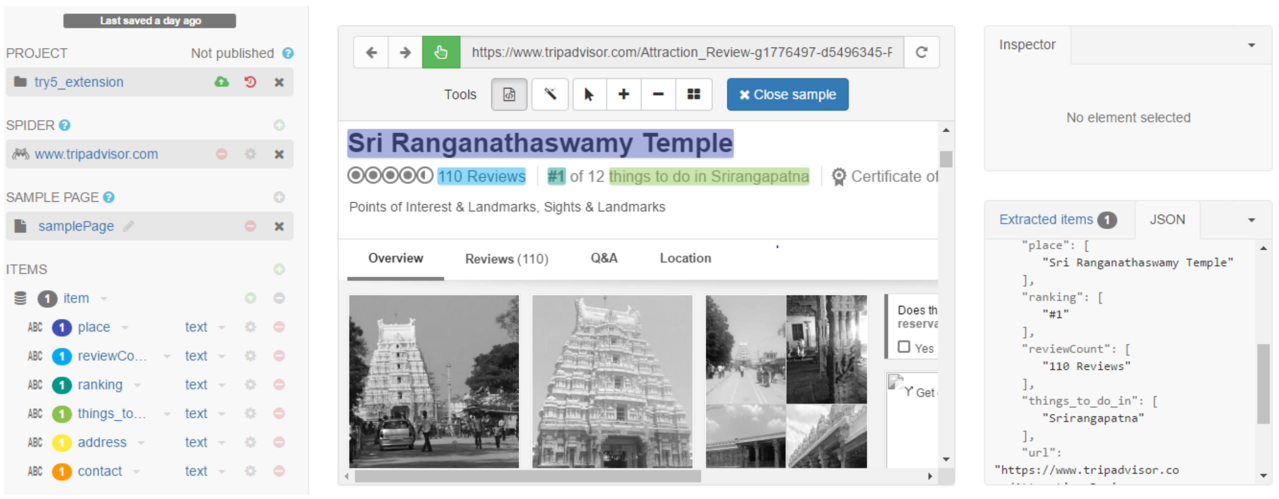
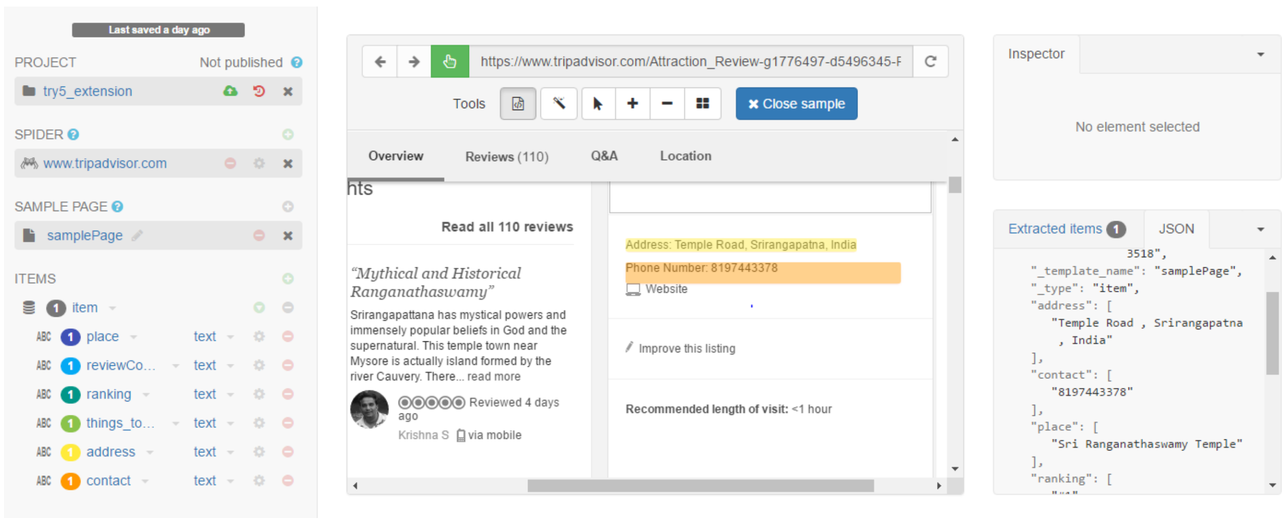
Website: https://www.tripadvisor.com/
An American travel website company providing reviews of travel-related content.

Field	Description
attraction_in (things_to_do_in)	Denotes the place of interest where different attractions can be found. (**things to do in Srirangapatna**)
noOfReviews (reviewCount)	No of reviews for the attraction (**110 Reviews**)
url	url of page being scraped
attraction (place)	Denotes one of the attractions (**Sri Ranganathaswamy Temple**)
address	Location of the attraction (**Address: Temple Road, Srirangapatna, India**)
contact	Point of contact for the attraction (**Phone Number: 8197443378**)
rank(ranking)	Rank for the attraction at the place of interest (**#1**)

NOTE : field names in (parenthesis), were used while annotating fields in portia, which were later mapped to field names using postCleanup.py.



```
    "place": [
        "Sri Ranganathaswamy Temple"
    ],
    "ranking": [
        "#1"
    ],
    "reviewCount": [
        "110 Reviews"
    ],
    "things_to_do_in": [
        "Srirangapatna"
    ],
    "url":
"https://www.tripadvisor.co
```



Read all 110 reviews

"Mythical and Historical Ranganathaswamy"
Srirangapattana has mystical powers and immensely popular beliefs in God and the supernatural. This temple town near Mysore is actually island formed by the river Cauvery. There... read more
Reviewed 4 days ago
Krishna S via mobile

Address: Temple Road, Srirangapatna, India
Phone Number: 8197443378
Website
Improve this listing
Recommended length of visit: <1 hour

```
                3518",
    "_template_name": "samplePage",
    "_type": "item",
    "address": [
        "Temple Road , Srirangapatna
         , India"
    ],
    "contact": [
        "8197443378"
    ],
    "place": [
        "Sri Ranganathaswamy Temple"
    ],
    "ranking": [
```

Portia - A visual web scraping tool was used to extract data
https://github.com/scrapinghub/portia

Reasons for choosing the tool:

1) Ease of Use: It's web-based UI helps us to choose the data we want to extract by annotating using mouse-clicks on elements of a webpage.
2) Handles badly formatted HTML : Portia uses Scrapely behind the scene to extract structured data from HTML pages. Other Wrapper Induction libraries work by building a DOM tree based on the HTML source, then use the given CSS or XPath selectors to find matches in that tree. Scrapely, treats a HTML page as a stream of tokens, hence Portia will be able to handle any type of HTML no matter how badly formatted.
3) Scrapely relies on the order of tags on a page and doesn't need to find a 100% match between unannotated and sample page, instead it looks for the best match. Thus even if the page is updated and tags are changed, Scrapely can still extract the data.More importantly, Scrapely will not extract the information if the match isn't similar enough. This approach helps to reduce false positives.
4) Its implemented using Ide (Instance-based Data Extraction) approach, which outperforms FETCH system, the commercial version of the state-of-the-art research system Stalker as stated by Zhai et al in "Extracting Web Data Using Instance-Based Learning"

Wrapper Description:

As cited in paper Extracting Web Data Using Instance-Based Learning, by Yanhong Zhai and Bing Liu, a major problem with inductive learning is that the initial set of labeled training pages may not be fully representative of the templates of all other pages. For pages that follow templates not covered by the labeled pages, learnt rules will perform poorly. The usual solution to this problem is to label more pages because more pages should cover more templates.

Scrapely however, employs a learner wrapper, based upon the Instance Based Learning algorithm and the matched items are combined into complex objects using a tree of parsers, inspired by A Hierarchical Approach to Wrapper Induction.

Wrapper Model: To treat each page to be labeled as a sequence of tokens, where a token can be any HTML element, a HTML tag, a word, a punctuation mark, etc.

Technique:

1. A random page is selected for labeling.
2. The user labels/marks the items of interest in the page.
3. A sequence of consecutive tags (also called tokens later) before each labeled item (called the prefix string of the item) and a sequence of consecutive tags after the labeled item (called the suffix string of the item) are stored.
4. The system then starts to extract items from new pages. For a new page d, the system compares the stored prefix and suffix strings with the tag stream of page d to extract each item. If some target items from d cannot be identified (i.e., this page may follow a different template), page d is passed to step 2 for labeling.