INTERNSHIP REPORT

Data Science in the Real World

PI Prescott (2020).

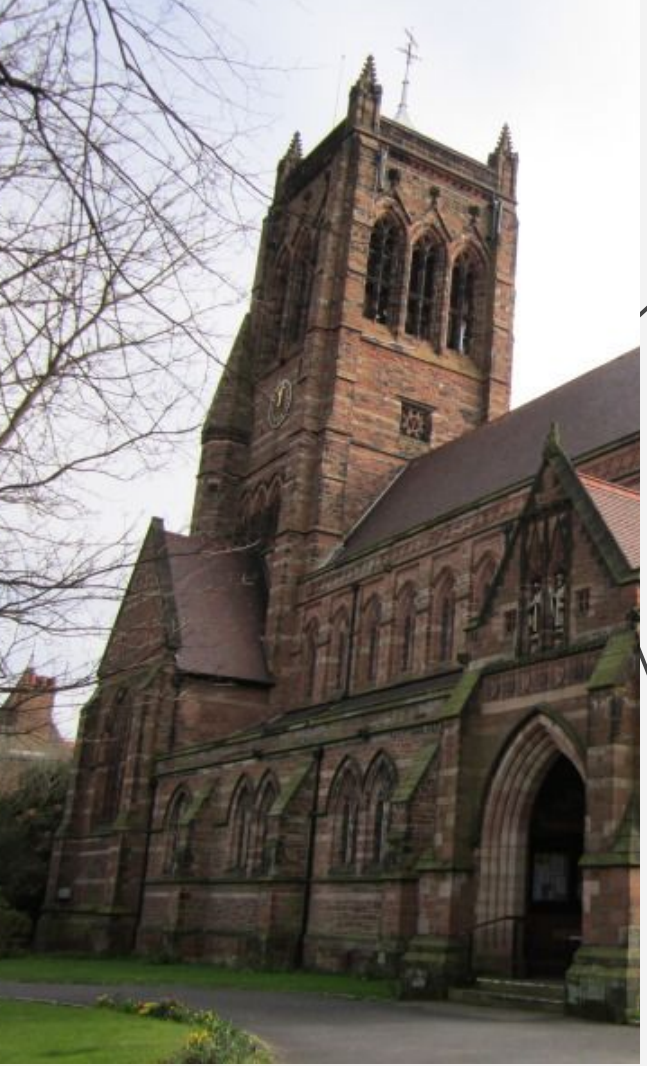
THE TASK

The student will work on a research project defined between the student and an external organisation.

The aims of the placement will be defined in terms of *progressive risks in effecting a solution*:

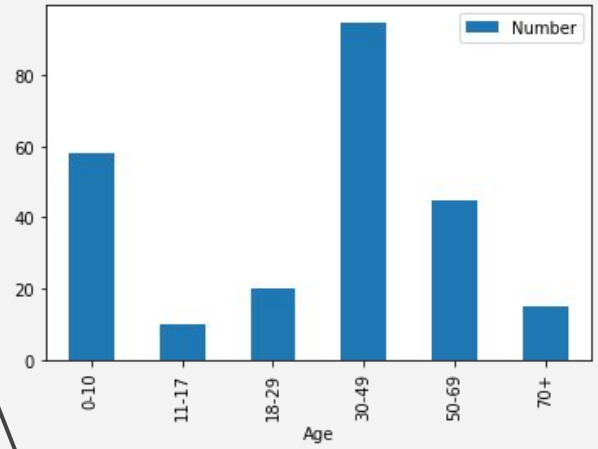
- The **first aim** should have a *low risk of failure*;
- the **second aim** will be *more challenging but capable of solution given initiative* and energy on the part of the student;
- and the **third aim** can have a 'blue skies' element, *a real research challenge* and consequently a high risk of failure but success will demonstrate exceptional competence and initiative.

https://vital.liv.ac.uk/webapps/UoL-uol-module-overview-BBLEARN/module/view.jsp?course_id=_861125_1&mode=view



THE PARTNER

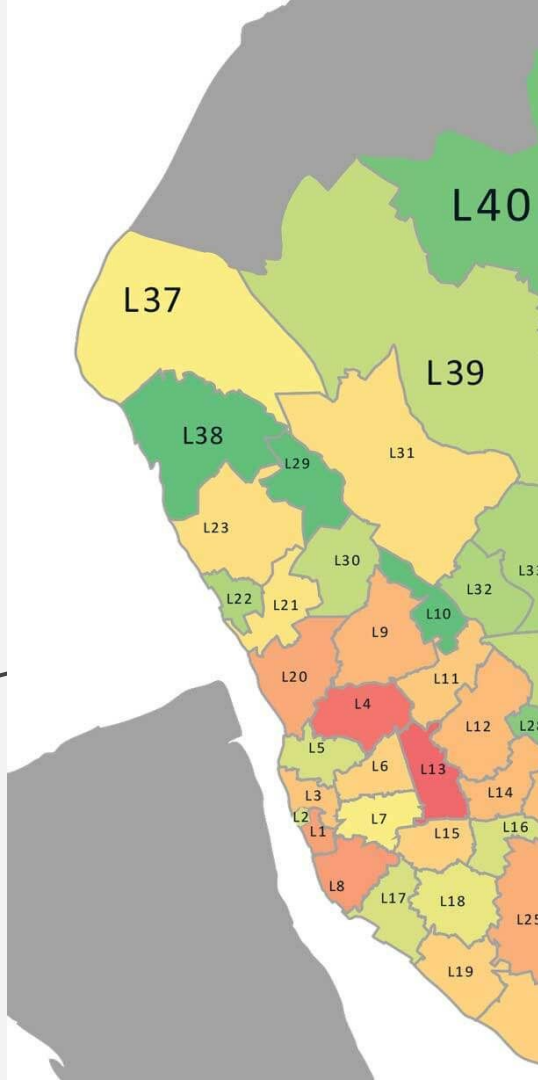
Mossley Hill Church
A Community Of Blessing



Led by the Rev Alan Kennedy.
Attended by Prof Alex Singleton.
A family church in south Liverpool.
An Anglican parish church,
with noteable Victorian architecture.

RESEARCH PROJECT

To transform the church members' postcode data into useful insight.



The Three Aims:

Neighbourhood Geodemographics

The analysis of people by where they live.

Catchment Area Analysis

The distance people will come for a service.

Developing a Data Dashboard

Individualized, instantaneous, interactive informational insight.

NEIGHBOURHOOD GEODEMOGRAPHICS

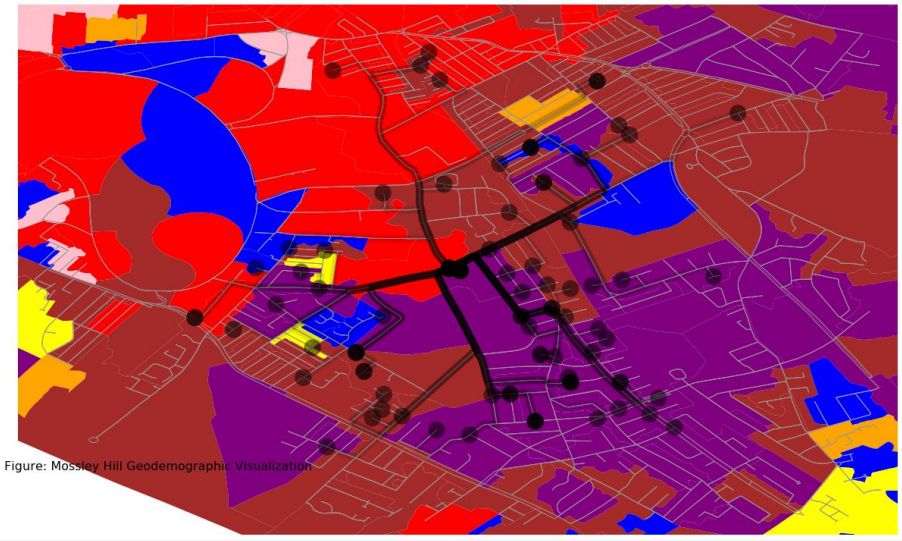
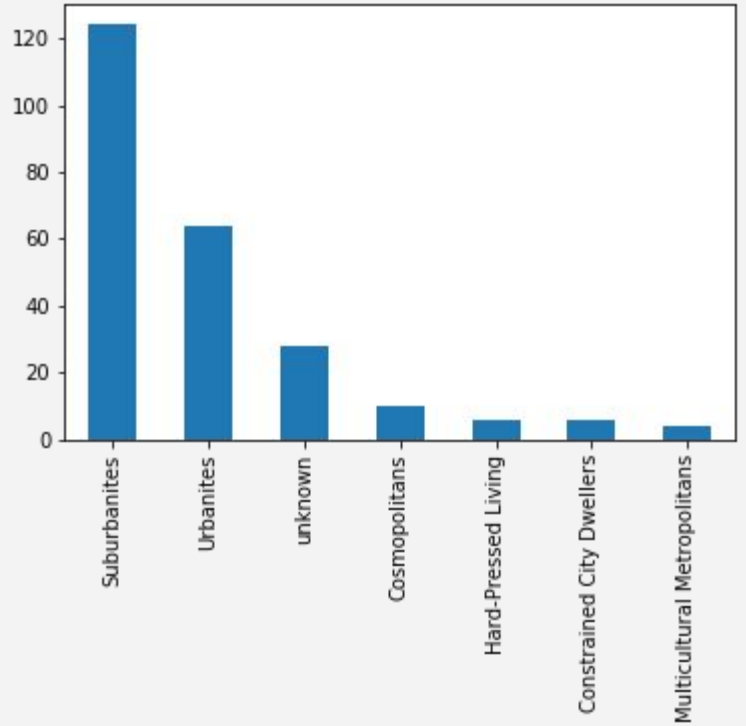
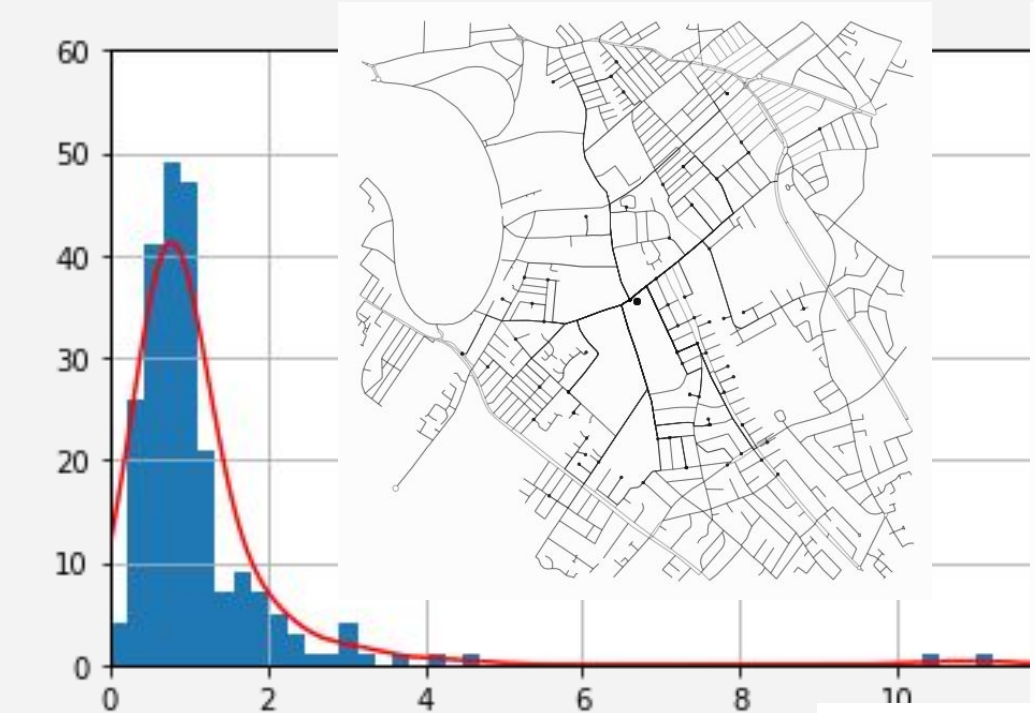


Figure: Mossley Hill Geodemographic Visualization



Analysis using the free and open-source 2011 Output Area Classification

CATCHMENT AREA ANALYSIS



Analysis used Geoff Boeing's OSMnx Python package.

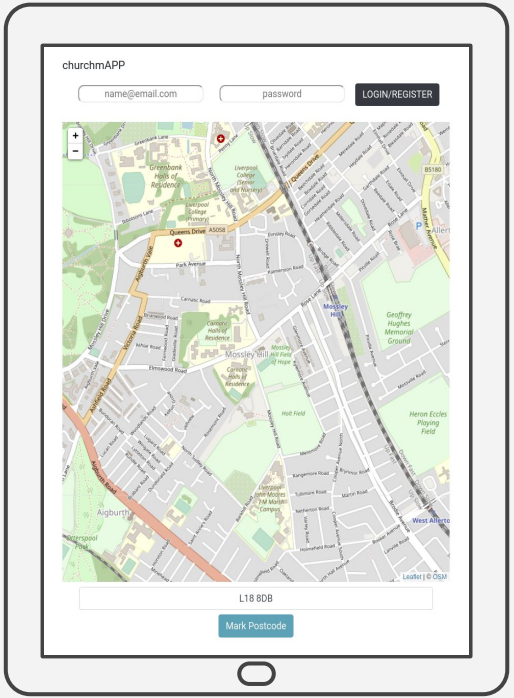
```
1  import osmnx as ox
2  ox.plot_graph(ox.graph_from_place('
```

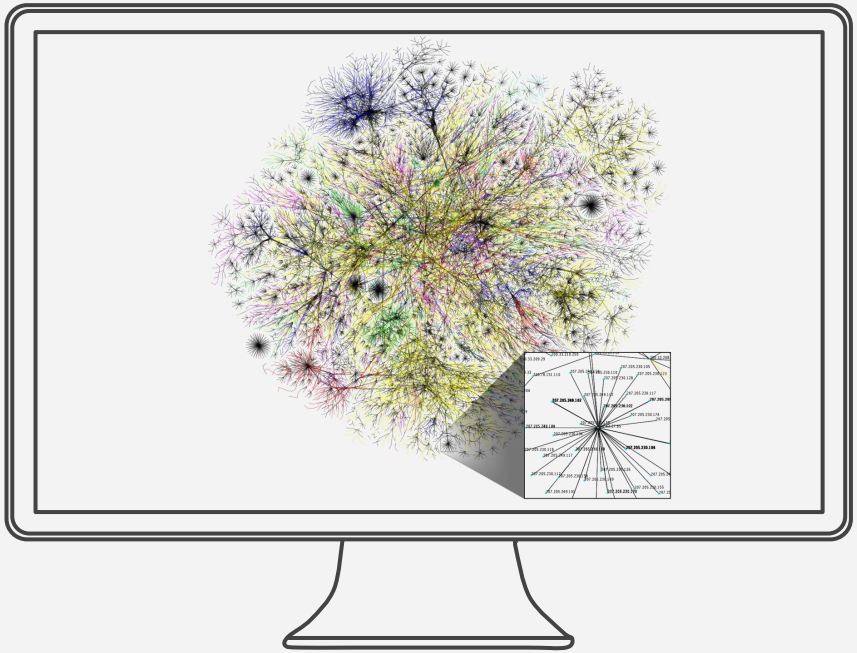
https://churchmAPP.netlify.com

DEVELOPING A DATA DASHBOARD

__churchmAPP__

a Full-Stack
Modern Web App





What is the internet?

-750,000 miles of cable

-Internet (computers) vs. WWW (documents)

-Hypertext: HTTP(S) and HTML

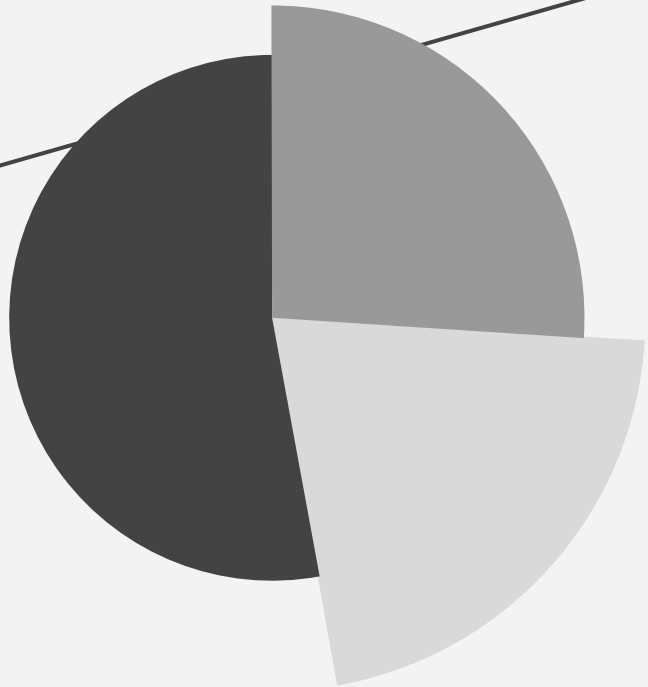
-Status Codes: 404, 500, (418!)

-HTML, CSS, JavaScript

-Front-end, Back-end, APIs

“The internet is a series of tubes.” (Senator Ted Stevens)

Modern Web Frameworks



Flask (Python Web Server)

SQLAlchemy (SQL ORM)

Flask-Restful (ReST API)

Bootstrap (CSS)

Vue (Javascript DOM)

```python
### user registration/authentication
@app.route('/auth', methods=['POST'])
def authenticate():
    try:
        email = request.json['email']
        password = request.json['password']

        user = session.query(User)\
                    .filter(User.email==email)\
                    .first()
        if user: # if user already exists...
            # ... authenticate password
            if not pbkdf2_sha256\
                    .verify(
                        password,
                        user.password_hash
                        ):
                return jsonify(
                    {'message':'Wrong password for that email'}
                    ), 401
            else:
                message = 'Authentication successful'

        else: # else create new password hash and user
            hash = pbkdf2_sha256.hash(password)
            user = User(email=email, password_hash=hash)
            session.add(user)
            session.commit()
            message = 'Registration successful'

        # everyone still here gets a token
        payload = { 'email': email }
        token = jwt.generate_jwt(
            payload, JWT_security_key,
            'PS256', timedelta(minutes=15))
        return jsonify({
            'message': message,
            'JWT': token
                    }), 202

    except Exception as e:
        return jsonify({'error':str(e)[:100]}), 500

### JWT validation
@app.route('/validate', methods=['POST'])
def validate():
    try:
        token = request.headers.get('JWT')
        header, claims = jwt.verify_jwt(token, JWT_security_key, ['PS256'])
        email = claims['email']
        payload = {'email': email}
        new_token = jwt.generate_jwt(
            payload, JWT_security_key,
            'PS256', timedelta(minutes=15))
        message = 'Token validation successful'

        return jsonify({'message': message,
                        'JWT': new_token}), 202
    except Exception as e:
        return jsonify({'message': 'Validation failed'}), 401
```

Data Security

The Elements:
- Authentication
- Authorization
- Encryption
- Crypto-Hashing
- Risk Management

The Implementation:
- SHA256 Hash
- JSON Web Tokens

https://churchmapp.netlify.com/auth

Modern DevOps Best Practices

- Source Control (git, GitHub, GitLab)
- Configuration as Code (YAML/TOML)
- Automated Testing
  (Jupyter Notebooks -- !)
- Immediate Release
  (Netlify, PythonAnywhere)
- Continual Deployment (webhooks)

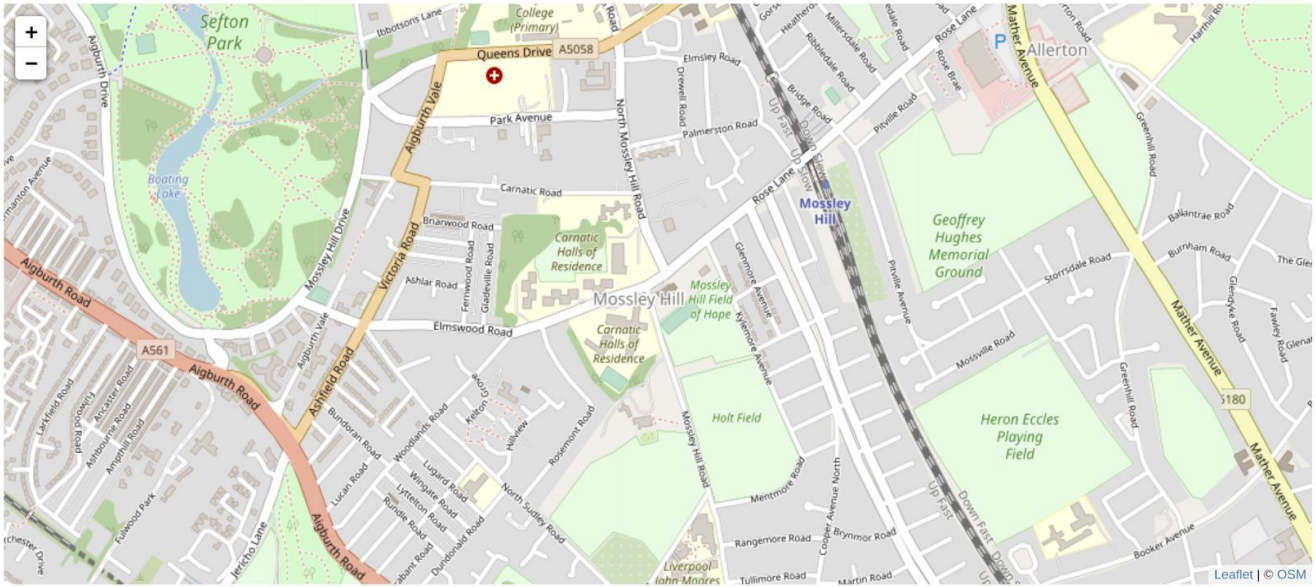
https://github.com/peterprescott/churchmapp

churchmAPP

name@email.com

password

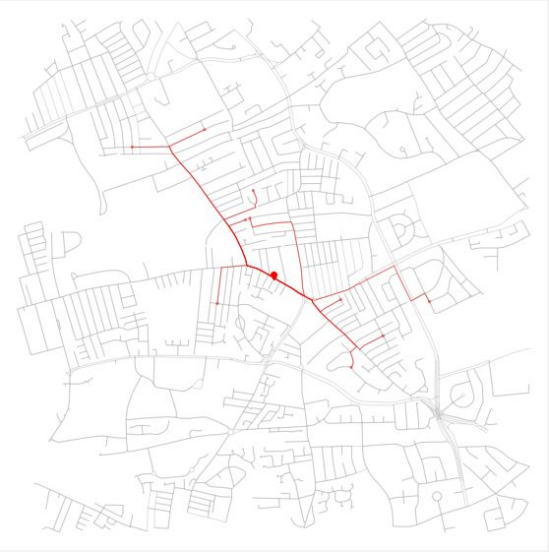
LOGIN/REGISTER

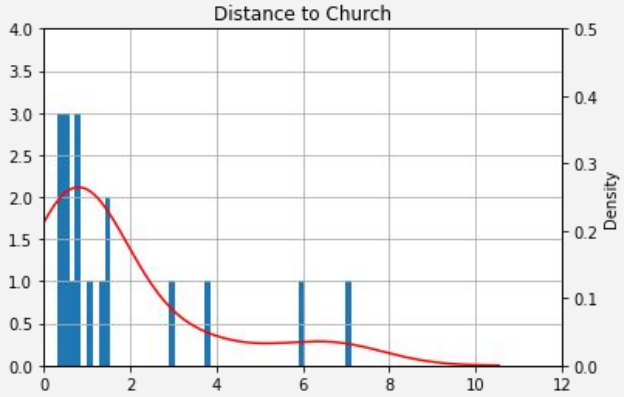


L18 8DB

Mark Postcode

From Mossley Hill to the uttermost ends of the earth...





THANKS!

Do you have any questions?