Кіровоградська Мала академія наук учнівської молоді

СИСТЕМА МОНІТОРИНГУ РОБОЧОЇ АКТИВНОСТІ ПРАЦІВНИКІВ

Манько Ілля Романович, учень 11 класу комунального закладу «Ліцей «Максимум» Кропивницької міської ради»
Наукові керівники: Мироненко Віктор Анатолійович, вчитель інформатики комунального закладу «Центральноукраїнський науковий ліцей – інтернат Кіровоградської обласної ради»
Шарун Павло Сергійович, вчитель фізики комунального закладу «Ліцей «Максимум» Кропивницької міської ради»

Рис. 1. Код програми (код автора)

Об'єкт дослідження:
процес моніторингу робочої активності працівників .

Предмет дослідження:
методи та програмно–апаратні засоби для автоматизації моніторингу робочої активності.

Методи дослідження:
Аналіз
Структурний підхід
Моделювання
Формалізація

Мета дослідження:
розробка системи моніторингу робочої активності працівників.

МАТЕРІАЛИ ТА ХІД ДОСЛІДЖЕННЯ:

Проаналізувавши програми–аналоги помітили, що жодна не фіксує час присутності на робочому місті автоматично. Відповідно, вирішили створити систему, яка за допомогою веб–камери визначає наявність працівника на робочому місці та веде облік відсутності. Додатково реалізовані супутні засоби перевірки: перевірка історії браузера та відкритих програм. Архітектура системи: клієнт–серверна з доступом до статистичних даних через веб–інтерфейс (Рис.2).

ІНСТРУМЕНТИ РОЗРОБКИ ТА БІБЛІОТЕКИ:

- Мова програмування python.
- Бібліотека cv2 для взаємодії з веб–камерою.
- Бібліотека mediapipe для обробки зображення.
- Фреймворк django для обробки запитів на сервері, взаємодії з базою даних та створення веб–інтерфейсу
- База даних MySQL (Рис.2).

Завдання дослідження:

1. Дослідити сучасні підходи до моніторингу робочої активності працівників.
2. Розробити програму, яка ідентифікує працівника за допомогою веб–камери, фіксує його присутність на робочому місці та аналізує використання комп'ютера.
3. Реалізувати клієнт–серверну архітектуру застосунку.
4.. Оцінити ефективність впровадження розробленої системи.

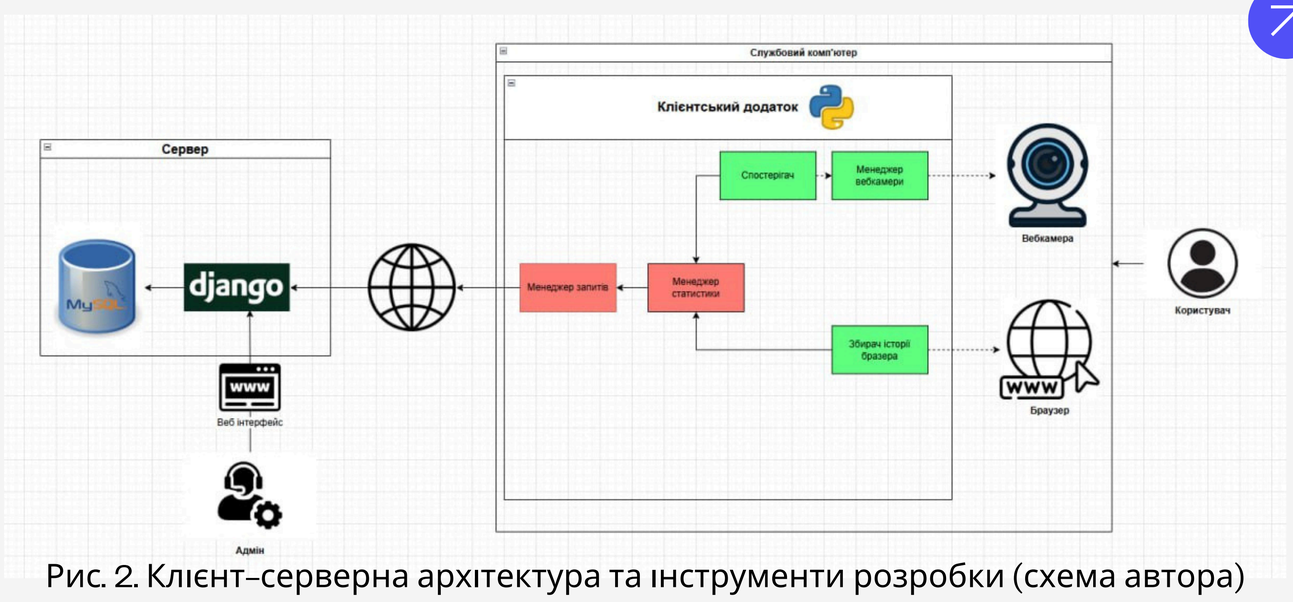

Рис. 2. Клієнт–серверна архітектура та інструменти розробки (схема автора)

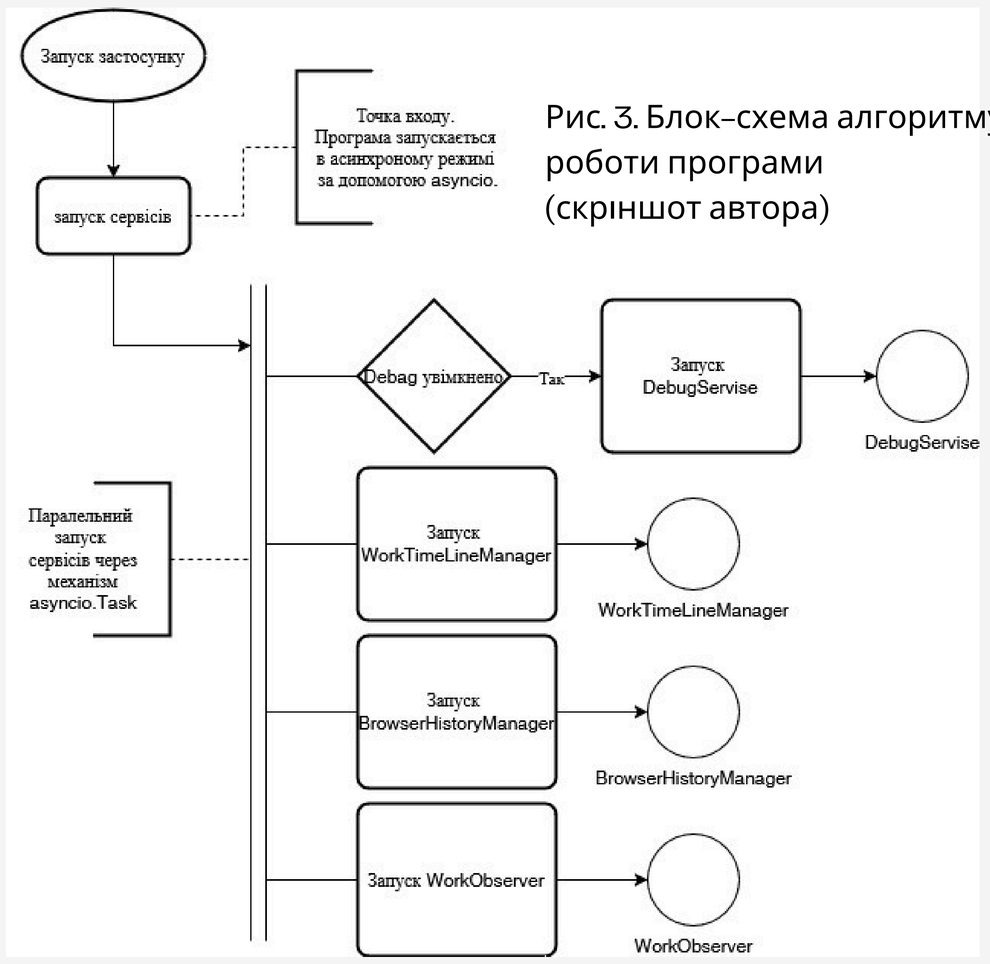

Рис. 3. Блок–схема алгоритму роботи програми (скріншот автора)

```
def auth_decorator(func):
  def wrapper(request, *arg, **kwargs):
    auth_header = request.headers.get('Auth')
    if not auth_header:
      return JsonResponse({'error': 'Authorization token is missing'}, status=401)
    try:
      decoded_token = jwt.decode(auth_header, JWT_KEY, algorithms=["HS256"])
      request.user = decoded_token
    except (IndexError, jwt.ExpiredSignatureError, jwt.InvalidTokenError):
      return JsonResponse({'error': 'Invalid or expired token'}, status=401)
    return func(request, *arg, **kwargs)
  return wrapper
```

Рис. 4. Фрагмент коду перевірки валідності токена (код автора)

На рис. 3 наведено блок–схему алгоритму роботи основної програми, де відображенно асинхронність роботи сервісів через механізм asyncio.Task.
На рис. 4 наведено фрагмент коду в якому RequestsManager при ініціалізації запитує у DBManager токен. На сервері є декоратор який перевіряє валідність токену та розпаковує його вміст передаючи його далі у функцію.

Результати та висновки:

1. Проведений порівняльний аналіз сприяв визначенню недоліків існуючих моніторингових систем та показав на що слід звернути увагу..
2. Вдалий вибір інструментів розробки дозволив розробити працюючу систему моніторингу, перевагою якої стала автоматична фіксація часу початку та кінця активності.
3. Розроблена клієнт–серверна архітектура забезпечила системі баланс між продуктивністю, безпекою та гнучкістю.
4. Проведені тестування довели ефективність роботи системи моніторингу. Вірність розпізнавання облич становила 90%, визначення часових проміжків – 95%.