**Local Search System for Wiki News**

2020-08-06

Leader: Shashidhar Reddy Daida (11399488) Module: User Management

Members: Rohith Reddy Poreddi (11409019) Module: Item Management

Asrithamary Kondaveeti (11396153) Module: Scrapy Management & Searching

Shresta Sankapally (11395196) Module: Indexing & Searching

**INTRODUCTION**

The main objective of the project is to develop a local search system for Wiki News website. The project aimed to scrap the data from wiki news website and store in the local database. This system contains the latest news as it is being scrapped from wiki news website frequently. The system will help the users to search and read the news. The system holds the information of every user whoever registers with the website.

1. **OBJECTIVES**

* To develop an effective and efficient Local Search System for Wiki News.
* Scrap the news data from wiki news website and store in the local database.
* To provide an efficient user interface for users where user can search and read news content.
* This system is maintained by Admin who has access to user management, item management, Web Scrapping and Indexing.
* Holds user information.
* Latest news of wiki news in the local database.
* To provide a search interface that could retrieve all the items by name or content of each item.

1. **OVERALL ARCHITECTURE**

A close up of text on a white background

Description automatically generated

Overall architecture of Local Search System for Wiki News

There are three main functional components in Local Search System for Wiki News.

* Web Scrapping
* Indexing
* Searching

**Web Scrapping:**

Web Scrapping is a technique to extract data from websites and save to database or local file in the computer. It is a form of copying data from web to a central local database for later retrieval. For this system, data has been scrapped from Wiki News website and stored in the local database. Python web scrapy has been used to scrap the data.

**Indexing:**

Indexing is a process in which the information is organized before a search to enable super-fast response to queries. Searching through individual pages for keywords and topics would be a very slow process for search engines to identify relevant information. Instead, search engines (including Google) [use an inverted index, also known as a reverse index.](https://allotment.digital/learn/technical-seo/how-search-engines-work/indexing/)

In this project, we have used Django haystack as a indexing algorithm and used elastic search in the backend.

**Searching:**

Searching is a component provided with the user interface where user can input some keywords and search for news items that are stored in the local database.

**Integration of the components:**

In this local search system project, all these three components are interdependent.

Firstly, we scrap/crawl the news items from wiki news website by using python web scrapy and store these news items in our local database.

The local database is updated with the news items, admin will run indexing in the backend so that all the news items data will be indexed and stored in the defined indexes. When user search for a news item in the interface, elastic search engine searches for the matching keywords in the indexed data and returns the relevant data in the search results. If there is no keyword match in the indexed data or if the data is not indexed by the admin then there will be no search results.

1. **SYSTEM DESIGN**

**3.1 System flow**

A screenshot of a cell phone

Description automatically generated

**System flow of Local Search System for Wiki News**

There are two types of logins in the system.

1. Admin login
2. Anonymous User login

**Admin login:**

There are three tasks performed by admin in the backend. Scrapy settings, Indexing and User creation.

**Scrapy settings:** In this settings, admin sets the url of the wiki news website <https://en.wikinews.org/wiki/Main_Page> and runs the scrapping algorithm. Once the scrapping is done, the latest news items from wiki news website will be crawled and stored in the local database.

**Indexing:** Admin runs a command “python manage.py rebuild\_index” in the backend. When this command is executed, the news data in the local database will be reindexed according to the index class defined in the code. Once the data is indexed, the data will be stored in a document file in the backend.

User creation:

Admin creates new user with a unique username and password and with the choice of admin rights.

**Anonymous User login:**

When anonymous user or non admin user logs into the system, the user can see a search page where they can search for a news item using keywords. When user enters keyword and searches, the search engine in the backend will look for that keyword in that indexed data and returns the matching results. If there are no matching keywords found in the indexed data, then the page displays saying “ No results found”.

* 1. **System functions**

A close up of a sign

Description automatically generated

System Functions of Local Search System for Wiki News

According to the system flow, the system has been divided into five modules/functions. And they are as follows:

* Users Management
* Scrapy Management
* Items Management
* Indexing
* Searching

**Users Management:**

In this module, admin can provide authentication to new user, modify the user passwords, delete the user and update the choice of admin rights. Only admin has access to make a scrapy setting, indexing setting. Anonymous user can login into the system and search for the news items and read the news.

**Scrapy Management:**

Scrapy is also called as web spider. It is built in the system. The admin user could config its setting such as the destination the system could grab. In this system, Python web scrapy has been used to scrap the data from wiki news website.

**Item Management:**

This module provides an interface that could get, modify or delete any news item grabbed by the scrapy module.

**Indexing:**

Indexing module is built in the system. When the admin user executes thefunction, the module will make the indexing work to the items in thelocal database according to the search algorithm. In this system, Django haystack algorithm has been used for indexing and elastic search as the search engine in the backend.

**Searching:**

In this module, anonymous user logins into the system, enters the keyword and search for news items that are stored in the local database.

* 1. **Database Design:**

MySQL Workbench 8.0 has been used as a database server. Tables are created in the database “Models.py” of each application. Database connection is configured in “settings.py” of the project.

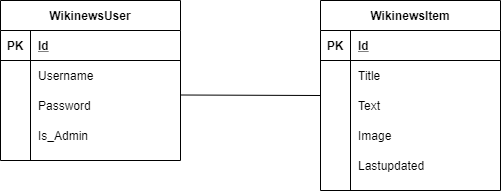
According to the system requirements, there are two tables in the database.

1. WikinewsUser
2. WikinewItem

These two table are not related according to the system function. In these tables, column “Id” is the autoincremented primary key which means once an Id is assigned it cannot be reassigned to any new record even the record with that Id is deleted.

**ER Diagram:**

The relationship between the tables is shown as untitled relation as there is no relationship.



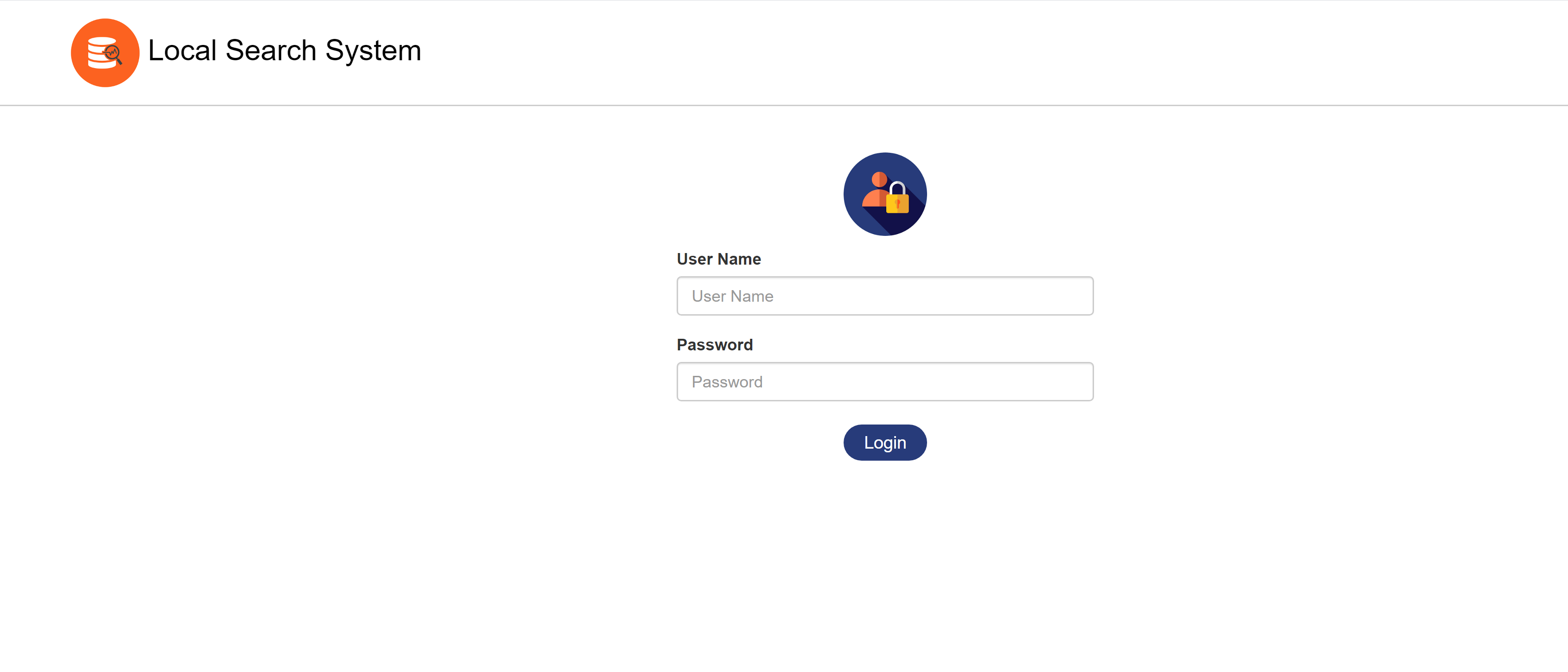
**Data dictionary:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table Name** | **Attribute Name** | **Data Type** | **PK** | **Contents** |
| WikinewsUser | Id | Int AI | PK | Id for User |
| Username | varchar(200) |  | Username |
| Password | varchar(100) |  | Password |
| Is\_admin | tinyint(1) |  | Admin rights selection |
| WikinewsItem | Id | Int | PK | Id for item |
| Title | varchar(500) |  | Item title |
| Text | longtext |  | News content |
| Image | longtext |  | First Image of the news from the article |
| Lastupdated | date |  | Time at which news last updated on the website |

**4. System Testing:**

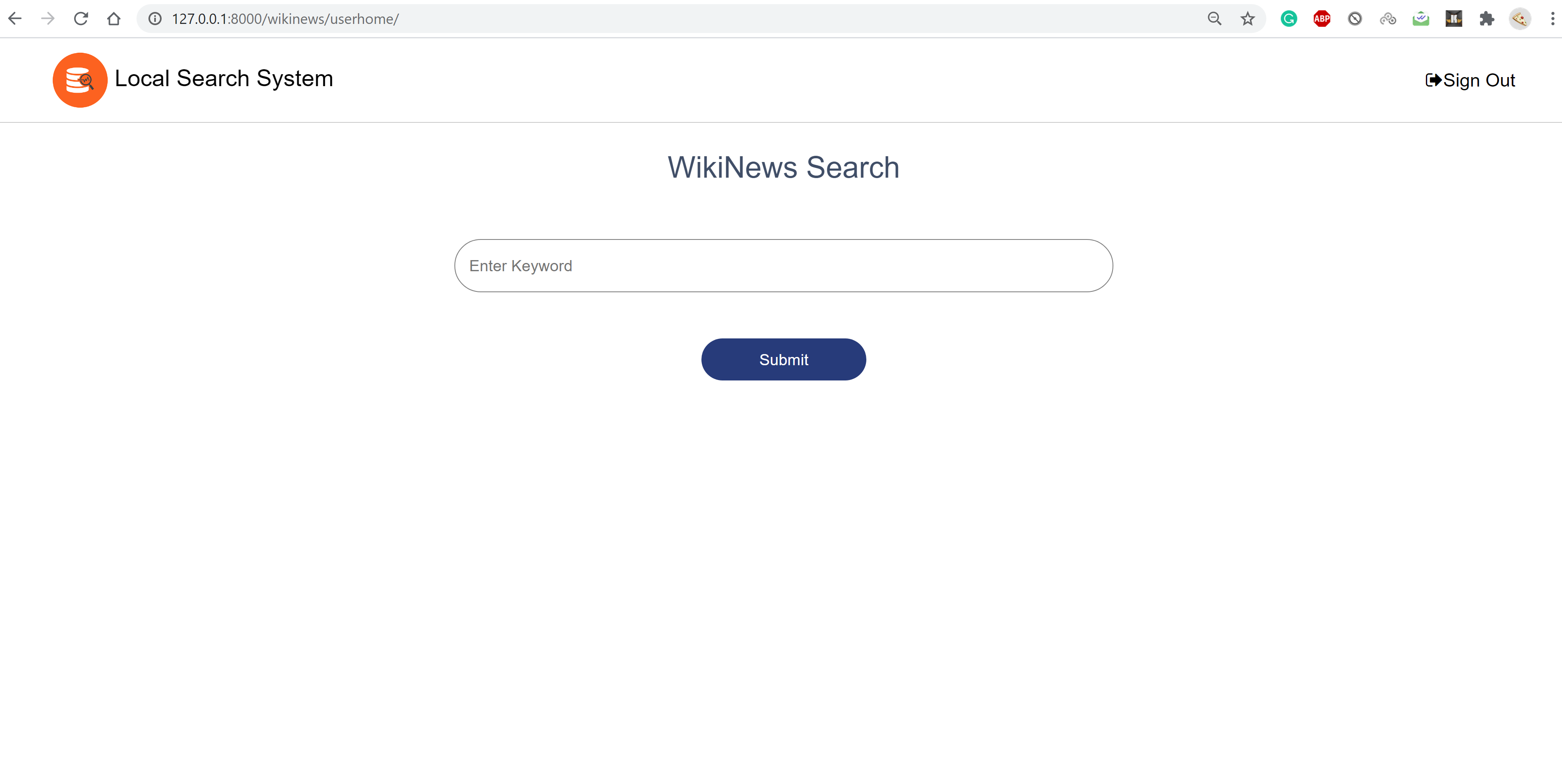
**Login page of Local Search System:**

Below is the screenshot of login page for local search system for both admin and anonymous user.

****

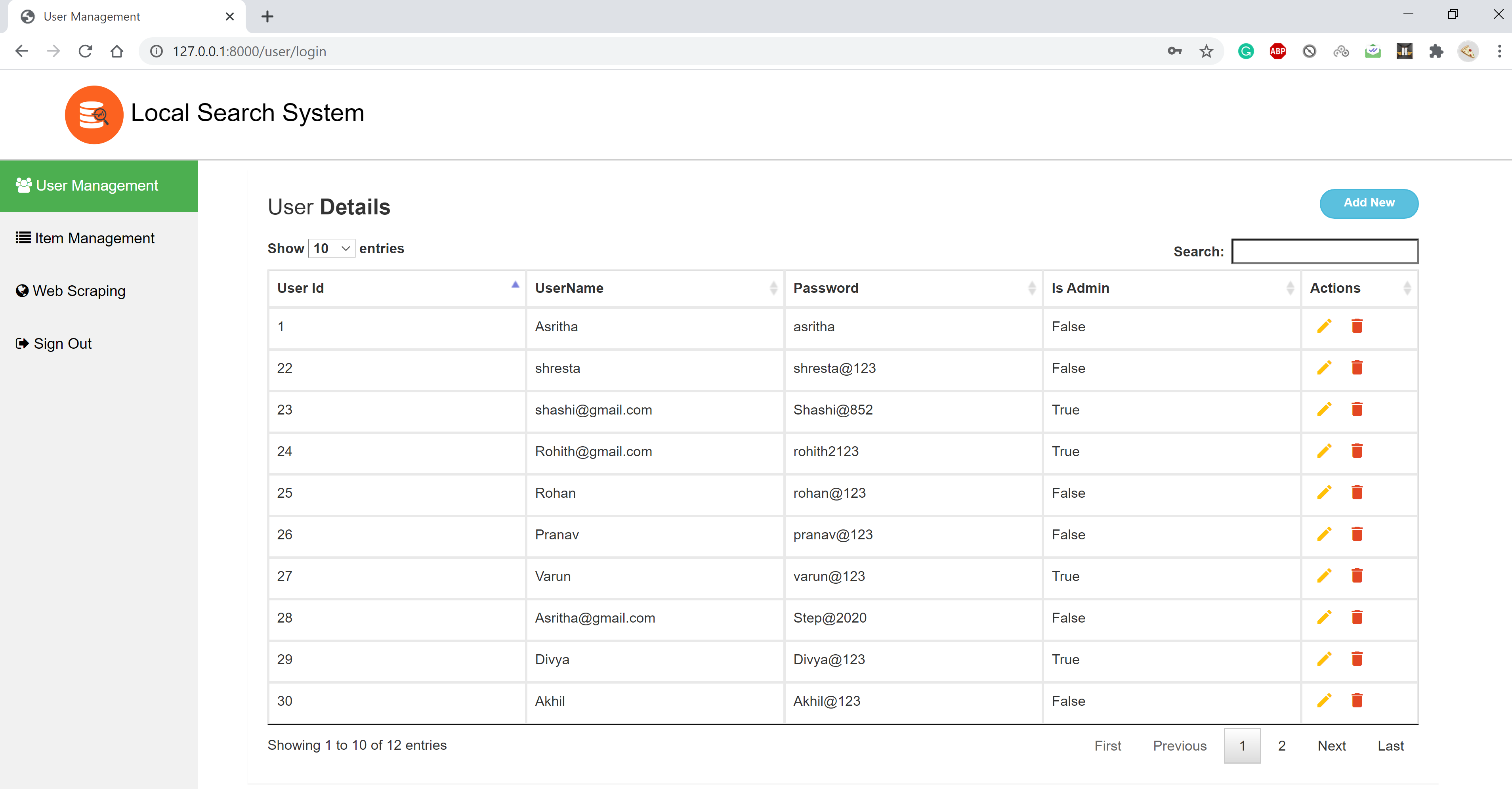
**Search page for anonymous user**

User can search the news from below search page once logins to the system.

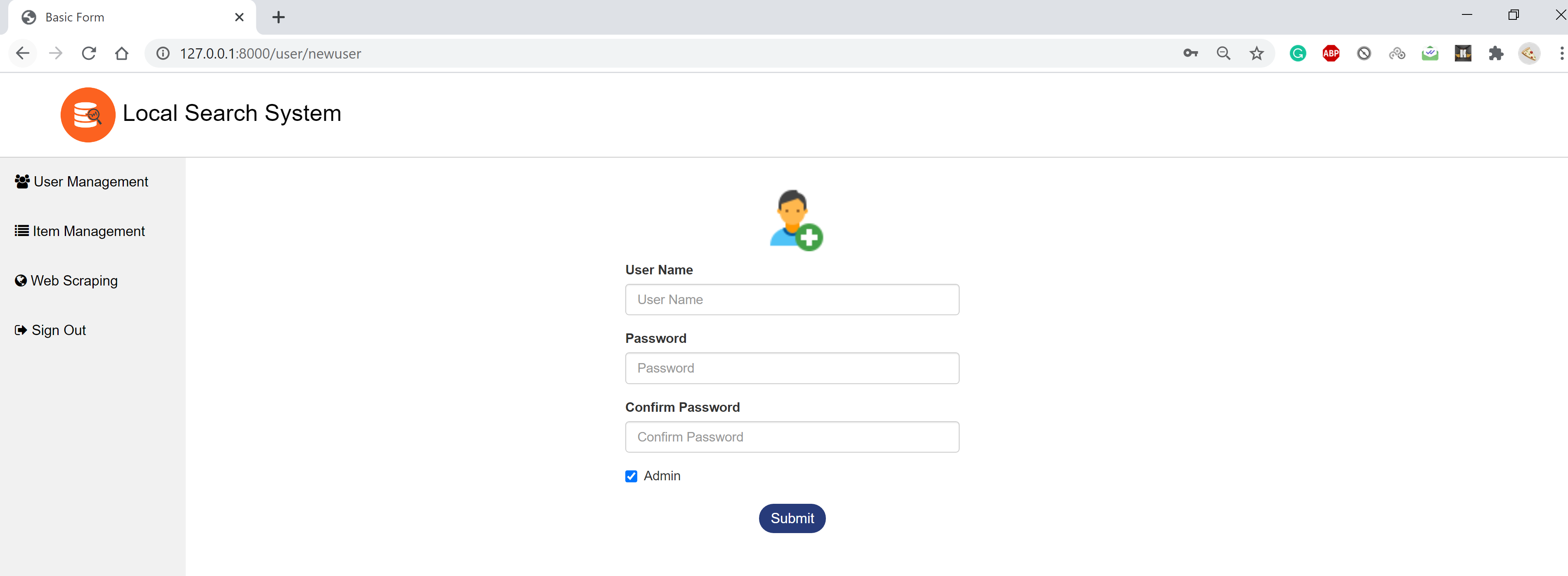


**Admin page in the system:**

Once admin logins to the system, under user management tab User details are displayed that are fetched from “WikinewsUser” table. Search bar is provided in this page to search for users. Admin can add new user from this page by clicking on “Add new” button.

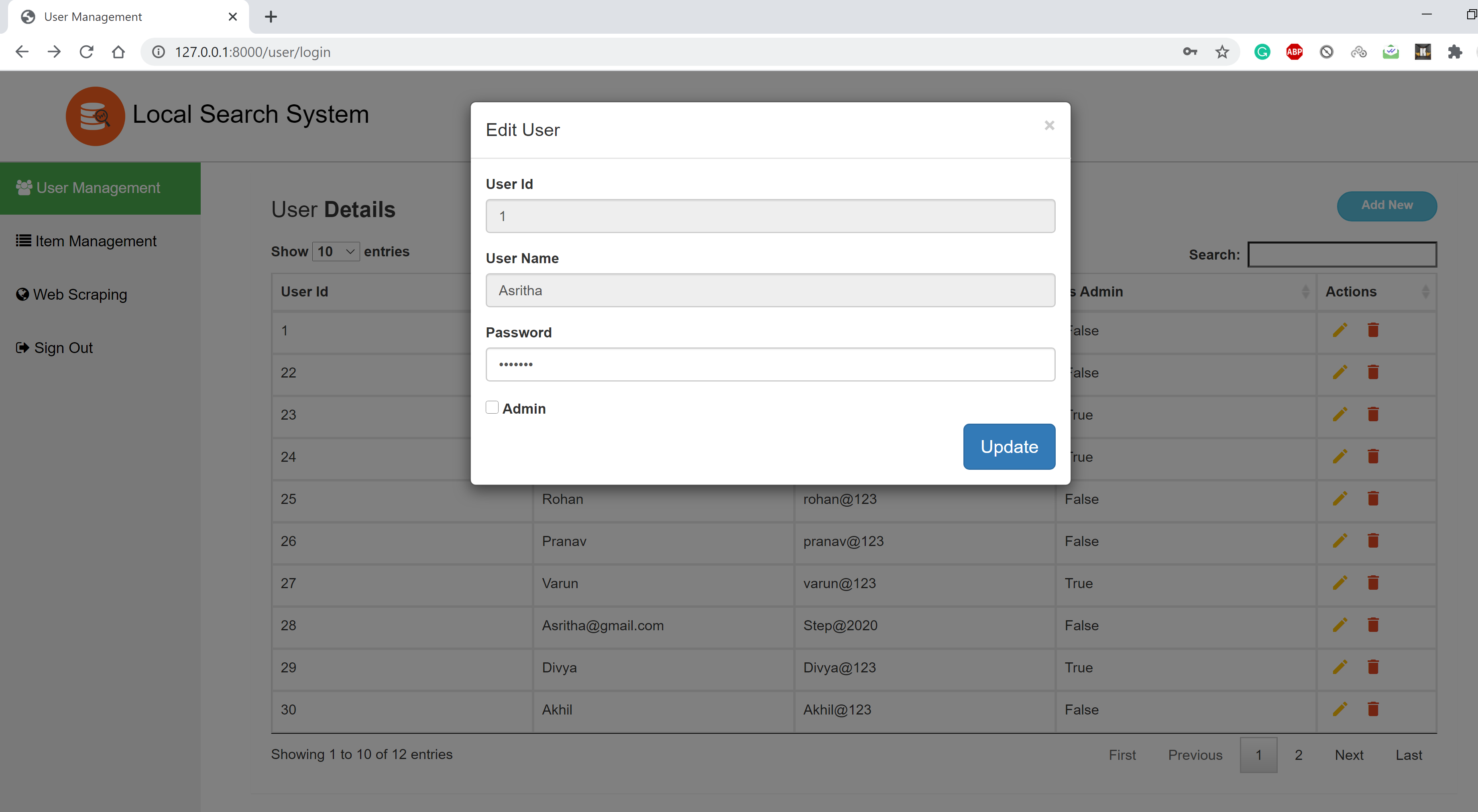


Once admin clicks on add new button in the above page, a new page displays as below. Admin can edit or delete the user.

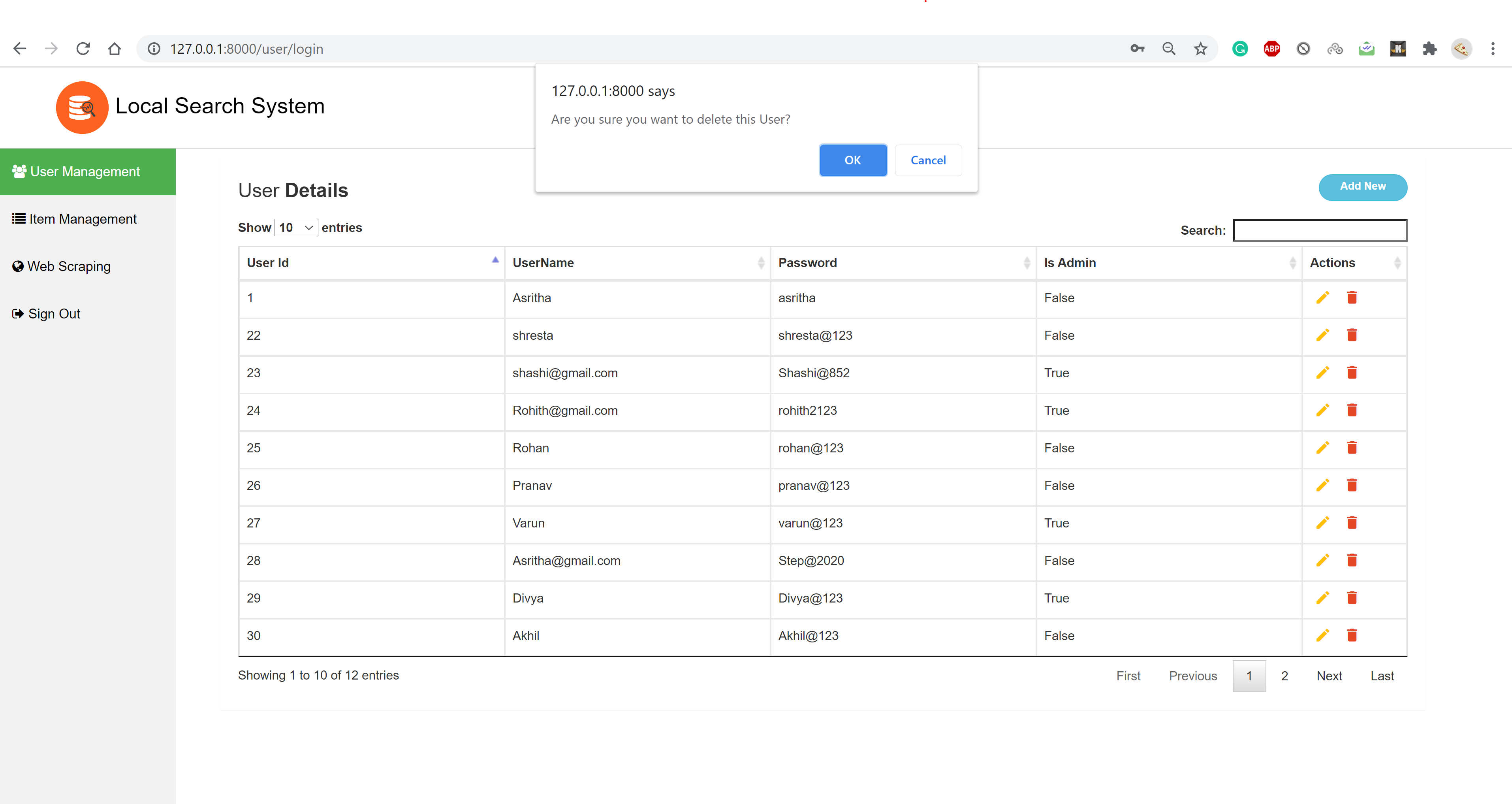


At this creation itself, user can give admin rights to the users by checking the “Admin” check box. If this is checked, it indicates user has admin rights.

Admin can update or modify the user information by clicking on edit button from User details page. Once admin clicks on edit, pop up appears as below. After modifying or updating the fields, “Update” button should be clicked to reflect the changes. This updates in the database as well.

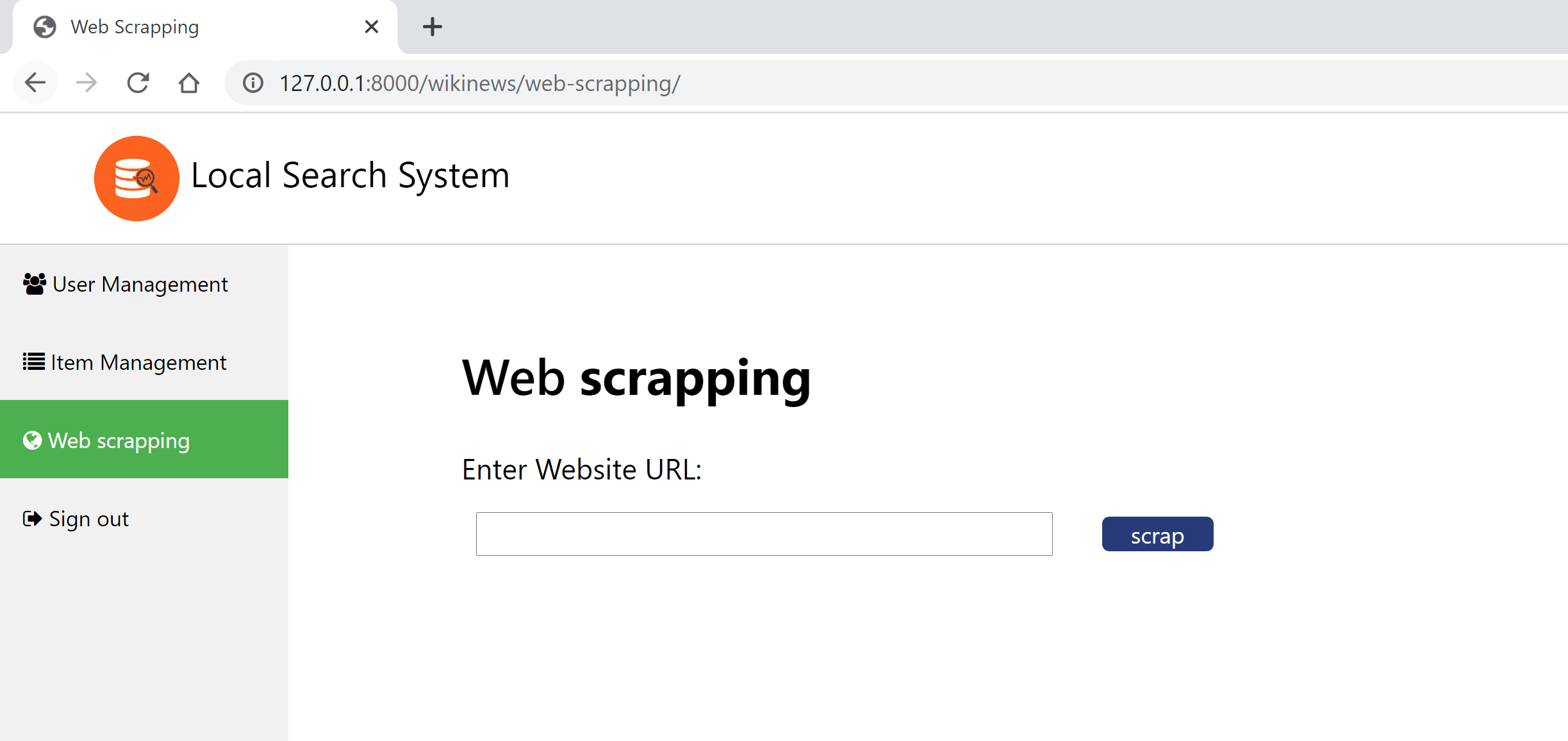


Admin can delete the user information by clicking on delete button from user details page. But the page asks for confirmation to delete the user as below because this action is not reversible once done. If admin clicks “yes”, then only user will be deleted in the WikinewsUser table in the database as well.



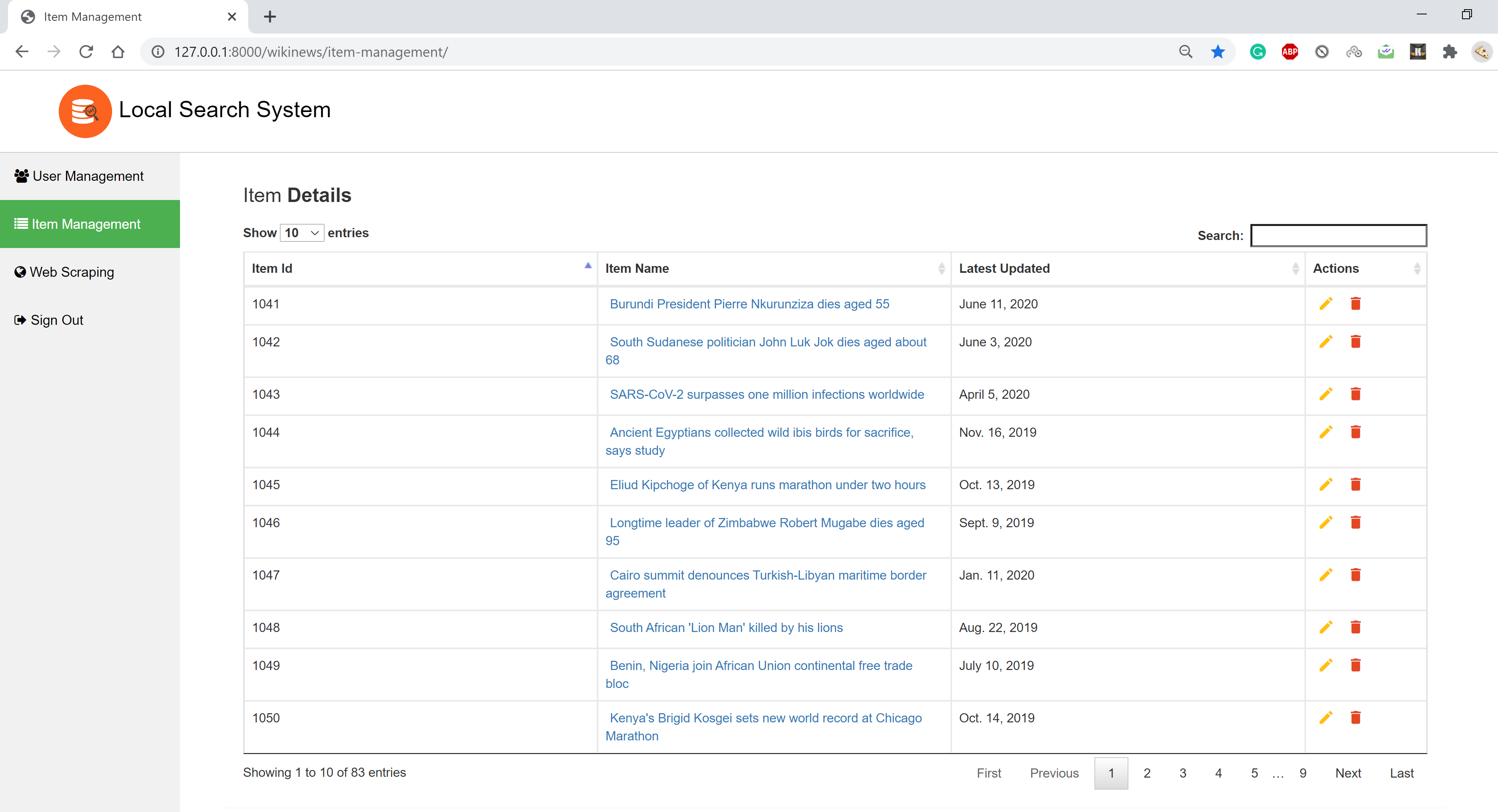
**Web Scrapping:**

Admin can enter the URL of website from which scrapping needs to be done. Wiki News website URL needs to be provided in the URL box. Once scrap button is clicked, it can be viewed in the backend that the data is being scrapped and stored in the local database.



**Item Management:**

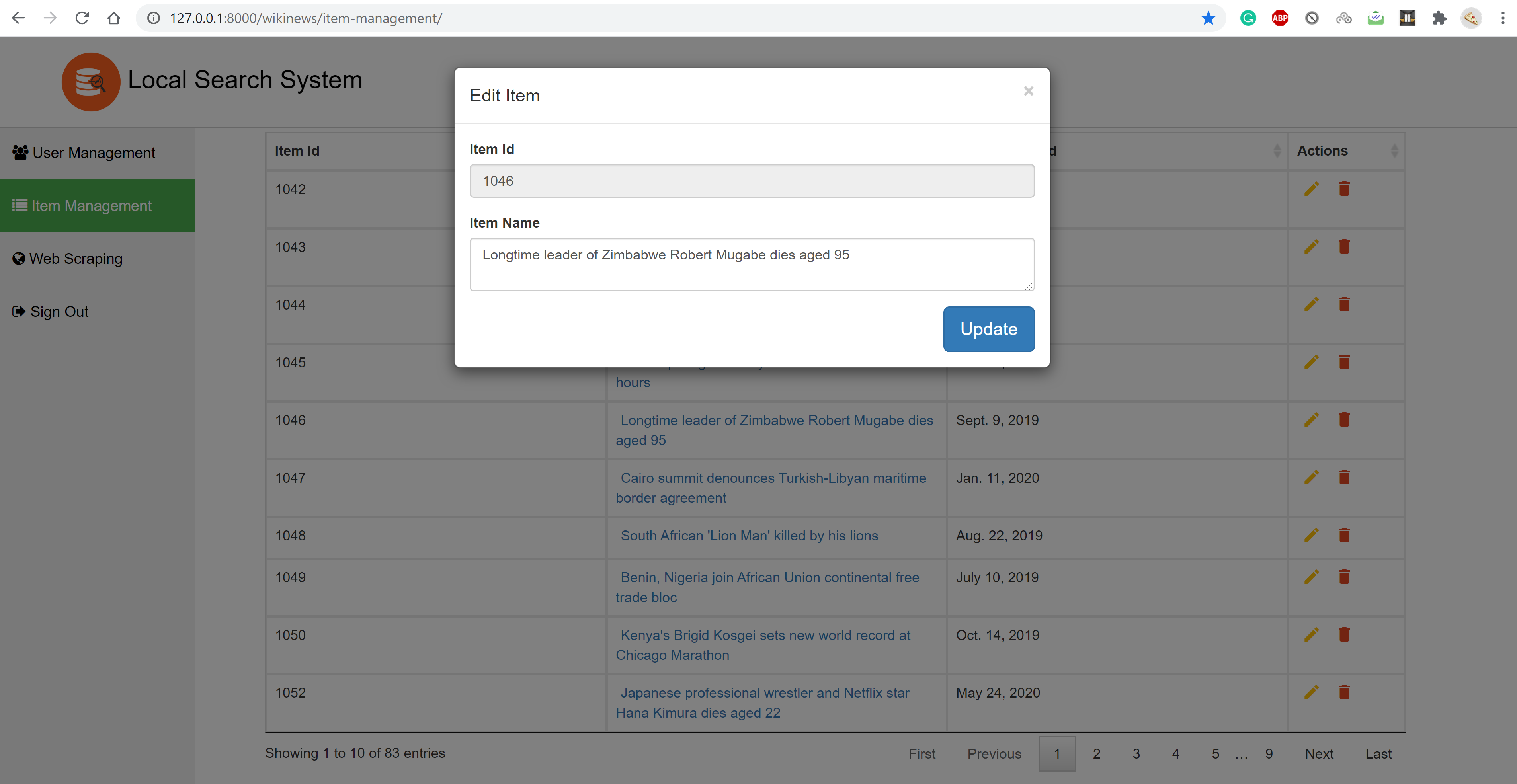
Admin can view the items in the database under “Item Management” tab. Once admin clicks on Item Management, below page appears which has Item details that are fetched from “WikinewsItem” table in database. These details in the table are obtained by scrapping the data that is performed by the admin. This page has pagination as only ten entries are displayed in a page and there are more than ten entries in this item details page. There is a search option in this page where admin can search for a particular item. Admin can view the item, edit and delete the item from this table.



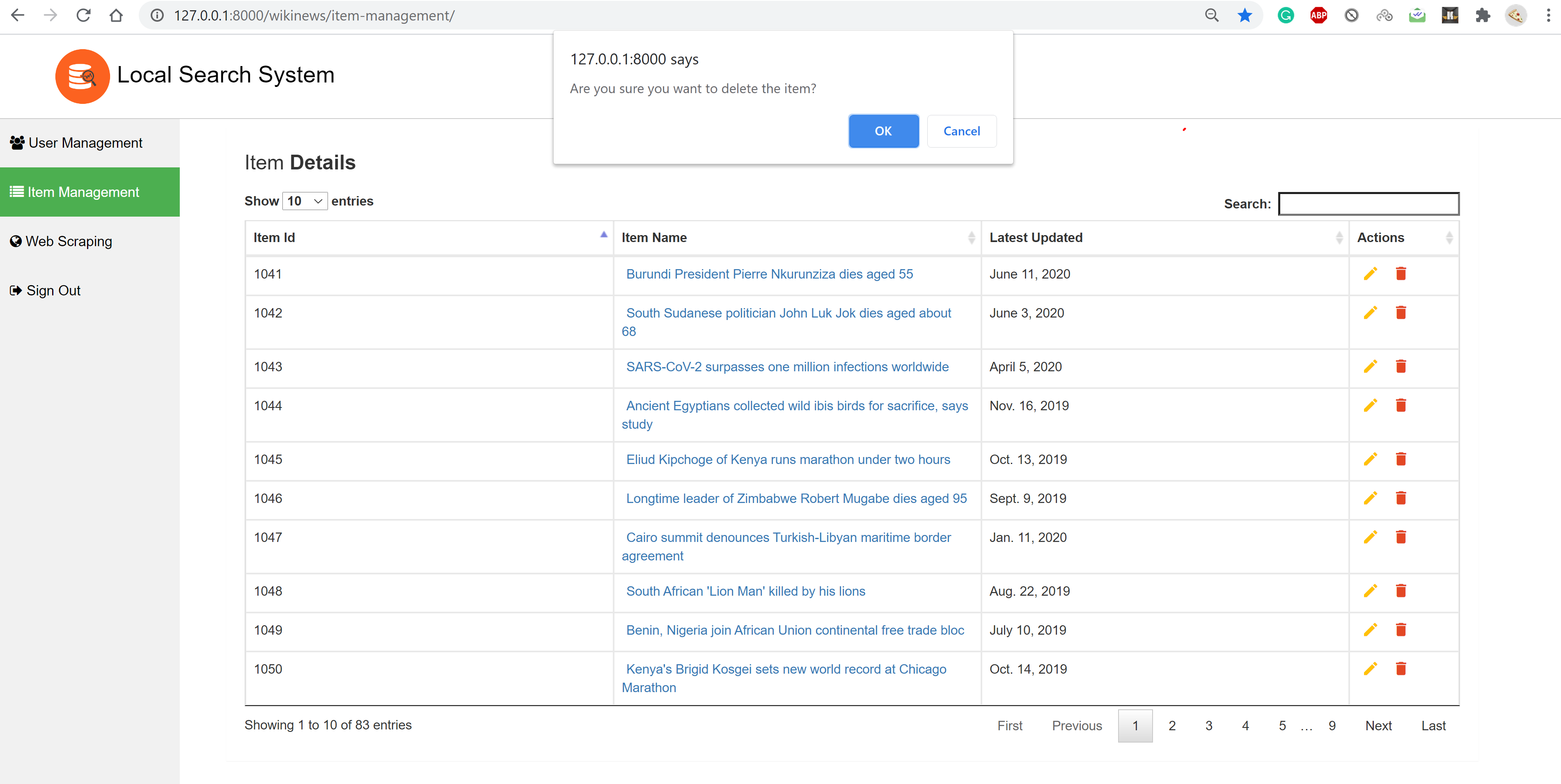
Below is the Sample page of the detailed page of item that appears once any of the Item Name is clicked from the Item details page. The image that appears in this page is the first image of that new article in the wiki news website.



Admin can edit the item details such as item id or the item name. When admin clicks on edit button from Item details page, a pop up appears as below. Once modified and clicked on update button, the details will be updated in the Item details table and WikinewsItem in the database.



Admin can delete any of the items by clicking on delete button and providing the confirmation. Below is the screenshot of pop up that appears once clicked on delete button for an item. The Item will be updated only when admin confirms as this action is not revertible. Once the item is deleted, it will be update in the database as well in the WikinewsItem table.

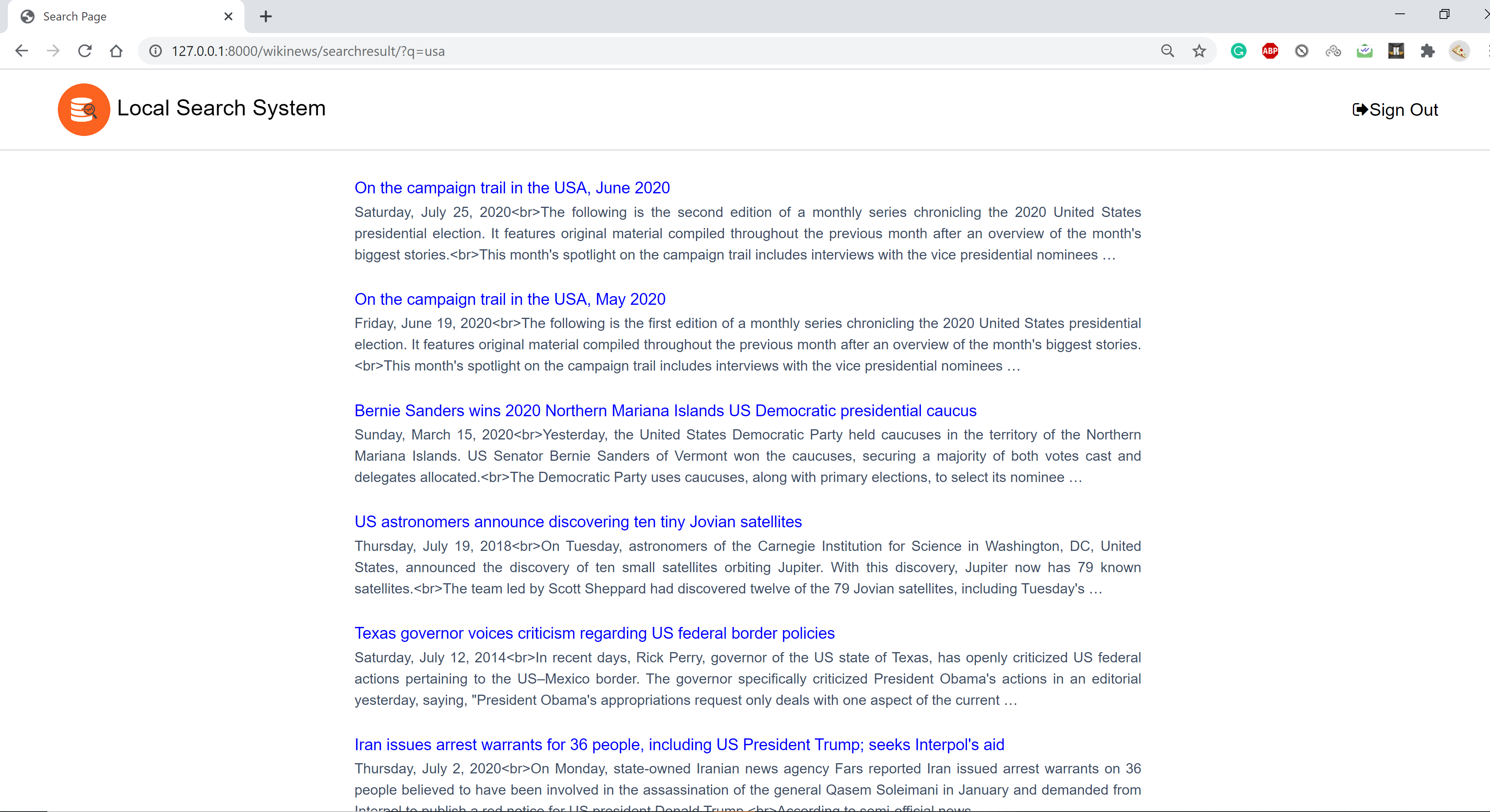


Admin can sign out any time by clicking on sign out button on the navigation bar of page.

**Indexing and Searching:**

Django Haystack is used as search algorithm which implements the indexing part in the backend. Elastic Search Engine is used in the backend.

Once anonymous user searches with any keyword in the search page, this word is matched against the content and title of the news items in the indexed document. Below is the sample search results page for a keyword “USA”.



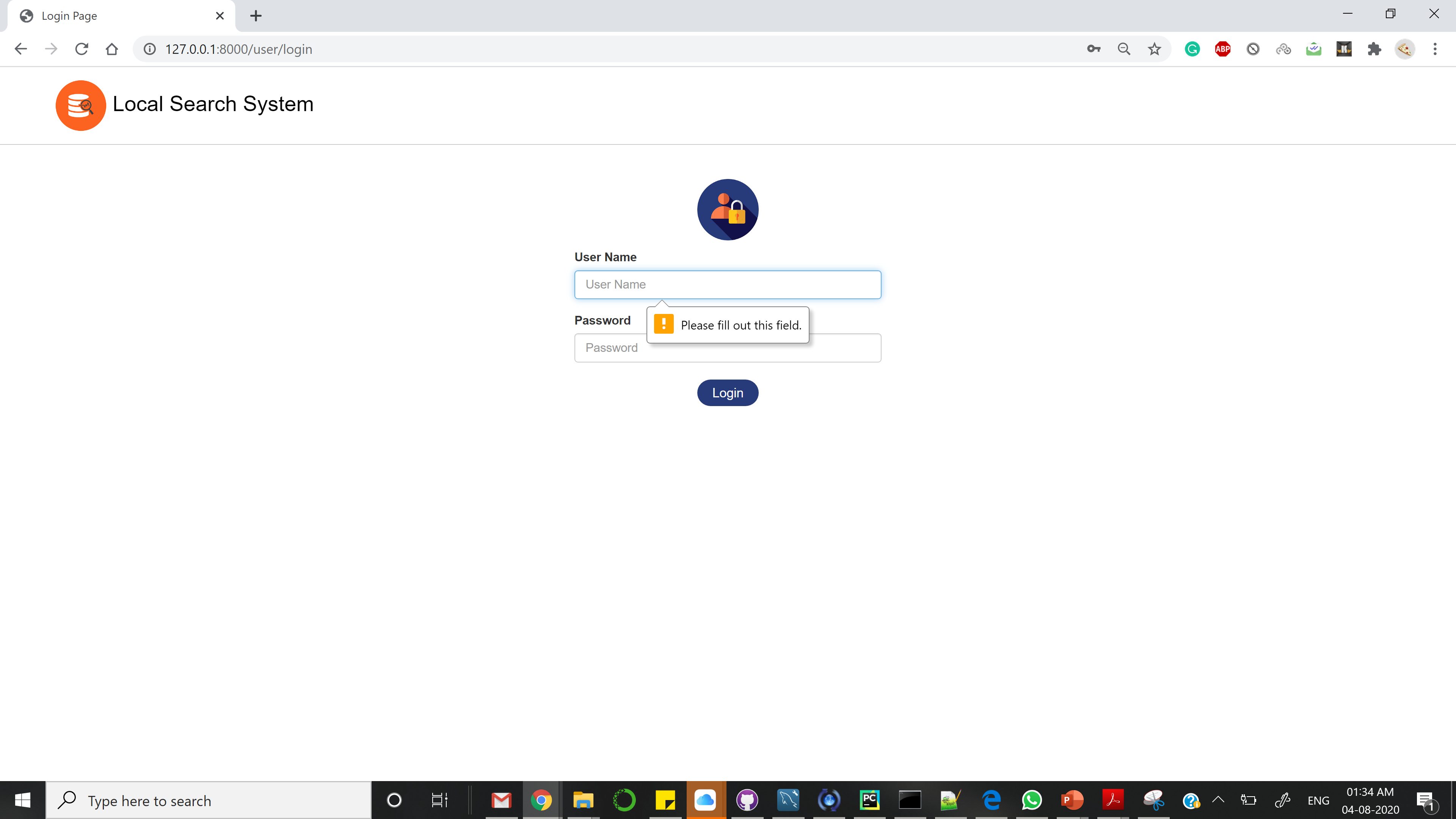
User can click on any of the links to get the detailed page of every news item and can sign-out by clicking the button on top right of the page.

**Validations:**

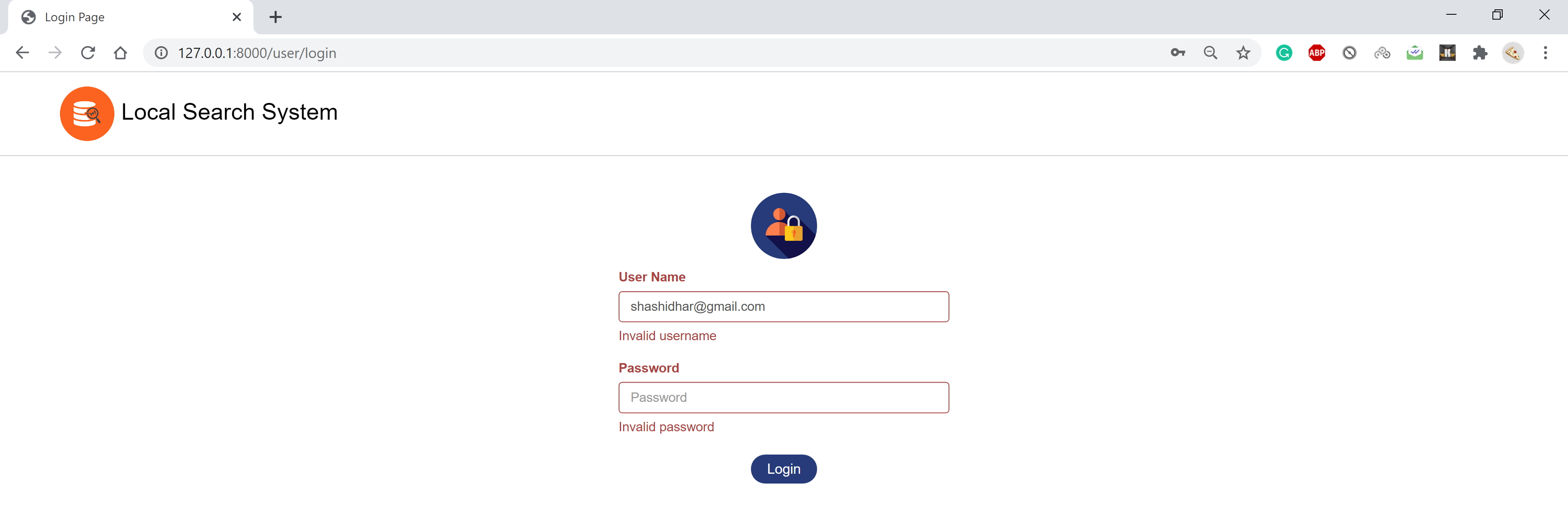
Every website requires validations to maintain the standards and some of the validations in the local search system are as follows.

* Django form class is used for validations.
* Different forms classes are developed for different pages like UserLoginForm and Newsroom.
* Handle the required field validations.
* Handle the database model validations through forms.

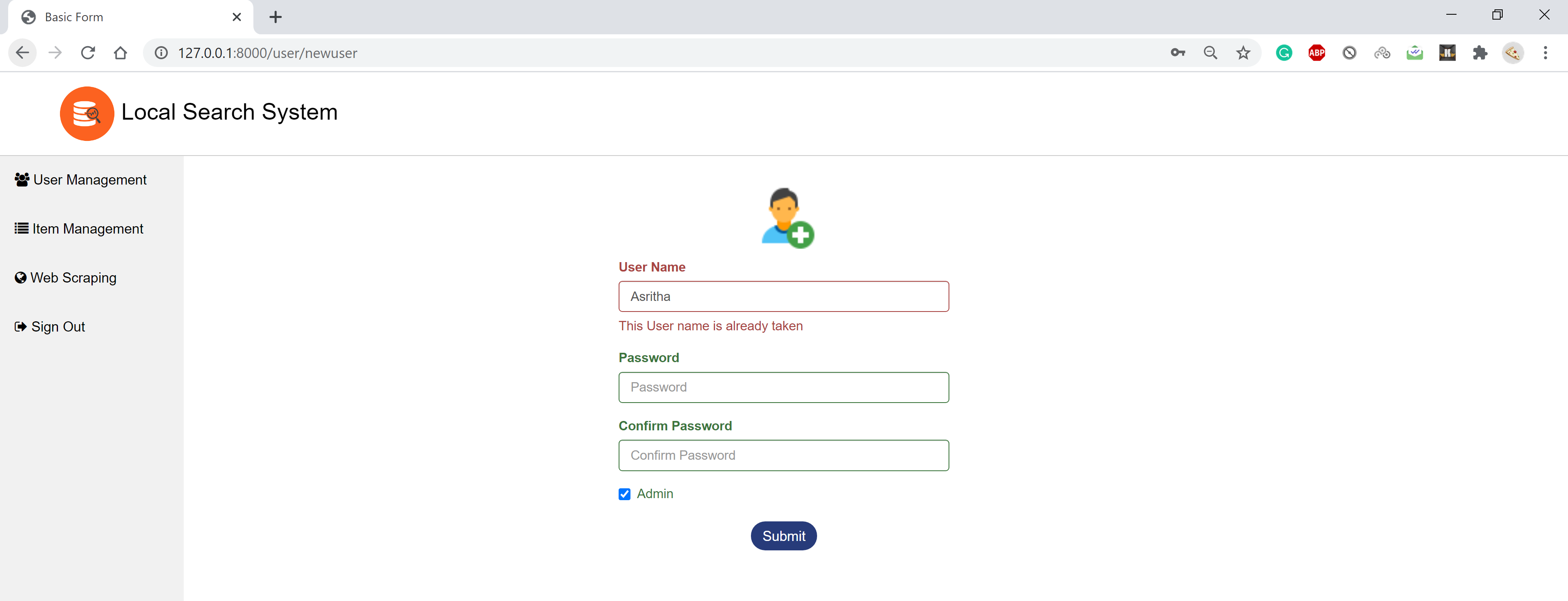
In the login page of the system, if user tries to login to the system with blank data the login fails saying “Please fill out the field” as below.



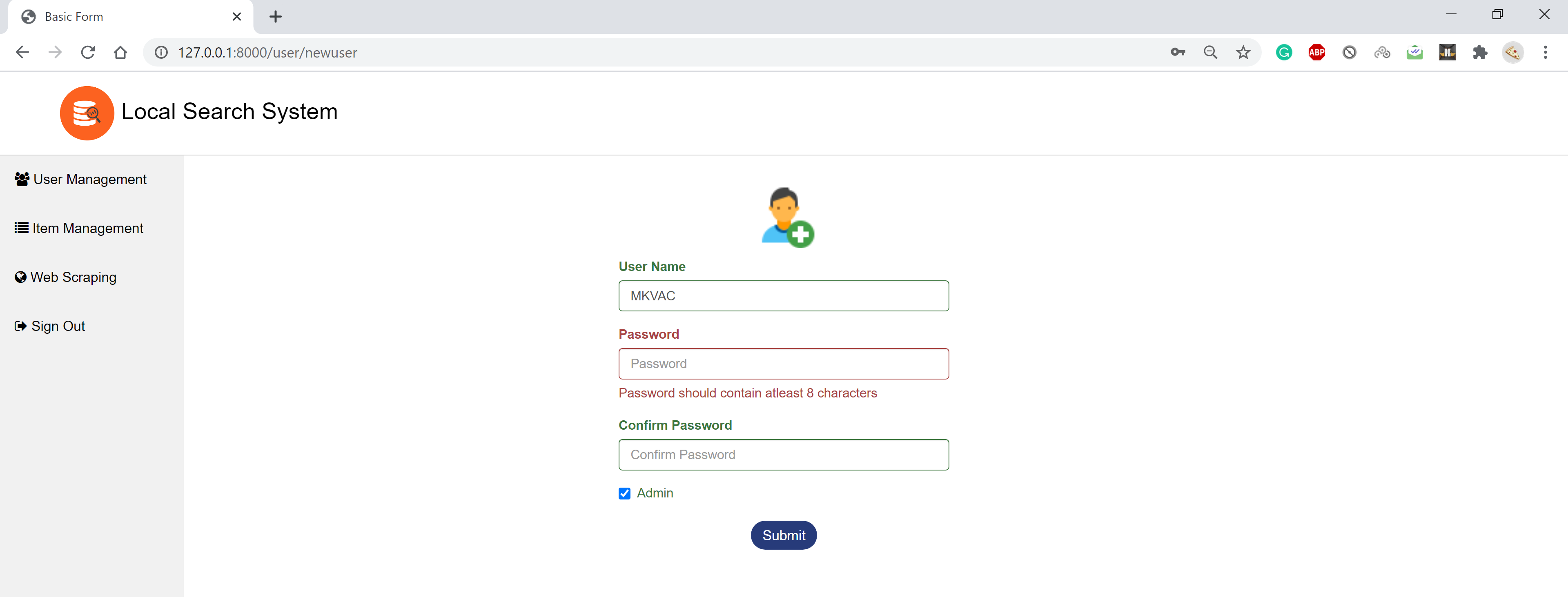
If user tries to login with incorrect details, validation occurs and displays “Invalid Username”/” Invalid Password”.



When creating a new user, if the admin tries to create new user with the existing user name there is a validation that says “This User name is already taken”.



There is a validation for password that tells at least eight characters are required as below.



Below is the validation for password and confirm password fields that validates both passwords should match. If doesn’t match, message displays saying, “Password and Confirm Password do not match”.

