

ДОДАТОК А

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ

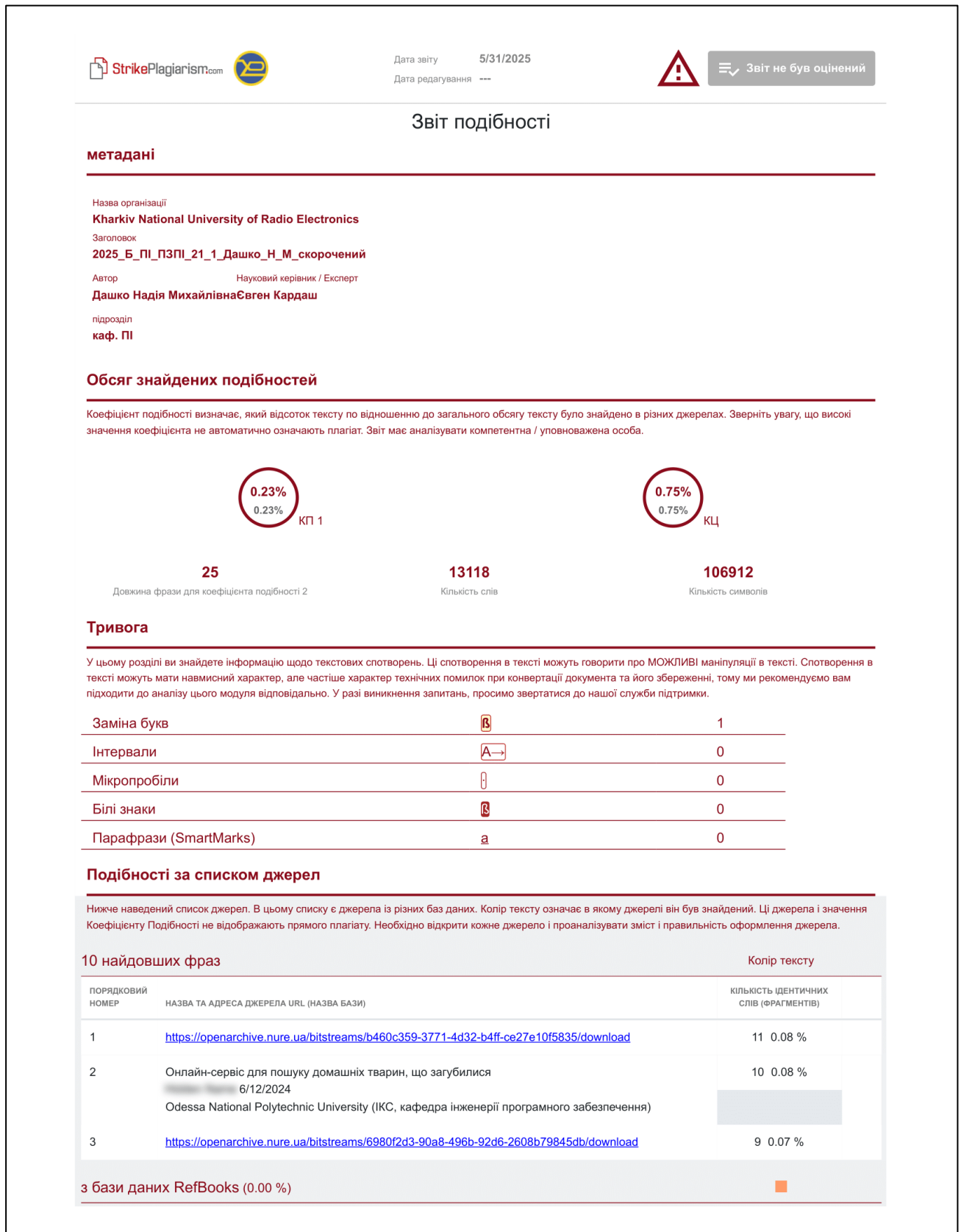



Рисунок А.1 – Результат перевірки на унікальність тексту

ДОДАТОК Б
Слайди презентації

Міністерство освіти і науки України Харківський національний університет радіоелектроніки

Кваліфікаційна робота бакалавра




**Програмна система для планування та моніторингу виконання особистих задач і досягнень.
Клієнтська частина**

Виконала:
ст. гр. ПЗПІ-21-1
Дашко Н. М.

Науковий керівник:
проф. каф. ПІ
Дудар З. В.

1

Рисунок Б.1 – Слайд 1



Мета роботи

Розробити клієнтську частину вебзастосунку для управління особистими задачами, що поєднує елементи гейміфікації, аналітики та соціальної взаємодії.

Основні завдання:

- Проаналізувати предметну область;
- Створити дизайн за принципами UI/UX;
- Обґрунтувати вибір платформи, мови та бібліотек для реалізації;
- Спроектувати архітектуру та структуру клієнтської частини;
- Реалізувати інтерфейс;
- Забезпечити інтеграцію з API та зовнішніми сервісами;
- Забезпечити якість і стабільність коду.

2

Рисунок Б.2 – Слайд 2

План презентації


- Мета роботи
- Аналіз предметної області
- UI/UX-дизайн
- Вибір платформи та технологій
- Архітектура та структура проєкту
- Побудова інтерфейсу
- Взаємодія з API
- Висновки

3

Рисунок Б.3 – Слайд 3

Аналіз предметної області. Habitica

Характеристика	Habitica	Розроблена система
Основна мета	Колективна мотивація, гейміфікація	Персональне управління завданнями
Соціальна взаємодія	Групові виклики, але без персональних зв'язків	Друзі, перегляд профілю, запити на додавання
Гейміфікація	Сильна: інвентар, еліксири, кастомізація персонажа	Обмежена: досягнення, подарунки, візуальні відзнаки
Візуальний стиль	Стилізований під гру з персонажами	Мінімалістичний, зручний для повсякденного користування
Орієнтоване на	Командна діяльність	Індивідуальні цілі та адаптивність під користувача

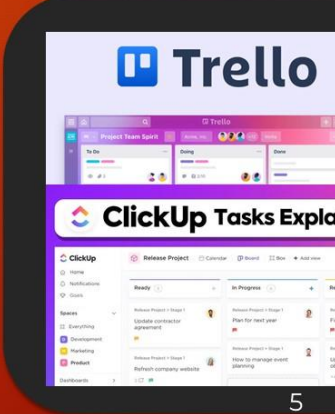


4

Рисунок Б.4 – Слайд 4

Аналіз предметної області. Trello, ClickUp

Характеристика	Trello	ClickUp
Основна мета	Керування проєктами	Комплексний менеджмент
Соціальна взаємодія	Колаборація в межах дошок	Командна співпраця
Гейміфікація	Відсутня	Відсутня
Візуальний стиль	Канбан-дошка	Навантажений, професійний
Орієнтоване на	Команди, проєктні групи	Корпоративне середовище



5

Рисунок Б.5 – Слайд 5

UI/UX-дизайн

Реєстрація

Ім'я

Прізвище

Пошта

Пароль

Підтвердь пароль

Стать Жінка Чоловік

Дата народження

І на останок! Як ти хочеш, щоб до тебе звертались однодумці?

[Уже з нами? Увійти](#)

→

Реєстрація

Ім'я

Прізвище

Пошта

Пароль

Підтвердь пароль

Стать Жінка Чоловік

Дата народження

І на останок! Як ти хочеш, щоб до тебе звертались однодумці?

[Уже з нами? Увійти](#)

6

Рисунок Б.6 – Слайд 6

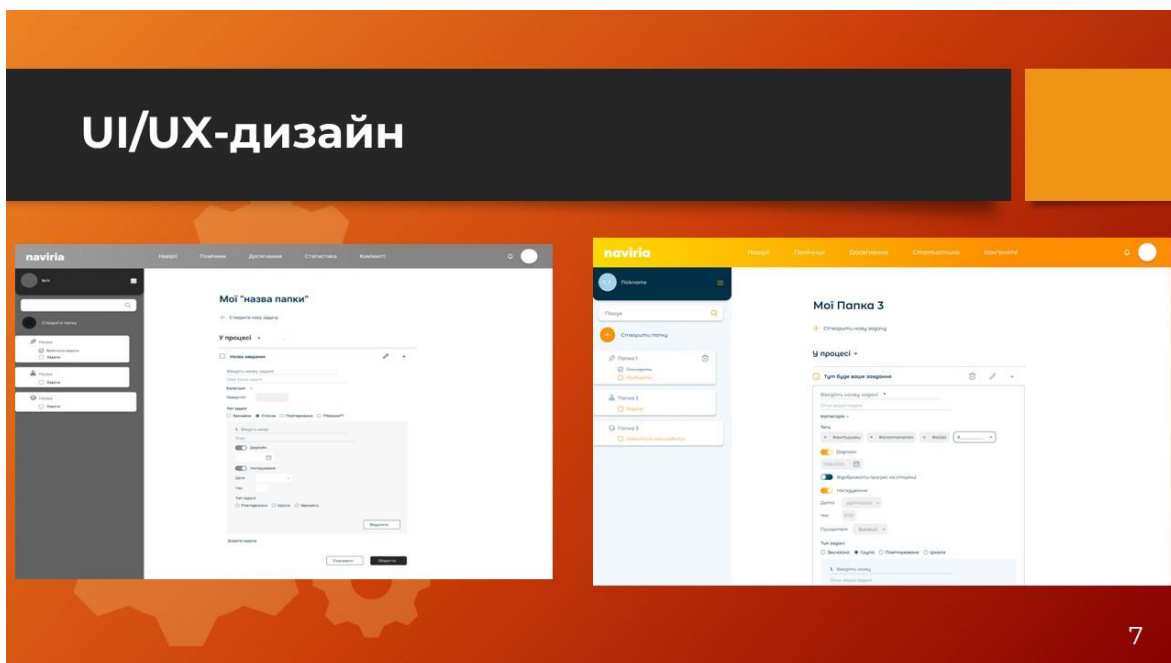


Рисунок Б.7 – Слайд 7

Вибір платформи та технологій

Обрана платформа: Веббраузер

- Універсальний доступ — з будь-якого пристрою з інтернетом
- Не потребує встановлення — достатньо сучасного браузера
- Зручне оновлення функціоналу в реальному часі
- Простота у розгортанні та масштабуванні застосунку

Переваги для користувача

- Миттєвий доступ
- Підтримка адаптивного інтерфейсу
- Незалежність від операційної системи

8

Рисунок Б.8 – Слайд 8

Вибір мови програмування

Обрана мова: JavaScript

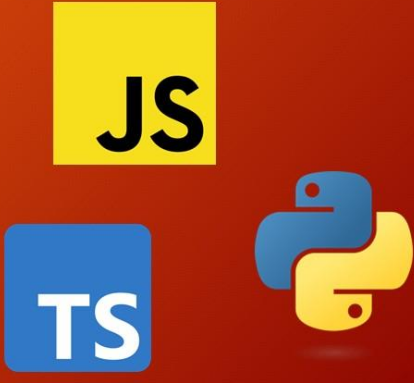
- Основна мова клієнтської частини вебзастосунків
- Підтримка всіма браузерами без додаткових модулів
- Величезна екосистема бібліотек (React, ESLint, Recharts тощо)

Переваги використання

- Гнучкість та інтерактивність інтерфейсу
- Активна спільнота розробників
- Простота інтеграції з API та зовнішніми сервісами

Порівняння

- TypeScript — складніший для початківців
- Python, Java — не підтримуються напряму в браузері



9

Рисунок Б.9 – Слайд 9

Вибір основної бібліотеки. React

Компонентний підхід

- Уся логіка інтерфейсу організована в ізольовані компоненти;
- Спрощує повторне використання коду та тестування;

Одностороннє управління даними

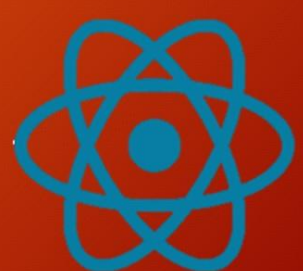
- Зменшує кількість помилок при зміні стану;
- Забезпечує передбачувану поведінку застосунку;

JSX-синтаксис

- Поєднання HTML та JavaScript в одному файлі;
- Полегшує розробку та читабельність коду;

Екосистема та підтримка

- Активна спільнота, велика кількість готових рішень;
- Сумісність із сучасними інструментами (Vite, ESLint, Recharts).



10

Рисунок Б.10 – Слайд 10

Інші технології та інструменти

Vite

- Сучасний інструмент для збірки фронтенду
- Швидкий старт і гаряче перезавантаження (Hot Module Replacement)
- Підтримка сучасного синтаксису та модульності

ESLint





- Аналіз якості коду
- Виявлення помилок на етапі розробки
- Підтримка єдиного стилю написання коду

Recharts

- Побудова діаграм і графіків
- Інтеграція з React, гнучка візуалізація статистики

@react-oauth/google

- Безпечна автентифікація через Google
- Швидка інтеграція з OAuth 2.0
- Зберігання та обробка токенів відповідно до стандартів

11

Рисунок Б.11 – Слайд 11

Архітектура та структура проєкту

Загальна архітектура:

- Односторінковий застосунок (SPA) на основі React
- Побудований за компонентним принципом із чітким поділом логіки, візуалізації та обміну даними
- Підтримує маршрутизацію, взаємодію з API, адаптивну верстку та модульну структуру

Ключові принципи реалізації:

- **SRP (Single Responsibility Principle):** кожен модуль має єдину відповідальність
- **Ізоляція API-запитів:** бізнес-логіка не змішується з UI
- **Адаптивна верстка:** використання flex, media-запитів та окремих стилів для елементів

12

Рисунок Б.12 – Слайд 12

Архітектура та структура проєкту

Папка / Файл	Призначення
views/	Сторінки інтерфейсу: профіль, задачі, статистика, ком'юніті тощо
services/	API-сервіси: усі HTTP-запити винесені у функціональні модулі
hooks/	Кастомні React-хуки для повторної бізнес-логіки
styles/	CSS- та LESS-файли зі стилями, організовані за компонентами
App.jsx / main.jsx	Ініціалізація застосунку, маршрутизація, підключення глобальних контекстів

```

  Website
  ├── node_modules
  ├── public
  └── src
      ├── assets
      ├── hooks
      ├── services
      ├── styles
      ├── views
      ├── App.jsx
      ├── index.css
      ├── main.jsx
      ├── .gitignore
      ├── baseURLEnv
      ├── eslint.config.js
      ├── index.html
      ├── package-lock.json
      └── package.json
  
```

13

Рисунок Б.13 – Слайд 13

Побудова інтерфейсу

```

const handleSubmit = async (e) => {
  e.preventDefault();
  if (!validate()) return;
  try {
    await registration(
      name + " " + surname,
      nickname,
      gender,
      birthdate,
      email,
      password
    );
  } catch (err) {
    if (
      err.message.includes("user already exists") ||
      err.message.includes("409")
    ) {
      setErrors((prev) => ({
        ...prev,
        email: "Користувач з такою поштою вже існує",
      }));
    } else {
      console.error("Помилка реєстрації:", err);
    }
  }
};
  
```

```

const validate = () => {
  const newErrors = {};
  if (!name.match(/^[a-zA-Zа-яёіїіііііііе\ś'-]{1,20}$/))
    newErrors.name = "Ім'я введено некоректно";

  return (
    <div className="registration-page">
      <div className="horizontal">
        <h1>Реєстрація</h1>
        <div className="registration-form">
          <div className="column1">
            <label>
              Ім'я
              <input
                type="text" ...
                onChange={(e) => setName(e.target.value)}
              />
              {errors.name && <span className="error-text">{errors.name}</span>}
            </label>
          
```

14

Рисунок Б.14 – Слайд 14

Побудова інтерфейсу

```

.registration-page {
  background: linear-gradient(288deg, #ff8700 11.5%, #ffb703 100%);
  height: 100vh;
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  padding: 10vw;
}
.horizontal {
  display: flex;
  flex-direction: column;
  justify-content: start;
  align-items: start;
  background: white;
  border-radius: 25px;
  margin: auto;
  padding: 5vh 5vw;
}
h1 { ...
}
.registration-form {
  display: flex;
  flex-direction: row;
  justify-content: space-between;
}

```

```

.registration-page {
  background: linear-gradient(288deg, #ff8700 11.5%, #ffb703 100%);
  height: 100vh;
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  padding: 10vw;
}
.registration-page .horizontal {
  display: flex;
  flex-direction: column;
  justify-content: start;
  align-items: start;
  background: white;
  border-radius: 25px;
  margin: auto;
  padding: 5vh 5vw;
}
.registration-page .horizontal h1 {
  font-weight: 600;
  margin-bottom: 1.7rem;
}
.registration-page .horizontal .registration-form {
  display: flex;
  flex-direction: row;
}

```

15

Рисунок Б.15 – Слайд 15

Взаємодія з API

```

const API_URL = "http://localhost:5186";

export async function login(email, password) {
  const res = await fetch(`${API_URL}/api/Auth/login`, {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
    },
    body: JSON.stringify({ email, password }),
  });

  const data = await res.json();
  if (!res.ok) {
    throw new Error(data.message || "Сталася помилка при вході");
  }

  localStorage.setItem("token", data.token);

  const payload = parseJwt(data.token);
  if (payload?.sub) {
    localStorage.setItem("id", payload.sub);
  }

  return data;
}

```

```

export async function fetchCategories() {
  const res = await fetch(`${API_URL}/api/Category`);
  if (!res.ok) throw new Error("Не вдалося отримати категорій");
  return await res.json();
}

```

```

export async function deleteTask(id) {
  const res = await fetch(`${API_URL}/api/Task/${id}`, {
    method: "DELETE",
  });
  if (!res.ok) throw new Error("Не вдалося видалити задачу");
}

```

16

Рисунок Б.16 – Слайд 16

Висновки

Реалізовано клієнтську частину вебзастосунку для персонального планування задач з елементами гейміфікації. Інтерфейс побудовано на основі React, із використанням сучасних бібліотек і адаптивної верстки. Забезпечено інтеграцію з серверною частиною через API та автентифікацію через Google OAuth

Можливості практичного застосування:	Подальший розвиток:
<ul style="list-style-type: none">• Особисте управління задачами, звичками та цілями;• Мотиваційний інструмент для школярів, студентів або працівників;• Використання в освітньому середовищі для підвищення дисципліни та продуктивності.	<ul style="list-style-type: none">• Розширення соціальної складової: чати, спільні задачі, виклики;• Підключення аналітики на основі ШІ;• Додавання інтеграцій з календарями, трекерами активності, Notion, Google Tasks тощо;• Монетизація.

17

Рисунок Б.17 – Слайд 17

ДОДАТОК В

Програмний код

Лістинг В.1 – Код з файлу Registration.jsx

```

<div className="registration-page">
  <div className="horizontal">
    <h1>Реєстрація</h1>
    <div className="registration-form">
      <div className="column1">
        <label>
          Ім'я
          <input
            type="text"
            name="name"
            placeholder="Лайт"
            pattern="^[a-zA-Za-яА-ЯёЁїІіİë€\s'\-]{1,20}$"
            value={name}
            onChange={(e) => setName(e.target.value)}
          />
          {errors.name && <span className="error-
text">{errors.name}</span>}
        </label>
        <label>
          Прізвище
          <input
            type="text"
            name="surname"
            placeholder="Ягами"
            pattern="^[a-zA-Za-яА-ЯёЁїІіİë€\s'\-]{1,20}$"
            value={surname}
            onChange={(e) => setSurname(e.target.value)}
          />
          {errors.surname && (
            <span className="error-text">{errors.surname}</span>
          )}
        </label>
        <label>
          Пошта
          <input
            type="email"
            name="email"
            placeholder="example@gmail.com"
            value={email}
            onChange={(e) => setEmail(e.target.value)}
          />
          {errors.email && (
            <span className="error-text">{errors.email}</span>
          )}
        </label>
        <div className="password">
          <label>
            Пароль
            <input
              type="password"

```

```

        name="password"
        placeholder="*****"
        pattern="^(?=.*[a-z])(?=.*[A-Z])(?=.*\d).+$"
        value={password}
        onChange={ (e) => setPassword(e.target.value) }
      />
    </label>
    <button className="show-hide">
      
    </button>
  </div>
  {errors.password && (
    <span className="error-text">{errors.password}</span>
  )}
  <div className="password">
    <label>
      Підтвердіть пароль
      <input
        type="password"
        name="repeat-password"
        placeholder="*****"
        onChange={ (e) => setPasswordCheck(e.target.value) }
        value={passwordCheck}
      />
    </label>
    <button className="show-hide">
      
    </button>
  </div>
  {errors.passwordCheck && (
    <span className="error-text">{errors.passwordCheck}</span>
  )}
</div>
<div className="column2">
  <div className="gender-select">
    <p>Стать</p>

    <div className="selection">
      <label className="radiobutton">
        <input
          type="radio"
          name="gender"
          value="F"
          onChange={ (e) => setGender(e.target.value) }
          checked={gender === "F"}
        />
        Жінка
      </label>
      <label className="radiobutton">
        <input
          type="radio"
          name="gender"
          value="m"
          onChange={ (e) => setGender(e.target.value) }
          checked={gender === "m"}
        />
        Чоловік
      </label>
    </div>
  </div>

```

```

    </div>
    {errors.gender && (
      <span className="error-text">{errors.gender}</span>
    )}
  </div>

  <label>
    Дата народження
    <input
      name="birth-date"
      type="date"
      onChange={ (e) => setBirthDate(e.target.value) }
      value={birthDate}
    />
    {errors.birthDate && (
      <span className="error-text">{errors.birthDate}</span>
    )}
  </label>

  <label>
    І на останок! Як ти хочеш, щоб до тебе звертались однодумці?
    <input
      className="nickname"
      type="text"
      name="nickname"
      placeholder="sigma-killer3000"
      minLength="3"
      maxLength="20"
      pattern="^[a-zA-Z0-9]+$"
      onChange={ (e) => setNickname(e.target.value) }
      value={nickname}
    />
    {errors.nickname && (
      <span className="error-text">{errors.nickname}</span>
    )}
  </label>
  <div className="registration-submit">
    <input
      className="submit-button"
      type="submit"
      value="Почати!"
      onClick={handleSubmit}
    />
    <Link to="/login">
      Уже з нами? <b>Увійти</b>
    </Link>
  </div>
</div>
</div>
</div>
</div>
</div>

```

Лістинг В.2 – Код з файлу registration.css

```

.registration-page .horizontal .registration-form .column1 label {
  display: flex;

```

```

    flex-direction: column;
    gap: 0.5rem;
}
.registration-page .horizontal .registration-form .column1 label input {
    padding: 0.8rem;
    border: none;
    border-radius: 10px;
    background: rgba(2, 48, 71, 0.1);
}
.registration-page .horizontal .registration-form .column1 .password {
    display: flex;
    align-items: end;
    gap: 0.5rem;
}

```

ЛІСТИНГ В.3 – Код з файлу header.less

```

.site-header {
    position: sticky;
    top: 0;
    display: flex;
    flex-direction: row;
    align-items: center;
    justify-content: space-between;
    z-index: 10;
    padding: 2vh 5vw;
    background: linear-gradient(90deg, #ffd900 0%, #ff8f00 70.11%);
    .logo {
        height: 2rem;
    }
    .links {
        display: flex;
        flex-direction: row;
        gap: 5vw;
        flex-shrink: 0;
        margin-right: 2rem;
        margin-left: 2rem;
        a {
            text-decoration: none;
            color: white;
            transition: 0.2s ease;

            &:hover {
                scale: 1.1;
                font-weight: 500;
            }
        }
    }
    .actions {
        display: flex;
        flex-direction: row;
        align-items: center;
        gap: 1rem;
        .notifications {
            background: none;
            border: none;
        }
    }
}

```

```

.avatar {
  width: 5rem;
  height: 5rem;
  border-radius: 50%;
  object-fit: cover;
}
}
}

```

ЛІСТИНГ В.4 – Код з файлу Header.jsx

```

<header className="site-header">
  
  <nav className="links">
    <Link to="/tasks">Навірїї</Link>
    <Link to="/assistant">Помічник</Link>
    <Link to="/achievements">Досягнення</Link>
    <Link to="/statistics">Статистика</Link>
    <Link to="/friends">Ком'юніті</Link>
  </nav>
  <div className="actions">
    <button className="notifications"
onClick={handleToggleNotifications}>
      <img
        src={hasUnread ? "new-notif.svg" : "bell.svg"}
        alt="notifications"
      />
    </button>
    {showNotifications && (
      <Notifications
        notifications={notifications}
        onMarkRead={fetchNotificationsAgain}
      />
    )}

    <Link to="/profile">
      <img
        className="avatar"
        src={user && user.photo ? user.photo : "Ellipse.svg"}
        alt={user ? user.nickname : "avatar"}
      />
    </Link>
  </div>
</header>

```

ЛІСТИНГ В.5 – Код з файлу Notifications.jsx

```

export default function Notifications({ notifications, onMarkRead }) {
  if (!notifications.length) {
    return (
      <div className="notifications-popup">
        <p className="no-notifications">Немає нових повідомлень</p>
      </div>
    );
  }
}

```

```

const handleMarkAllRead = async () => {
  await markAllNotificationsRead();
  onMarkRead();
};

const formatTime = (isoString) => {
  const date = new Date(isoString);
  return date.toLocaleString("uk-UA", {
    hour: "2-digit",
    minute: "2-digit",
    day: "2-digit",
    month: "2-digit",
    year: "numeric",
  });
};

return (
  <div className="notifications-popup">
    <div className="notifications-header">
      <h3 className="notifications-title">Повідомлення</h3>
      <button className="mark-read-btn" onClick={handleMarkAllRead}>
        Прочитати всі
      </button>
    </div>
    {[...notifications]
      .sort((a, b) => new Date(b.recievedAt) - new Date(a.recievedAt))
      .map((notif) => (
        <div
          key={notif.id}
          className={`notification-item ${notif.isNew ? "unread" :
"read"}`}
        >
          <p className="notification-text">{notif.text}</p>
          <span className="notification-time">
            {formatTime(notif.recievedAt)}
          </span>
        </div>
      )
    )}
  </div>
);
}

```

Лістинг В.6 – Код з файлу notifications.less

```

.notifications-popup {
  font-size: @small_text;
  background: #fff;
  box-shadow: 0 5px 15px rgba(0, 0, 0, 0.2);
  border-radius: 12px;
  padding: 1rem;
  width: 25rem;
  position: absolute;
  top: 60px;
  right: 10px;
  z-index: 100;
  max-height: 500px;
  overflow-y: auto;
}

```

```

.notification-item.read {
  background: #f9f9f9;
  color: #555;
  font-size: @small_text;
}

.notification-item.unread {
  background: #fff8dc;
  font-weight: 600;
}
}

```

ЛІСТИНГ В.7 – Код з файлу Friends.jsx

```

useEffect(() => {
  const fetchData = async () => {
    try {
      if (activeTab === "my") {
        const data = await getUserFriends();
        setFriends(data);
        setFilteredFriends(data);
      } else if (activeTab === "requests") {
        const data = await getFriendRequests();
        setRequests(data);
      } else if (activeTab === "discover") {
        const data = await getDiscoverUsers();
        setDiscover(data);
      }
    } catch (e) {
      console.error("Помилка при завантаженні:", e.message);
    }
  };
  fetchData();
}, [activeTab]);

```

ЛІСТИНГ В.8 – Код з файлу friends.less

```

.tabs {
  display: flex;
  flex-direction: row;
  gap: 1.2rem;
  margin-bottom: 3rem;
  button {
    padding: 1rem 1.5rem;
    border: none;
    border-radius: 7px;
    background: rgba(32, 153, 183, 0.08);
    box-shadow: 0px 3px 6px 0px rgba(0, 0, 0, 0.25);
    transition: all 0.2s ease;
    cursor: pointer;
    &:hover {
      background: rgba(32, 153, 183, 0.3);
    }

    &.active {
      box-shadow: none;
    }
  }
}

```

```

        pointer-events: none;
    }
}
}

```

ЛІСТИНГ В.9 – Код з файлу Subtasks.jsx

```

export function Subtasks(props) {
  const [addValue, setAddValue] = useState("");
  const [adding, setAdding] = useState(false);

  const handleAddValue = async () => {
    if (!addValue || isNaN(addValue)) return;
    setAdding(true);
    try {
      const updatedSubtask = {
        ...props,
        currentValue: (props.currentValue || 0) + Number(addValue),
        subtask_type: getServerSubtaskType(props.type),
      };
      await updateSubtask(props.taskId, props.id, updatedSubtask);
      await props.fetchTasks();
    } catch {
      setError("Не вдалося додати значення");
    }
    setAdding(false);
    setAddValue("");
  };

  const getServerSubtaskType = (type) => {
    switch (type) {
      case "simple":
        return "standard";
      case "repeat":
        return "repeatable";
      case "scale":
        return "scale";
      default:
        return "standard";
    }
  };

  let subtaskType;
  switch (props.type) {
    case "standard":
      subtaskType = "simple";
      break;
    case "repeatable":
      subtaskType = "repeat";
      break;
    case "scale":
      subtaskType = "scale";
      break;
    case "with_subtasks":
      subtaskType = "list";
      break;
    default:
      subtaskType = "simple";
  }
}

```

```

}
const weekDays = [
  { short: "Пн", eng: "Monday" },
  { short: "Вт", eng: "Tuesday" },
  { short: "Ср", eng: "Wednesday" },
  { short: "Чт", eng: "Thursday" },
  { short: "Пт", eng: "Friday" },
  { short: "Сб", eng: "Saturday" },
  { short: "Нд", eng: "Sunday" },
];

const today = new Date();
const weekDayEng =
  weekDays[today.getDay() === 0 ? 6 : today.getDay() - 1].eng;
const todayISO = today.toISOString().split("T")[0];

const canCheckin =
  Array.isArray(props.repeatDays) &&
  props.repeatDays.includes(weekDayEng);
const alreadyChecked =
  Array.isArray(props.checkedInDays) &&
  props.checkedInDays.some((d) => d.startsWith(todayISO));
console.log("checkedInDays after fetch:", props.checkedInDays);

const [loading, setLoading] = useState(false);
const [error, setError] = useState("");

return (
  <div className="subtask">
    {subtaskType === "simple" ? (
      <div className="info-subtask-simple">
        <div className="subtask-name">
          <input type="checkbox" checked={props.isCompleted} disabled />
          <label>{props.title}</label>
        </div>
        <div className="desc">{props.description}</div>
      </div>
    ) : subtaskType === "repeat" ? (
      <div className="info-subtask-repeat">
        <div className="subtask-name">
          <input type="checkbox" checked={props.isCompleted} disabled />
          <label>{props.title}</label>
        </div>
        <div className="desc">{props.description}</div>
        <div className="values-repeat">
          <div className="naming">
            <p>Дні для занять</p>
            <div className="days">
              {weekDays.map((day, i) => {
                const isActive = props.repeatDays?.includes(day.eng);
                return (
                  <div className={`day${isActive ? " active" : ""}`>
                    key={i}>
                      <p>{day.short}</p>
                    </div>
                );
              })}
            </div>
          </div>
        </div>
      </div>
    ) : null}
  </div>
);

```

```

</div>
{canCheckin && (
  <button
    className="done-btn"
    onClick={async () => {
      setError("");
      setLoading(true);
      try {
        await checkinRepeatableSubtask(
          props.taskId,
          props.id,
          new Date().toISOString()
        );
        await props.fetchTasks();
      } catch {
        setError("Помилка при фіксації прогресу.");
      }
      setLoading(false);
    }}
    disabled={!canCheckin || alreadyChecked || loading}
  >
    {alreadyChecked ? "Вже відмічено" : "Відмітити"}
  </button>
)}
{error && <p className="error">{error}</p>}
</div>
</div>
) : (
<div className="info-subtask-scale">
  <div className="subtask-name">
    <input type="checkbox" checked={props.isCompleted} disabled />
    <label>{props.title}</label>
  </div>
  <div className="desc">{props.description}</div>
  <div className="values">
    <div className="add-value">
      <p>Додати значення</p>
      <div className="add-action">
        <input
          placeholder="Введіть значення"
          type="number"
          value={addValue}
          onChange={(e) => setAddValue(e.target.value)}
          min={1}
        />
        <button
          className="add-value-btn"
          onClick={handleAddValue}
          disabled={!addValue || isNaN(addValue) || adding}
        >
          Додати
        </button>
      </div>
    </div>
  </div>

  <div className="scale-info">
    <div className="scale">
      <div

```

```

        className="color-scale"
        style={{
          width: props.targetValue
            ? `${Math.round(
              (props.currentValue / props.targetValue) * 100
            )}%`
            : "0%",
        }}
      <</div>
    </div>
    <p className="points">
      {props.currentValue || 0}/{props.targetValue || 0}
      {props.unit ? (
        <>
          {" "}
          <br />
          {props.unit}
        </>
      ) : null}
    </p>
  </div>
</div>
</div>
  )}
</div>
  );
}

```

Лістинг В.10 – Код з файлу TaskForm.jsx

```

<div className="task-form">
  <input
    className="title"
    placeholder="Введіть назву задачі"
    value={titleValue}
    onChange={(e) => setTitleValue(e.target.value)}
    required
  />
  <input
    className="description"
    placeholder="Опис вашої задачі"
    value={descriptionValue}
    onChange={(e) => setDescriptionValue(e.target.value)}
  />

  {/* Теги */}
  <label>Теги</label>
  <div className="tags-block">
    <div className="tags">
      {tags.map((tag) => (
        <span key={tag} className="tag">
          <button
            className="tag__remove"
            onClick={() => handleDeleteTag(tag)}
          >
            x
          </button>

```

```

        <span className="tag__text">#{tag}</span>
      </span>
    )))
  </div>
  <div className="tag-input-wrap">
    <span className="tag-hash">#</span>
    <input
      className="tag-input"
      value={tagInput}
      onChange={(e) => setTagInput(e.target.value)}
      onKeyDown={(e) => e.key === "Enter" && handleAddTag()}
      maxLength={30}
    />
    <button type="button" className="tag-add-btn"
onClick={handleAddTag}>
      +
    </button>
  </div>
</div>

<div className="toggles">
  <div>
    <ToggleSwitch
      checked={showDeadline}
      onChange={(e) => setShowDeadline(e.target.checked)}
      label="Дедлайн"
    />
    {showDeadline && (
      <input
        type="datetime-local"
        value={deadlineDateValue}
        onChange={(e) => setDeadlineDateValue(e.target.value)}
        required
      />
    )}
  </div>
  <div>
    <ToggleSwitch
      checked={showReminder}
      onChange={(e) => setShowReminder(e.target.checked)}
      label="Нагадування"
    />
    {showReminder && (
      <input
        type="datetime-local"
        value={notificationDateValue}
        onChange={(e) => setNotificationDateValue(e.target.value)}
        required
      />
    )}
  </div>
</div>

```

Лістинг В.11 – Код з файлу AuthService.js

```

export async function googleLogin(googleToken) {
  const res = await fetch(`${API_URL}/api/Auth/google-login`, {
    method: "POST",

```

```

    headers: {
      "Content-Type": "application/json",
    },
    body: JSON.stringify({ token: googleToken }),
  });

  const data = await res.json();

  if (!res.ok) {
    throw new Error(data.message || "Помилка при вході через Google");
  }

  localStorage.setItem("token", data.token);

  const payload = parseJwt(data.token);
  if (payload?.sub) {
    localStorage.setItem("id", payload.sub);
  }

  return data;
}

```

Лістинг В.12 – Код з файлу Tasks.jsx

```

export async function checkinRepeatableTask(taskId, date) {
  return fetch(`${API_URL}/api/Task/${taskId}/checkin`, {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      // додай авторизацію, якщо потрібно
    },
    body: JSON.stringify({ date }),
  }).then((res) => {
    if (!res.ok) throw new Error("Check-in failed");
    return res.json().catch(() => ({}));
  });
}

```

Лістинг В.13 – Код функції fetchCategories()

```

export async function fetchCategories() {
  const res = await fetch(`${API_URL}/api/Category`);
  if (!res.ok) throw new Error("Не вдалося отримати категорії");
  return await res.json();
}

```

Лістинг В.14 – Приклад обробки помилок

```

export async function deleteTask(id) {
  const res = await fetch(`${API_URL}/api/Task/${id}`, {
    method: "DELETE",
  });
  if (!res.ok) throw new Error("Не вдалося видалити задачу");
}

```