

Документація. Волонтерська діяльність MVP

🎯 Ціль [↗](#)

Створити онлайн-платформу, яка спрощує та прискорює процес надання допомоги тим, хто її потребує, за допомогою залучення охочих благодійників. Надання зручного інтерфейсу створення, надання даних комунікації, а також пошуку людей яким потрібна допомога.

📋 Вимоги до функціоналу



Загальні відомості. Перший крок [↗](#)

Користувача відвідувавши сервіс, зустрічає головне меню, звичайна статична сторінка з загальною інформацією про сам проект, FAQ, і перехід до сторінки оголошень, футер, хедер з навігацією до авторизації, аутентифікації, сторінка оголошень.

Користувач який відвідує сервіс без аутентифікації, має можливість переглядати вакансії, але не отримувати контакту інформацію людини яка створила оголошення про допомогу, а також створювати свої оголошення.

Сама сторінка (скрін, дизайн): TODO

Реєстрація і аутентифікація користувачів [↗](#)

- Для обох типів користувачів (ті хто просить допомогу, і ті, хто надає її) потрібно мати окремі форми реєстрації.
- Система аутентифікації повинна бути безпечною та надійною.

Реєстрація:

На формі реєстрації в користувача є вибір з двох кнопок. В залежності від вибору, змінюються можливості обох юзерів на сервісі.

- Хочу допомогти - користувачу буде доступний перегляд вакансій, і також отримання додаткової інформації по оголошенню.
- Хочу допомоги - користувачу буде доступна кнопка створення одного оголошення про допомогу.

Поля реєстрації:

- Тип юзера (кнопка вибору типу юзера)
- First name
- Second name
- Email
- Password
- Repeat password

Логіка:

Перевіряється чи паролі співпадають, перевірка валідації. Шифруються паролі і всі дані записуються в базу. Після успішного проходження реєстрації, користувач попадає на основне меню знову, вже будучи авторизованим.

API:

POST method /registration

Parameter	Type	Description
api-key	string	Api key for getting access to the endpoint

Body:

```

1 {
2   "firsrt_name": str
3   "second_name": str
4   "user_type": "need help" | "give help"
5   "email": str
6   "password_1": str
7   "password_2": str
8 }
```

Response: (201)

```

1 {
2   "status": "ok"
3 }
```

Response: (404)

```

1 {
2   "status": "validation_error"
3 }
```

Сама сторінка (скрін, дизайн): TODO

Аутентифікація:

На формі аутентифікації. Користувач вводить свої дані для отримання доступу до свого аккаунта.


Поля аутентифікації:

- Email
- Password

Логіка:

Перевіряється чи є такий користувач, чи збігається пароль. Генеруються токен JWT. Після успішної аутентифікації, користувач попадає на основне меню знову, вже будучи авторизованим. І ідентифікуєця в системі.

API:

 POST method /login

Parameter	Type	Description
api-key	string	Api key for getting access to the endpoint

Body:

```
1 {
2   "email": str
3   "password": str
4 }
```

Response: (200)

```
1 {
2   "refresh": str # token
3   "access": str. # token
4 }
```

Response: (404)

```
1 {
2   "status": "validation_error"
3 }
```

Сама сторінка (скрін, дизайн): TODO

Профіль юзера:

Після успішної аутентифікації, в системі можна отримувати дані про користувача, для подальшої ідентифікації його в системі

Логіка:

За допомогою токена JWT. Після успішної аутентифікації, можна витягнути дані юзера щоб ідентифікувати його тип аккаунта, специфічного відображення, і повної інформації по користувачу в його профілі.

API:

 GET method /profile

AuthToken:

Parameter	Type	Description
Bearer-token	string	User token

Header:

Parameter	Type	Description
api-key	string	Api key for getting access to the endpoint

Body:

```
1 {}
```

Response: (200)

```
1 {
2   "user_type": str
```

```
3  "first_name": str
4  "second_name": str
5  "email": str
6  "date_joined": datetime
7  }
```

Response: (404)

```
1  {
2    "status": "validation_error"
3  }
```

Сама сторінка (скрін, дизайн): TODO

Створення та перегляд запитів на допомогу [↗](#)

- Форма створення запиту на допомогу повинна містити поля для детального опису ситуації або проблеми, для якої потрібна допомога.
- Користувачам, які просять допомогу, слід надати можливість завантаження зображень або документів, які можуть допомогти пояснити їхню ситуацію.
- Після створення запиту на допомогу користувачам слід надати можливість перегляду свого запиту та редагування або видалення його за потреби.

Створення оголошення про допомогу.

Користувач (який просить допомогу) у своєму профілі має можливість створити оголошення. Ліміт **1 оголошення**, це створено, через те щоб люди не спамили різними оголошеннями, є потреба - 1 оголошення з детальним описом.

Поля на створення оголошення:

- **Title** - Короткий, але зрозумілий заголовок, що відображає суть оголошення.
- **Description** - Розгорнутий опис ситуації або потреби, де можна розповісти більше про себе, свою ситуацію та те, яка саме допомога потрібна.
- **Categories** - є список категорій з яких можна вибрати, для подальшого фільтрування.
- **Location** - Інформація про місце, де потрібна допомога (наприклад, місто, район), що допоможе користувачам знаходити оголошення, які є для них релевантними.
- **Contact information** - Обов'язково потрібно вказати контактні дані (електронна пошта, номер телефону) для зв'язку з автором оголошення авторизованих користувачів.
- **Pictures** - Можливість додати фото, що ілюструє ситуацію або потребу, що може зробити оголошення більш привабливим та довірливим для користувачів.
- **Status** - Інформація про поточний статус оголошення (**активне, закрито, виконано**), що допоможе користувачам зрозуміти, чи є оголошення ще актуальним.
- **Time validity** - Вказання терміну, протягом якого потрібна допомога (наприклад, терміново, до певної дати).
- **Priority** - Можливість вказати, наскільки термінова або важлива потреба, щоб користувачі могли швидше реагувати на неї (**помірна, важка ситуація**).

Логіка:

Коли в профілі користувач нажимає кнопку "Створити оголошення на допомогу", він переходить на сторінку з формами для створення оголошення. Після заповнення, дані виставляються на сайті, зберігаються в базу даних.

API:

i POST method /create_help_advert

Parameter	Type	Description
api-key	string	Api key for getting access to the endpoint

AuthToken:

Parameter	Type	Description
Bearer-token	string	User token

Body:

```
1 {
2   "title": str
3   "description": str
4   "categories": list[str] # список категорій
5   "location": str # місце локації
6   "contats": {
7     "email": str,
8     "phone": str
9   }
10  "pictures": files
11  "status": str # (активне, закрито, виконано)
12  "time_validity": datetime
13  "priority": str # (помірна, важка ситуація)
14 }
```

Response: (200)

```
1 {
2   "status": "ok"
3 }
```

Response: (404)

```
1 {
2   "status": "validation_error"
3 }
```

Сама сторінка (скрін, дизайн): TODO

Профіль з оголошенням про допомогу.

Користувач (який просить допомогу) у своєму профілі бачить своє поточне оголошення, і має можливість його видалити.

Поля які відображаються в самому профілі:

- **Name** - Ім'я користувача якому належить аккаунт.
- **Id** - Id як ідентифікатор самого оголошення.
- **Title** - Заголовок оголошення.
- **Description** - Опис оголошення.

Логіка:

Коли користувач у своєму профілі, в нього є можливість видалити або створити оголошення. Для видалення йому потрібно вибрати конкретне оголошення, і натиснути кнопку видалити.

API:

Отримання оголошень користувача:

 GET method /get_profile_advert

Parameter	Type	Description
api-key	string	Api key for getting access to the endpoint

AuthToken:

Parameter	Type	Description
Bearer-token	string	User token

Body:

```
1 {}
```

Response: (200)

```
1 {
2   "advert_id": int
3   "user_name": str
4   "title": str
5   "description": str
6 }
```

Response: (404)

```
1 {
2   "status": "validation_error"
3 }
```

Видалити оголошення

 DELETE method /delete_help_advert

Parameter	Type	Description
api-key	string	Api key for getting access to the endpoint

AuthToken:

Parameter	Type	Description
Bearer-token	string	User token

Body:

```
1 {
2   "
3 }
```

Response: (200)

```
1 {
2   "advert_id": int
3 }
```

Response: (404)

```
1 {
2   "status": "validation_error"
3 }
```

Сама сторінка (скрін, дизайн): TODO

Самі оголошення. [↗](#)

Загальна сторінка з оголошеннями про допомогу.

Будь-який користувач може переглядати сторінку на якій знаходяться оголошення інших користувачів.


Поля кожного окремого оголошення на сторінці:

- **Author** - Ім'я автора
- **Title** - Короткий, але зрозумілий заголовок, що відображає суть оголошення.
- **Description** - Розгорнутий опис ситуації або потреби, де можна розповісти більше про себе, свою ситуацію та те, яка саме допомога потрібна.
- **Categories** - є список категорій з яких можна вибрати, для подальшого фільтрування.
- **Location** - Інформація про місце, де потрібна допомога (наприклад, місто, район), що допоможе користувачам знаходити оголошення, які є для них релевантними.
- **Pictures** - Можливість додати фото, що ілюструє ситуацію або потребу, що може зробити оголошення більш привабливим та довірливим для користувачів.
- **Status** - Інформація про поточний статус оголошення (**активне, закрито, виконано**), що допоможе користувачам зрозуміти, чи є оголошення ще актуальним.
- **Time validity** - Вказання терміну, протягом якого потрібна допомога (наприклад, терміново, до певної дати).
- **Priority** - Можливість вказати, наскільки термінова або важлива потреба, щоб користувачі могли швидше реагувати на неї (**помірна, важка ситуація**).

Логіка:

Коли користувач заходить на сторінку зі всіма оголошеннями, він може побачити перед собою оголошення окремих користувачів, біля кожного є кнопка, на яку може натиснути тільки авторизований користувач, щоб потрапити в середину оголошення, і отримати контактну інформацію, для неавторизованих користувачів цієї кнопки немає. Збоку є фільтр який працює на всі оголошення по **категоріям, локації, статусу**. Сама сторінка це load сторінка, прокручуємо нижче і появляються нові оголошення.

API:

 GET method /get_all_advert

Parameter	Type	Description
api-key	string	Api key for getting access to the endpoint

AuthToken:

Parameter	Type	Description
Bearer-token	string	User token

Body:

Витягується за один запит 10 оголошень.

Filters:

- byCategory=[str, ...]
- byLocation=[str,...]
- byStatus=str

Response: (200)

```

1  {
2    "advertisies": [
3      {
4        "advert_id": int
5        "title": str
6        "description": str
7        "categories": list[str] # список категорій
8        "location": str # місце локації
9        "pictures": files
10       "status": str # (активне, закрито, виконано)
11       "time_validity": datetime
12       "priority": str # (помірна, важка ситуація)
13     },
14     ...
15   ]
16 }
```

Response: (404)

```

1  {
2    "status": "validation_error"
3  }
```

Сама сторінка (скрін, дизайн): TODO

Конкретне оголошеннями про допомогу.

Аутентифікований користувач може перейти на сторінку конкретного оголошення, де буде приватна інформація автора оголошення, а саме його контактні данні. За допомогою цього в подальшому, користувач який прагне допомогти, зможе зв'язатися з автором.

Поля кожного окремого оголошення на сторінці:

- **Author** - Ім'я автора
- **Title** - Короткий, але зрозумілий заголовок, що відображає суть оголошення.
- **Description** - Розгорнутий опис ситуації або потреби, де можна розповісти більше про себе, свою ситуацію та те, яка саме допомога потрібна.
- **Categories** - є список категорій з яких можна вибрати, для подальшого фільтрування.
- **Location** - Інформація про місце, де потрібна допомога (наприклад, місто, район), що допоможе користувачам знаходити оголошення, які є для них релевантними.
- **Pictures** - Можливість додати фото, що ілюструє ситуацію або потребу, що може зробити оголошення більш привабливим та довірливим для користувачів.
- **Status** - Інформація про поточний статус оголошення (**активне, закрито, виконано**), що допоможе користувачам зрозуміти, чи є оголошення ще актуальним.
- **Time validity** - Вказання терміну, протягом якого потрібна допомога (наприклад, терміново, до певної дати).
- **Priority** - Можливість вказати, наскільки термінова або важлива потреба, щоб користувачі могли швидше реагувати на неї (**помірна, важка ситуація**).
- **Contact information** - **Список контактів юзера.**

Логіка:

Користувач перейшов на сторінку з детальною інформацією оголошення.

API:

 GET method /get_advert/{advert_id}

Parameter	Type	Description
api-key	string	Api key for getting access to the endpoint

AuthToken:

Parameter	Type	Description
Bearer-token	string	User token

Body:

```
1 {}
```

Response: (200)

```
1 {
2   "advert_id": int
3   "title": str
4   "description": str
5   "categories": list[str] # список категорій
6   "location": str # місце локації
7   "pictures": files
8   "status": str # (активне, закрито, виконано)
9   "time_validity": datetime
10  "priority": str # (помірна, важка ситуація)
11  "contats": {
12    "email": str,
13    "phone": str
14  }
```

```
15 }
```

Response: (404)

```
1 {  
2   "status": "validation_error"  
3 }
```

Сама сторінка (скрін, дизайн): TODO