King Saud University

College of Computer and Information Sciences

Department of Information Technology

IT 362 Course Project

Semester-1, 1447H

Phase 1 - logbook

Saudi food cuisine multilabel classification

Prepared by

Section No.	Name	ID
56594	Walah Alsaeed	444201689
	Norah Alshamsan	444200817
	Joud Alzahrani	444200862
	Norah Alkathiri	444200439
	Aljudi Altorkistani	444203157

Supervised by

Khulood Alyahya

King Saud University

College of Computer and Information Sciences

Department of Information Technology

<div align="center">

IT 362 Course Project

Semester-1, 1447H

</div>

Entry 1: Najd Village Menu

URL: https://najdvillage.com/menu/?lang=en

Date of Collection: September 25th, 2025

Check robots.txt:

```
User-agent: *
Disallow: /wp-admin/
Allow: /wp-admin/admin-ajax.php

Sitemap: https://najdvillage.com/sitemap.xml
Sitemap: https://najdvillage.com/sitemap.xml?lang=zh-hans
Sitemap: https://najdvillage.com/sitemap.xml?lang=en
```

Web scraping the menu URL is not disallowed according to robots.txt.

Tools and Methods:
- **requests and BeautifulSoup:** used to send GET requests and parse the response
- **csv and json:** used to export scraped data into structured files
- **datetime:** used to record the date the website was scraped
- **os:** used to create file systems
- **re:** used for cleaning strings such as image file names

The method included sending a GET request then parsing the response. Next, the relevant data was extracted for each food item and descriptions were cleaned initially by removing unnecessary words. Finally, the data was exported as CSV and JSON files.

Data Collected:
- Dish name
- Dish image
- Dish description
- Date scraped

Challenges:
- Unwanted sections: the page is divided into sections (e.g., main course, magloba). Certain sections, like drinks, are not relevant to our project.
- Offers section: some of the menu items are offers with multiple dishes in a single image.
- Missing data during full-page scraping: there were a lot of missing or misidentified data when an attempt to scrape the full page was made.
- Solution: I targeted the relevant sections individually, slightly altering the code to access and scrape each specific HTML block.

Processing:

As an initial step in processing the data scraped, an array of descriptive and non-essential words was removed from the item descriptions collected. Dish titles were sanitized and used as filenames for the saved images. The item's information was stored in a dictionary then appended to a general list of items, which was then converted into CSV and JSON formats.

King Saud University

College of Computer and Information Sciences

Department of Information Technology

IT 362 Course Project

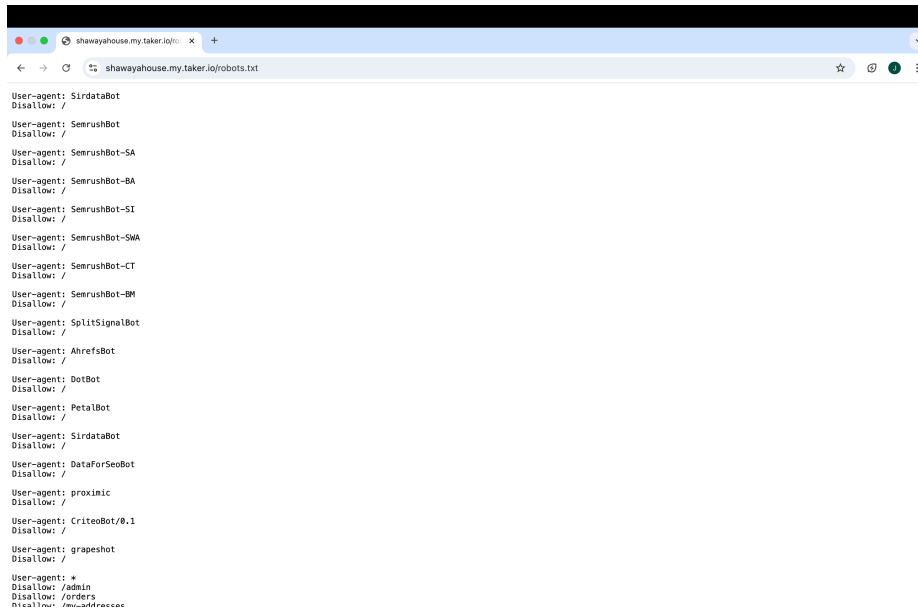Semester-1, 1447H

The final output consisted of:

- Najd_village_menu.csv and Najd_village_menu.json files that contain the name, image file path, classification, and scrape date of the item
- An images folder containg 45 images

Entry 2: Shawaya house Menu

URL: https://shawayahouse.my.taker.io/menu?language=ar

Date of Collection: September 22nd, 2025.

Check robots.txt:

The menu URL is not disallowed according to robots.txt, so scraping was permitted.

Tools and Methods:

We used **Python** as the main programming language and combined several tools:

IT 362 Course Project

Semester-1, 1447H

- **Playwright (headless Chromium)** — chosen instead of BeautifulSoup because the site is **dynamic** (items appear after rendering/scrolling). Playwright opens the page like a real browser and scrolls until all dishes load.
- **requests** — downloads the actual image bytes from each discovered image URL.
- **Pillow (PIL)** — inspects image bytes to detect a suitable file extension (jpg/png/webp) and ensures files are saved correctly.
- **nest_asyncio** — allows Playwright's asynchronous code to run cleanly in notebook environments (e.g., Colab).
    - **Workflow:** open the Arabic menu → auto-scroll to trigger lazy loading → parse dish cards (name + image) → label with regex rules.

Data Collected:

- Dish name (Arabic; normalized)
- Dish image (downloaded to local folder)
- Classification label (Rice, Chicken, Salad, Drinks, Dessert, etc.)
- Scrape date
- Dish URL

Challenges:

- Dynamic website (Playwright vs BeautifulSoup): Static parsers alone could not see all items because content loads dynamically.
- Solution: Used Playwright with headless Chromium to render and scroll automatically.
- Unclassified items: Some dishes didn't match the initial regex dictionary.
- Solution: Expanded the regex keywords (especially Arabic dessert/side terms) so previously unclassified dishes receive correct categories.

Processing:

After scraping, dish names were cleaned from diacritics and tatweel. Images were normalized to jpg/png format and renamed using slugified dish names. Classification labels were automatically assigned using regex. The final outputs were:

- **dishes.csv** and **dishes.json** containing name, image file path, classification, and scrape date.
- **images/** folder containing 37 saved dish images.

King Saud University
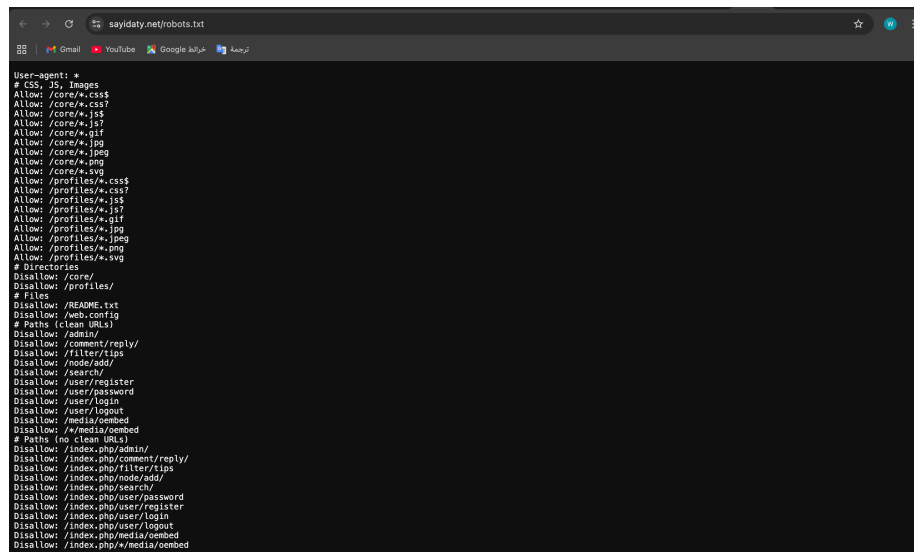
College of Computer and Information Sciences

Department of Information Technology

IT 362 Course Project

Semester-1, 1447H

Entry 3: kitchen Sayidaty

URL: https://kitchen.sayidaty.net/

Date of Collection: September 24-9- 2025.

Check robots.txt:

The screenshot shows the **robots.txt** file of the site. It allows crawlers to fetch images, CSS, and JavaScript, but it blocks private sections like login, register, search, and admin pages. This means the public recipes can be scraped, but the backend and user sections are off-limits.

Tools and Methods

To perform the web scraping of Saudi dishes from *kitchen.sayidaty.net*, a combination of Python libraries and custom logic was employed:

Tools (Libraries Used):

IT 362 Course Project

Semester-1, 1447H

- **Requests**: to send HTTP GET requests and retrieve the HTML content of webpages.
- **BeautifulSoup (bs4)**: to parse the HTML structure and extract relevant elements such as links, titles, ingredients, and images.
- **CSV & JSON**: to store the collected data in structured formats suitable for later analysis (CSV for tabular data, JSON for hierarchical storage).
- **Datetime & Time**: to record the exact scrape date/time for each dish and manage controlled delays between requests.
- **re (Regular Expressions)**: to clean and normalize Arabic text by removing diacritics, handling variations of Arabic letters, and filtering unwanted characters.

Methodology (Steps Taken):

- **Accessing robots.txt**: The site's robots.txt file was reviewed to confirm which sections of the website are permitted for crawling, ensuring compliance with allowed paths.
- **Link Extraction**: The scraper navigated through the *Saudi cuisine category pages* and collected all dish URLs by locating <a> tags containing /node/ in their paths.
- **Dish Details Extraction**: For each dish page:
  - o The **dish name** was obtained from header or title elements.
  - o The **main image** was collected from meta tags (og:image) or <img> elements.
  - o The **ingredients list** was parsed from the designated div.ingredients-area block, cleaned, and normalized using custom functions.
- **Arabic Text Normalization**: A custom function was implemented to standardize Arabic script, such as removing diacritical marks, normalizing alif/hamza forms, and replacing ta marbuta (ة) with ha (ه).
- **Data Cleaning**: Duplicate links and repeated ingredients were removed to maintain data consistency.
- **Data Storage**: The final dataset was exported to two formats:
  - o **CSV file** for easy tabular viewing and analysis.
  - o **JSON file** to preserve full structure and enable reuse in applications or further processing.
- **Ethical Scraping**: A 1-second delay between requests was introduced to avoid overloading the server.
-

Data Collected:

- Dish name (Arabic; normalized)
- Dish image (downloaded to local folder)

IT 362 Course Project

Semester-1, 1447H

- Classification label (Rice, Chicken, Salad, Drinks, Dessert, etc.)
- Dish URL (dish_url)
- Scrape date

Challenges

HTML inconsistency

- *Challenge:* Titles and images appeared in different places on the website.
- *Solution:* Different locations were checked so the correct title and image could always be found.

Ingredient extraction

- *Challenge:* Ingredients were too detailed (like "2 cups sugar" instead of just "sugar") and it doesn't the main components of the dish (for example, for chicken kabsa we expect just "rice" and "chicken").
- *Solution:* Extra details such as numbers, verbs, and quantities were removed, and ingredients were mapped to simple labels like *rice* or *chicken*.

Duplicate dishes

- *Challenge:* The same dish appeared more than once across pages.
- *Solution:* Repeated dishes were filtered out to keep only one copy of each.

Connection errors

- *Challenge:* Some pages failed to load, which could stop the process.
- *Solution:* Failed pages were skipped, and short pauses were added between requests to keep the scraping stable.

Processing

After scraping, ingredient text was normalized by removing diacritics, numbers, and extra symbols. Synonyms were mapped to simple food labels (e.g., رز، دجاج). Duplicate dishes and repeated ingredients were removed. Error handling and delays were applied to ensure stable scraping.

The final outputs were:

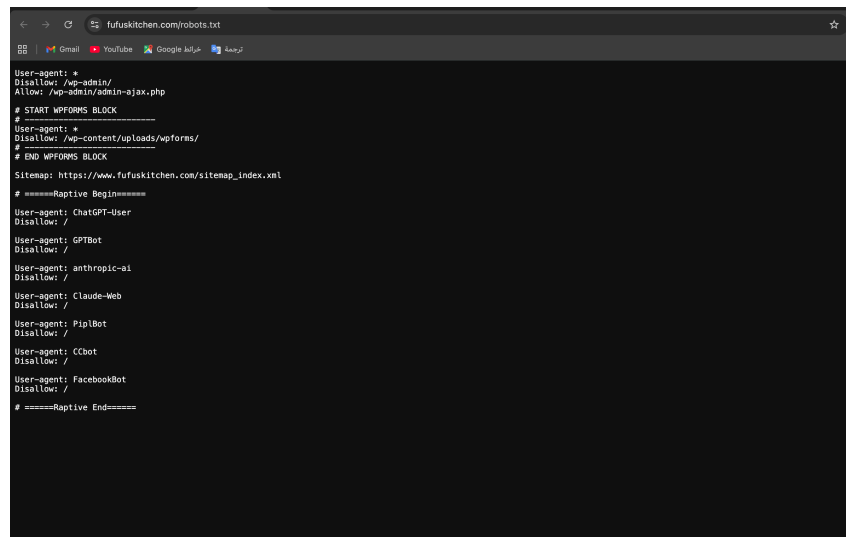# IT 362 Course Project

Semester-1, 1447H

- **saudi_dishes.csv** and **saudi_dishes.json** containing dish name, ingredient labels, image link, dish URL, and scrape date.

Entry 4: Fufu's Kitchen

Url: https://www.fufuskitchen.com/category/middle-eastern/

Date of Collection: September 24-9- 2025

Check robots.txt:

The robots.txt file of the website permits access to public content, including recipes, CSS, JavaScript, and images, but restricts sensitive paths such as admin pages, user authentication, and media embedding. This means that web scraping of recipe data (publicly available content) is allowed, while private or administrative sections remain off-limits.

Tools and Methods

Tools/Libraries:

- *Requests* – for sending HTTP requests and fetching HTML pages.
- *BeautifulSoup (bs4)* – for parsing HTML and extracting titles, images, and ingredients.
- *re (Regex)* – for cleaning text and matching ingredient patterns.

IT 362 Course Project

Semester-1, 1447H

- *Pandas* – for saving structured data into CSV format.
- *JSON* – for exporting data into a structured JSON file.
- *os* – for creating folders and saving images locally.
- *datetime & time* – for recording the scraping date and managing polite delays between requests.

Methods:

- Scraper navigates all pages in the **Middle Eastern** category.
- Collects recipe links from each page.
- For each recipe: extracts dish name, image, and raw ingredients.
- Ingredients are normalized using a predefined vocabulary, mapping detailed free-text into simple labels (e.g., "2 cups rice" → *rice*, "chopped onions" → *onion*).
- Images are downloaded and saved in an **images/** folder with filenames based on the dish name.
- Results are stored in **CSV** and **JSON** files.

Data Collected:

- Dish name (Arabic; normalized)
- Image url (downloaded to local folder)
- label (Rice, Chicken, Salad, Drinks, Dessert, etc.)
- Scrape date

Challenges

- **Ingredient detail**
  Challenge: Ingredients were very detailed with amounts and verbs (e.g., *"2 cups rice"*, *"1 tbsp chopped onion"*).
  Solution: A controlled vocabulary and regex cleaning reduced them into high-level labels like *rice, onion, chicken*.
- **Connection errors**
  Challenge: Some pages failed to load or returned errors.
  Solution: Error handling and polite crawl delays were added so the scraper continued running.

IT 362 Course Project

Semester-1, 1447H

Processing:

After scraping, text was cleaned by removing diacritics, numbers, and extra symbols. Ingredients were normalized into main labels (e.g., chicken, rice). Duplicate dishes were removed, errors were skipped, and delays were added for stable crawling. The final data was organized into dish name, labels, image file, and scrape date.

The final outputs were:

- Fufus_middle_eastern.csv and Fufus_middle_eastern.csv containing dish name, ingredient labels, image link, dish URL, and scrape date.
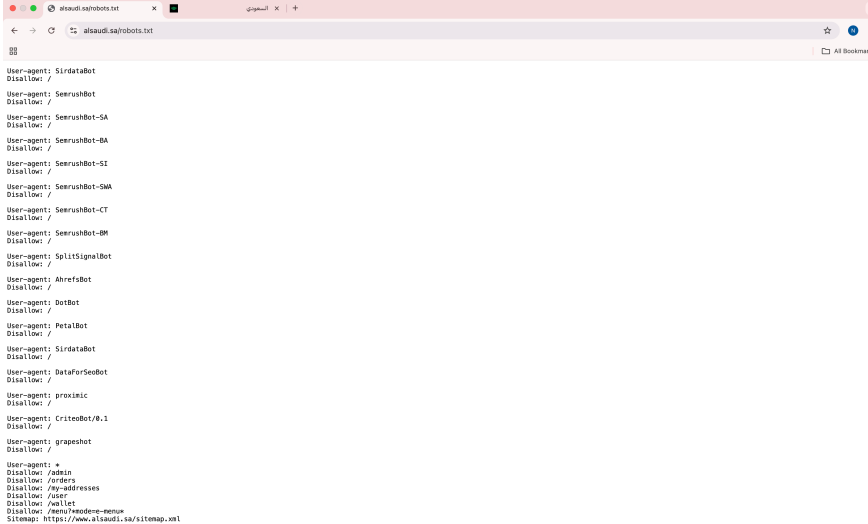
Entry 5: alsaudi menu.

URL: https://www.alsaudi.sa

Date of Collection: September 22nd, 2025

King Saud University

College of Computer and Information Sciences

Department of Information Technology

IT 362 Course Project

Semester-1, 1447H

Check robots.txt:

The alsaudi.sa robots.txt allows scraping of general menu content but blocks sensitive sections (admin, orders, user pages) and restricts specific commercial bots.

Tools and Methods:
- Playwright (headless Chromium) — chosen instead of BeautifulSoup because the site is dynamic; dishes only appear after JavaScript rendering and scrolling. Playwright opens the page like a real browser, executes scripts, and scrolls until all menu items are loaded.
- Requests (via Playwright context) — fetches the raw image bytes from each discovered image URL with correct headers.
- Pillow (PIL) — inspects image bytes, ensures the proper file extension (jpg/png/webp), and saves them into the images/ folder.
- Nest AsyncIO — enables Playwright's asynchronous scraping code to run smoothly in a notebook/Colab environment.

Data Collected:
- Dish name (in Arabic, with normalization to unify spellings)
- Dish image (downloaded and stored locally in images/)
- Classification label (Rice, Chicken, Meat, Dessert, etc.)
- Scrape date (YYYY-MM-DD format)
- Dish URL

IT 362 Course Project

Semester-1, 1447H

Challenges:
- Dynamic website: The website loads dishes dynamically, so items only appear after scrolling down.
- Arabic text variations: The text contained many spelling differences and diacritics, which made keyword detection harder.
- Category pages: The menu was split into category pages (e.g., Rice, Dessert, Meat), but some items were not consistently labeled.
- Unclassified items: Some dishes remained unclassified because they did not match any existing regex rules.

Solution:

To address these issues, we used Playwright to fully load dynamic content, applied Arabic text normalization, relied on category IDs as fallback labels, expanded regex rules with additional categories, and excluded irrelevant sections during preprocessing.

Processing:
As an initial step, the scraped data was cleaned by normalizing Arabic text and removing stopwords. Dishes were then classified using predefined keyword patterns with category IDs as fallback labels. Images were downloaded, renamed with safe filenames, and stored locally. Finally, the processed dataset was exported into structured alsaudi_menu.csv and alsaudi_menu.json (containing dish name, image file path, classification, and scrape date), along with an images/ folder storing all downloaded dish images.