**LO-FI**

Призначення та коротка характеристика програми:

Головною ідеєю є те, що користувачу пропонується плейлист або цілий альбом для слухання, який обирається на основі настрою або почуття користувача на момент користування. Уявімо, що існує людина А, яка постійно працює, але обожнює музику; ось їй це неабияк знадобиться.

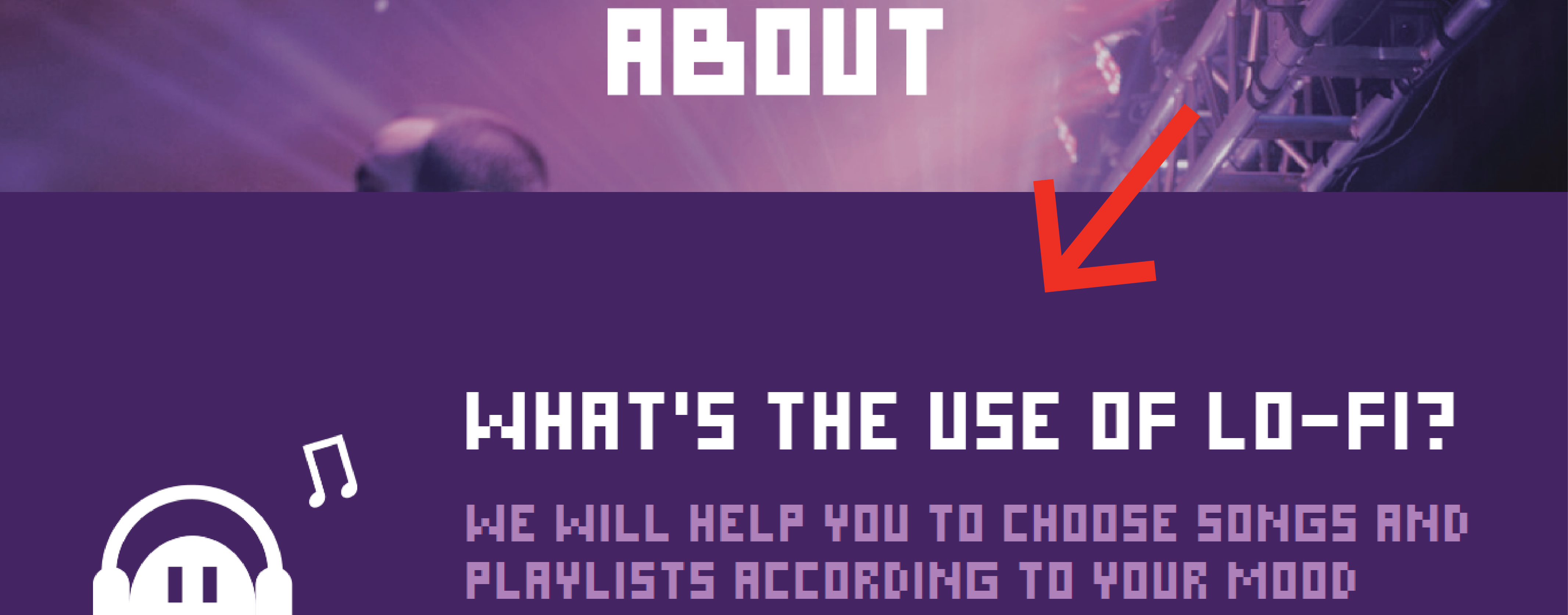
Функціонал сайту передбачає лише єдину інтеракцію – введення, так би мовити, стану людини. Далі запит обробляється і вже відноситься до певної категорії. Наступним кроком буде пошук пісень у SoundCloud за хештегами та вмістом і поміщення чотирьох знайдених до альбому.

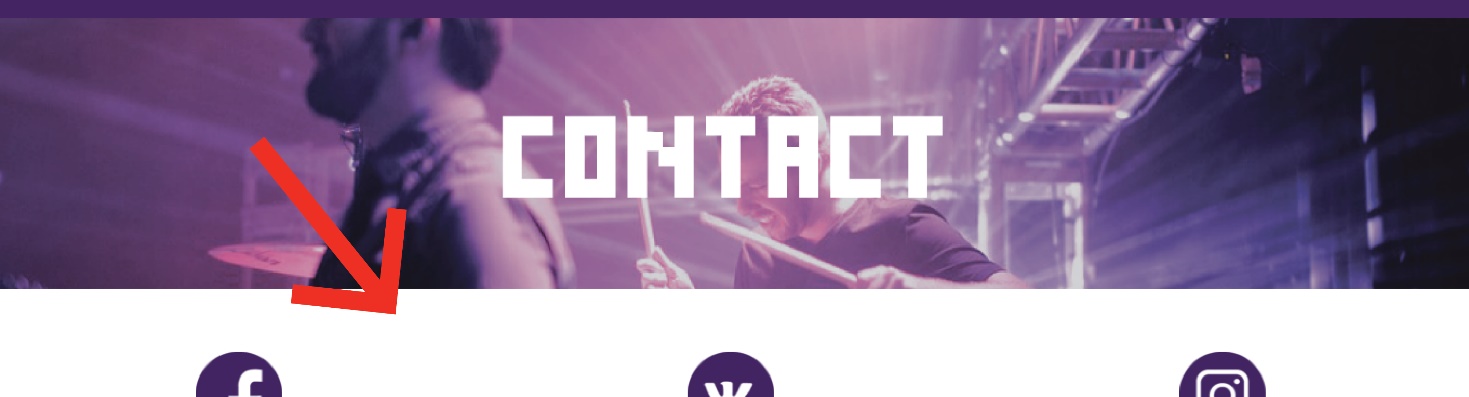
**Структура сайту**

**(**<http://lofi.pythonanywhere.com/>**)**

1. Навігація



1. Поле вводу
2. Опис
3. Контактні дані



**Структура проекту**

1. Головний модуль server.py
2. Допоміжні модулі arrays.py, solve.py
3. Модулі lo-fi.html, lo-fi\_music.html, style.css, style\_music.css (/templates, /static/css)
4. Файли дизайну (/static)

Вхідні та вихідні дані програми:

При роботі над даним проектом використовувались такі дані, як: дані отримані з пошуку, HTML – віджети треків та посилання у SoundCloud. Ці дані надходять у форматі об’єктів типу Resource (наприклад, soundcloud.resource.Resource object at 0x03C17950). Після обробки запиту та виконання дій над отриманими даними, функціонал SoundCloud ADT повертає масив із віджет-посиланнями на пісні у SoundCloud.

**SoundCloud ADT в LO-FI**

Структура програми з коротким описом модулів, функцій, класів та методів:

SoundCloud ADT – це тип даних, що надає набір методів, які допомагають працювати із даними отриманими при роботі з SoundCloud.

**Структура SoundCloud ADT**

**Ім’я класу:** SoundCloud

При створенні екземпляру даного класу користувач мусить ввести одне слово, яке характеризує його настрій. В подальшому SoundCloud використає дану інформацію для пошуку і підбору пісень.

Після отримання відповіді від користувача, дані будуть записані у змінну self.text. Після цього дана змінна буде використовуватись у методах класу.

**Методи:**

make\_url(self)

make\_playlist(self)

get\_widgets(self)

Опис методів:

*make\_url(self)*

Кінцевий результат: масив із посиланнями на пісні

На початку роботи, якщо змінна self.text не пуста. створюється змінна self.client. Створюючи дану змінну типу client, ми вводимо свій клієнтський id та присвоюємо їй всю необхідну інформацію для роботи із SoundCloud АРІ. Далі за допомогою методу get(‘/tracks’,q = self.text) створюється список з інформацією про пісні (tracks). Наступним кроком є перетворення даних з Resource об’єктів на список кортежів за допомогою методів fields() та items(). Після цього інформація сортується і відбираються лише лінки на треки. Усі посилання записуються до масиву url. Функція повертає вищезгаданий масив url

*make\_playlist(self)*

Кінцевий результат: масив із 4 посиланнями на пісні.

Робота починається із виклику функції *make\_url(self)* і присвоєння змінній url повернутого значення. Далі проводиться перевірка на вміст масиву. Якщо к-сть елементів менша за 4, то наступні дії не виконаються. Наступним кроком є створення масиву playlist та довільного вибору пісень, які будуть наповнювати утворений масив. Функція повертає заповнений масив playlist.

*get\_widgets(self)*

Кінцевий результат: масив із \*віджет-посиланнями.

Спочатку утворюється масив info та викликається функція *make\_playlist(self)* і повернуте значення присвоюється утвореному масиву playlist. Наступний крок – ітерація playlist. Після ітерації отримуємо масив widgets із віджетами. Потім утворюємо масив final і ітеруємо масив widgets. Під час ітерації відділяємо \*віджет-посилання із віджетів і додаємо їх до масиву final. Останній крок повернення функцією масиву final.

**Загальні відомості**

Даний тип даних використовувався при розробці сайту (lofi.pythonanywhere.com) та проекту загалом.

При розробці SoundCloud ADT було використано таку структуру даних як DynamicArray

Також, при створенні SoundCloud ADT було імпортовано такі бібліотеки та модулі, як: DynamicArray (з модулю arrays), soundcloud, random.

\*Віджет-посилання – посилання, які використовуються у HTML віджетах як джерело на відео/аудіо\*

**Опис використаних модулів:**

З модуля arrays масив DynamicArray:

\_\_init\_\_(self):

Створення змінних

self.\_size = 0 # actual number of elements

self.\_capacity = 1 # default array capacity

self.\_elements = self.\_make\_array(self.\_capacity)

\_\_len\_\_(self):

Повернення довжини масиву

append(self, value):

Додавання значення value до масиву

remove(self, value):

Видалення значення value до масиву

insert(self, index, value):

Вставлення значення value за індексом index до масиву

\_make\_array(self, size):

Створення нового масиву розміру size

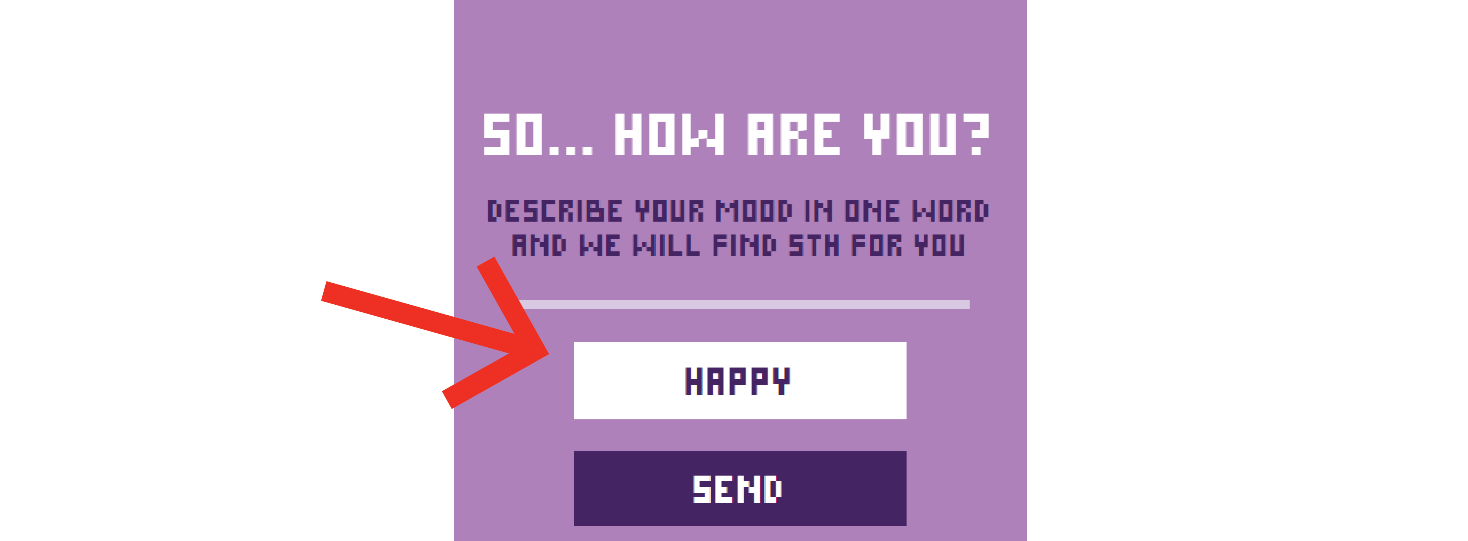
\_resize(self, capacity):

Змінити розмір даного масиву до місткості capacity

\_\_getitem\_\_(self, index):

Повертає значення елементу за індексом index

**Інструкція по користуванню**

1. Зайти на сайт <http://lofi.pythonanywhere.com/>
2. Знайти поле вводу як у пункті №2 розділу «Структура сайту»
3. Описати свій настрій одним словом (англійською) і ввести в форму так як показано на ілюстрації.
4. Прослухати пісні.
5. \*За бажанням\* Перейти до секції «Contacts» і поділитись своєю думкою.

\*Винятки

Інколи при роботі із сайтом ви можете отримати буквально нічого



Це може статись у таких випадках:

1. Коли ви нічого не ввели
2. Коли жодної пісні не знайдено
3. Коли пісні заблоковані авторами для прослуховування
4. При помилці доступу і ряду інших http винятків
5. При перевищенні кількості запитів