POST':
student.delete()
return redirect('/')
context={'student':student}
return render(request,'face_rec/delete.html',context)

```
# @unauthenticated_user
```

Продовження дод. А

```
# @login_required(login_url='login')
def home(request):
return render(request,'face_rec/dashboard.html')

def logoutUser(request):
logout(request)
return redirect('login')

@login_required(login_url='login')
@allowed_users(allowed_roles=['teacher','admin','student'])
def profile(request,pk):
id = int(pk)
request.FILES
if request.method=='POST':
id = int(pk)
name = request.POST.get('username','')
address=request.POST.get('address', '')
mob= request.POST.get('mobileno', '')
email=request.POST.get('email', '')
desc=request.POST.get('desc','')
user=student_profile(student_id=id,name=name,address=address,phone=mob,email
=email,description=desc)
user.save()

if user:
messages.success(request, 'Account was Updated For ' + name)
```

```
else:return messages.success(request,'Internal Server Error')
return render(request,'face_rec/student_profile.html')
```

Продовження дод. А

```
student=student_profile.objects.filter(student_id=id)
attendance=student_attendance.objects.filter(roll=id)
present=student_attendance.objects.filter(roll=pk).count()
absent=60-present
context={'student':student,'attendance':attendance,'present':present,'absent':absent}
return render(request,'face_rec/student_profile.html',context)


@login_required(login_url='login')
@allowed_users(allowed_roles=['teacher','admin'])
def all_students(request):
student=student_profile.objects.all()
total_students_absent=student_profile.objects.filter(attendance='Absent').count()
total_students_present=student_profile.objects.filter(attendance='Present').count()
total_students=student_profile.objects.count()
context={'student':student,'total_students':total_students,'total_students_present':total
_students_present,'total_students_absent':total_students_absent}
return render(request,'face_rec/total.html',context)

def about(request):
return render(request,'face_rec/about.html')

@login_required(login_url='login')
@allowed_users(allowed_roles=['teacher','admin'])
def absent_students(request):
student=student_profile.objects.filter(attendance='Absent').all()
```

```
total_students_absent=student_profile.objects.filter(attendance='Absent').count()
```

Продовження дод. А

```
total_students_present=student_profile.objects.filter(attendance='Present').count()
total_students=student_profile.objects.count()
context={'student':student,'total_students':total_students,'total_students_present':total
_students_present,'total_students_absent':total_students_absent}
return render(request,'face_rec/total.html',context)


@login_required(login_url='login')
@allowed_users(allowed_roles=['teacher','admin'])
def report(request):
student=student_profile.objects.all()
total_students_present=student_profile.objects.filter(attendance='Present').count()
total_students=student_profile.objects.count()
ts = time.time()
date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%M-%d')
timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')

context={'student':student,'total_students':total_students,'total_students_present':total
_students_present,'date':date,'time':timeStamp}
return render(request,'face_rec/report.html',context)

@login_required(login_url='login')@allowed_users(allowed_roles=['teacher','admin'
])
def send_file(request):
return render(request,'face_rec/file.html')
@login_required(login_url='login')
```

```
@allowed_users(allowed_roles=['teacher','admin'])
def send(request):
```

Продовження дод. А

```
if request.method=='POST':
request.POST, request.FILES
subject = request.POST.get('title')
message = request.POST.get('subject')
email = request.POST.get('email')
files = request.FILES.getlist('file')
try:
mail = EmailMessage(subject, message, settings.EMAIL_HOST_USER, [email])
for f in files:
mail.attach(f.name, f.read(), f.content_type)
mail.send()
# messages.success(request,"File Sent To " + email)
return render(request,'face_rec/file.html',{'error_message': 'Sent email to %s'%email})
except:
return render(request,'face_rec/file.html',{'error_message': 'Either the attachment is too big or corrupt'})
return render(request,'face_rec/file.html',{'error_message': 'Unable to send email. Please try again later'})
```

Код файлу models.py

```
from django.db import models
from django.contrib.auth.models import User

class student_profile(models.Model):
user=models.OneToOneField(User,null=True,blank=True,on_delete=models.CASC
ADE)
student_id = models.CharField(primary_key=True,max_length=20)
name = models.CharField(max_length=50)
address = models.CharField(max_length = 100)
phone = models.CharField(max_length =  50)
email = models.CharField(max_length = 50)
description = models.CharField(max_length = 200)
mark = (
('Absent', 'Absent'),
('Present', 'Present'),
)
attendance = models.CharField(max_length=20,default='Absent', choices=mark)
profile_pic=models.ImageField(default="static/images/nobody.jpg")

def __str__(self):
return self.student_id

class student_attendance(models.Model):
roll=models.ForeignKey(student_profile, null=True, on_delete=models.SET_NULL)
name = models.CharField(max_length=50)
date=models.DateField(max_length=50)
time=models.TimeField(max_length=50)
```

```
def __str__(self):
return self.name
```

Код файлу decorators.py

```
from django.http import HttpResponse
from django.shortcuts import redirect

def unauthenticated_user(view_func):
    def wrapper_func(request,*args,**kwargs):
        if request.user.is_authenticated:
            return redirect('home')
        else:
            return view_func(request,*args,**kwargs)
    return redirect('home')


def allowed_users(allowed_roles=['student']):
    def decorator(view_func):
        def wrapper_func(request,*args,**kwargs):
            group=None
            if request.user.groups.exists():
                group=request.user.groups.all()[0].name
            if group in allowed_roles:
                return view_func(request,*args,**kwargs)
            else:
                return HttpResponse('You are Not Authorized To View This
Page')
        return wrapper_func
    return decorator
```

Код файлу dashboard.html

```
{% extends 'face_rec/main.html' %}
{% block content %}
<body>
    <div class="container">
<form class="form-signin" action="/create_dataset" method="post">{% csrf_token
%}
<h4 class="text-center">Using Haar Cascades to Capture GreyScale Frontal Face
Images</h4>
<button class="btn btn-lg btn-primary btn-block" type="button" data-
toggle="modal" data-target="#myModal">Create Dataset</button>
<div class="modal fade" id="myModal" tabindex="-1" role="dialog" aria-
labelledby="myModalLabel">
<div class="modal-dialog" role="document">
<div class="modal-content">
<div class="modal-header">
<button type="button" class="close" data-dismiss="modal" aria-
label="Close"><span aria-hidden="true">&times;</span></button>
<h4 class="modal-title" id="myModalLabel">Input unique id and wait for
webcam</h4>
</div>
<div class="modal-body">
<ul>
<li>For higher accuracy, make sure to stay in room with good lightning.</li>
<li>Please remove any type of accesories from face.</li>
<li>Make sure your face in clearly visible in camera.</li>
<li>Make sure your hair is not blocking the face.</li>
<li>Make sure no other faces are within the cam frame.</li>
</ul>
<hr/>
<form action="#" method="POST">
                        {% csrf_token %}
<div class="form-group">
<label for="inputId">Enter Unique ID*</label>
<input type="text" class="form-control" name="userId" id="inputId"
placeholder="Enter Id">
</div>
<div class="form-group">
<label for="inputId1">Enter Unique Name*</label>
<input type="text" class="form-control" name="userId1" id="inputId"
```

```
placeholder="Enter Name">
</div>
<button type="submit" class="btn btn-warning">Submit</button>
</form>
</div>
</div>
</div>
</div>
</form>
<br><hr>

<form class="form-signin" action="/trainer" method="post">{% csrf_token %}
<h4 class="text-center">Using LBPH Classifier to Train Model</h4>
<button class="btn btn-lg btn-primary btn-block" type="submit">Train The
Classifier</button>
</form><br><hr>
<form class="form-signin" action="/detect" method="post">{% csrf_token %}
<h4 class="text-center">Recognizing Faces Using LBPH Recognizer</h4>
<button class="btn btn-lg btn-primary btn-block" type="submit">Detect By
Camera</button>
        {% for message in messages %}
<p class="message">{{message}}</p>
        {% endfor %}
</form>
</div>
</body>
{% endblock %}
```