Hacking 101

Hacking for Humanity, July 8, 2017

Meggie Mahnken and Bonnie Schulkin

Hackbright Academy

Hacking

You might be imagining something like this...

Nowadays, closer to this...

Hacker Ethic

- Access
- Freedom of information
- Improvement to quality of life

Source: Steven Levy, *Hackers: Heroes of the Computer Revolution*

The Basics

So you came to a hackathon...

What now?

- Figure out a problem to solve
- Choose a data source
- Decide on technology
- Draw the user flow
  - Think big, start small
- Break up into tasks and distribute
  - New to hacking? Pair program!
- Code!

Example Applications

- **Nearest Food Shelter** using Web Scraping
- **Confidence on Demand** using Twilio API
- https://github.com/flyrightsister/h4h_2017 <https://github.com/flyrightsister/h4h_2017>

Nearest Food Shelter: Overview

- User can input their address, find nearest SF food shelter
- Python / Flask app

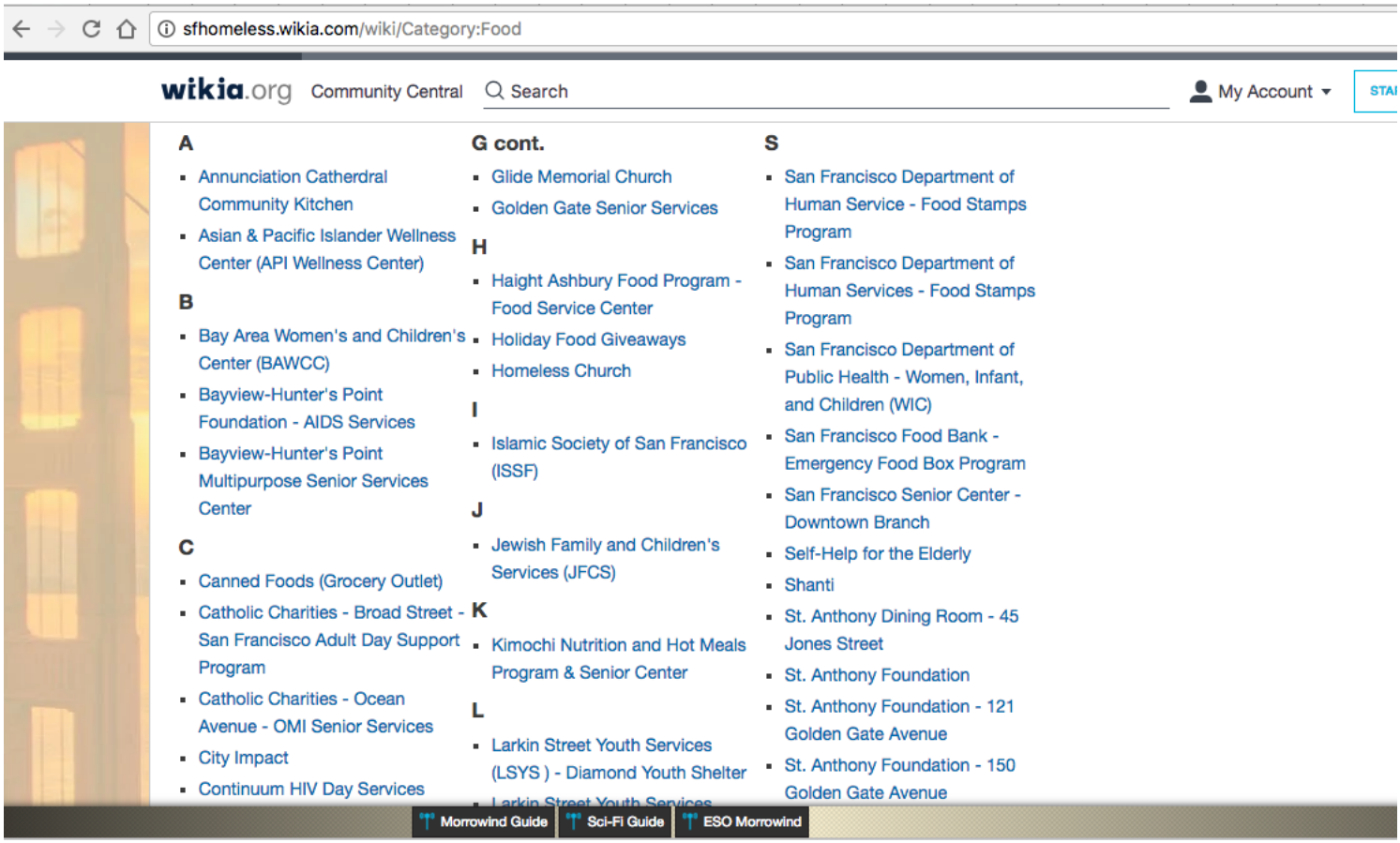
Overall Process

- **Web scraping** to get a list of SF Homeless shelters
  - Use the **Google Places API** to get addresses for each shelter
- User provides their address via a **web form**

- **Google Distance Matrix API** to find nearest shelter

Web Scraping

- Programmatically collect data from web pages
- Use cases
  - Avoiding lots of copying/pasting
  - Seeding a database with real data
  - Collecting lots of data from same web page over time
- Warning: Terms of Use, Service Agreements, etc.
  - Good resource: To Scrape or Not To Scrape <http://www.storybench.org/to-scrape-or-not-to-scrape-the-technical-and-ethical-challenges-of-collecting-data-off-the-web/>



- Install `requests` package to make web requests in Python
- Install `BeautifulSoup` package to parse an HTML webpage

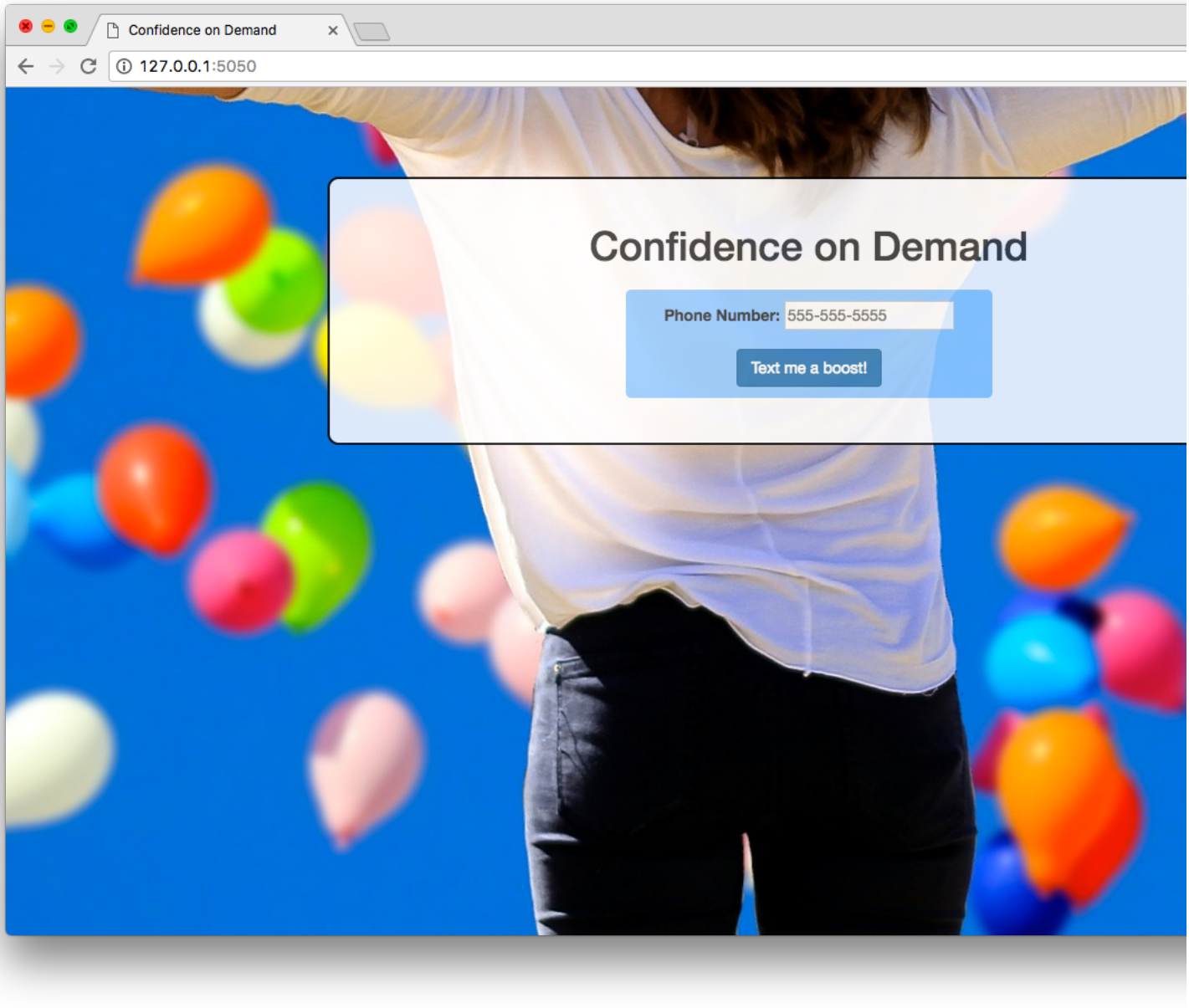
```
def get_soup(url):
  """Request webpage and return a Beautiful Soup object."""

  resp = requests.get(url)
  soup = BeautifulSoup(resp.text)
  return soup

def get_resource_names(webpage_soup):
    """Get food urls from a homelessness resources webpage."""

    holding_div = webpage_soup.find('div', {'class':'mw-content-ltr'})
    all_food_links = holding_div.findAll('a')
    names = [ a['title'] for a in all_food_links ]
    return names
```

Confidence on Demand: Overview



- Send confidence-boosting SMS messages via Twilio API
- Python / Flask app
- Simple HTML / CSS front end

The Background

- Data: google search ***confidence boosting quotes***
  - Copy from websites and store in Python list
- User flow: User clicks button and receives text

The Flask App

- One template:
  - home page with form / button
- Two routes:
  - display home page
  - process form

The Twilio API

- Get Twilio account for credentials

- https://www.twilio.com/try-twilio <https://www.twilio.com/try-twilio>

- Use Python wrapper
  - https://github.com/twilio/twilio-python <https://github.com/twilio/twilio-python>

- Calling Twilio:

```
from twilio.rest import Client
client = Client()

...

message = client.messages.create(
    to=phone,
    from_=FROM_PHONE,
    body=msg)
```

Now it's Your Turn!