|  |
| --- |
| Google Reviews Scraper  Deliverable 3 |

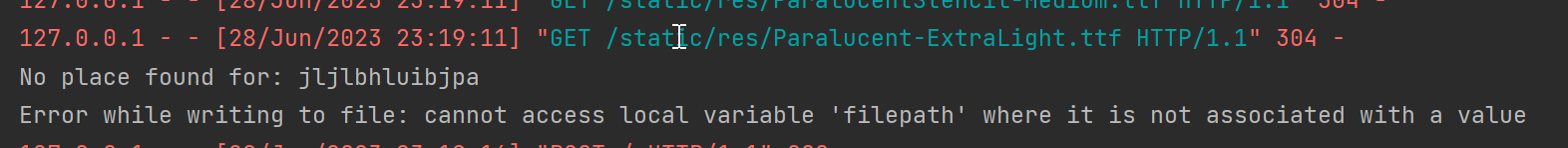
|  |  |  |
| --- | --- | --- |
| Group 1 |  |  |
| Monique Dai/ Chester Rae de Vera/ Heping Song  Snehalata / Hua Zhang / Yateng Geng / Chester Rae de Vera/ Heping Song |  |  |



# 1.UPDATES

1. Exceptions

a. Exception 1: place doesn’t exist on google map

****

**A screenshot of a google reviews scraper

Description automatically generated with medium confidence**

b. Exception 2: the place exist, but there are no views for it

**A picture containing text, screenshot, font

Description automatically generated**

**A screenshot of a computer

Description automatically generated with medium confidence**

1. Scraping logic

By using selenium and BeautifulSoup properly, now we could scrape all reviews instead of only 5.

A screenshot of a computer

Description automatically generated with low confidence

1. Scraping process showcasing in python

**A screen shot of a computer

Description automatically generated with medium confidence**

1. Front-end layout

We have decided the elements that we want to show for users on front-end. We have those front-end template layouts: home.html, account.html, register.html, login.html, etc.

1. Register and login functions

**A picture containing text, font, line, screenshot

Description automatically generated**

User now could register and login to our system to use the scraper.

1. Database and data output

User’s register information would be saved in users.sqlite. And scraping output data would be saved in the output\_data folder as csv files.

A screenshot of a computer

Description automatically generated with medium confidence

**A screen shot of a computer

Description automatically generated with medium confidence**

# 2. Plans for the rest of the project

1. Testing and debugging procedures to ensure a robust and error-free application.
2. Enhancing the front-end design and optimizing the user experience to achieve a more polished and professional appearance.
3. Addressing the current limitation of the scraper, specifically its inability to efficiently handle a substantial volume of reviews, such as those exceeding 8000. This necessitates investigating whether this limitation stems from Google Maps' scrolling logic or if alternative approaches can be devised to overcome this obstacle.
4. User guide.

# 3. set up

1. Replace **'Your\_API\_Key'** with your real google API key for testing in all python files.
2. Install Firefox brower, and the matched **geckodriver** from <https://github.com/mozilla/geckodriver/releases>. You could use **test\_selenium.py** to check if you installed a matched version. If so when you run it, you will see it open a Firefox browser window on google (it will be closed quickly which is expected). Also set path to **geckodriver** as per your configuration, change it to your path accordingly in app.py line 222.
3. Set the configuration and install all necessary packages.
4. Run **app.py**, register and login, then input the place you want to scrape reviews for. You could check the scraping process in python console.

Please check our repo for more details: **https://github.com/raeDV/Google-Reviews-Scraper**