COLLEGE OF ENGINEERING

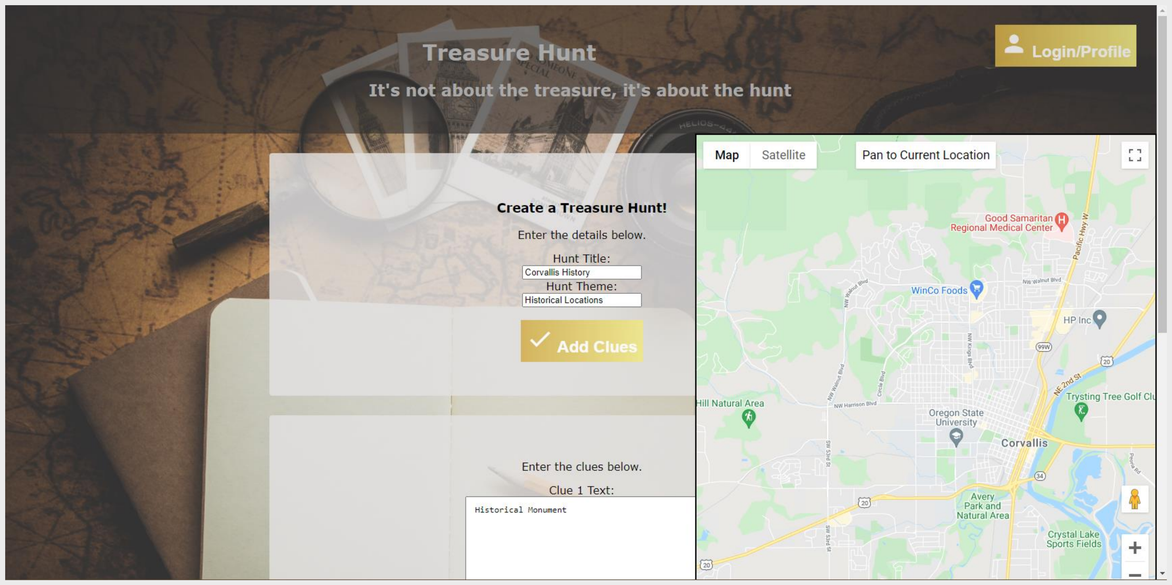
TREASURE HUNT WEB APP

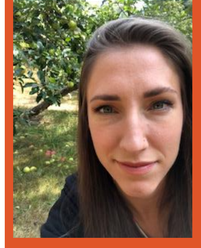
Treasure Hunt is a mobile webapp that reveals the treasures hidden all around us. Create a treasure hunt and send your friends and neighbors on an adventure. Play by solving clues to a treasured experience.

MOTIVATION

- This project built on skills that our team had learned in Web Development, Database, and Cloud Computing courses completed at OSU.
- The front end uses an HTML/CSS framework with Javascript and jQuery.
- Google Maps Javascript API enables all map functionality
- The back end uses Google Cloud Platform and Google Datastore for the server and database. The server is built using Python.

```python
# This route is used for the creation and retrieval of hunts.
# Doing a post request will allow a hunt to be created. JSON with name, theme, empty clue and treasure list req.
# Doing a get request will retrieve the hunts in the database 5 at a time. With a link to the next page in the list.
@app.route('/hunts', methods=['POST','GET'])
def hunts_get_post():
    if request.method == 'POST':
        if 'Authorization' in request.headers:
            payload = verify_jwt(request)
            if payload == 401:
                error = {'Error': 'Missing or invalid JWTs'}
                return error, 401
            content = request.get_json()
            if len(content) != 4:
                error = {'Error': 'The request object is missing at least one of the required attributes'}
                return error, 405
            new_hunt = datastore.entity.Entity(key=client.key(constants.hunts))
            new_hunt.update({"name": content["name"], "theme": content["theme"],
                             "creator": payload["email"], "clues": [], "treasures": []})
            client.put(new_hunt)
            new_hunt["hunt id"] = new_hunt.key.id
            new_hunt["self"] = URL_HUNT + '/' + str(new_hunt.key.id)
            return json.dumps(new_hunt), 201
        else:
            error = {'Error': 'Missing or invalid JWTs'}
            return error, 401
    elif request.method == 'GET':
        if 'Authorization' in request.headers:
            payload = verify_jwt(request)
            if payload == 401:
                error = {'Error': 'Missing or invalid JWTs'}
                return error, 401
            query = client.query(kind=constants.hunts)
            query.add_filter("creator", "=", payload['email'])
            # get total number of hunts
            number_of_hunts = list(query.fetch())
            # set up pagination
            q_limit = int(request.args.get('limit', '5'))
            q_offset = int(request.args.get('offset', '0'))
            l_iterator = query.fetch(limit=q_limit, offset=q_offset)
            pages = l_iterator.pages
            results = list(next(pages))
            if l_iterator.next_page_token:
                next_offset = q_offset + q_limit
                next_url = request.base_url + "?limit=" + str(q_limit) + "&offset=" + str(next_offset)
            else:
                next_url = None
            for e in results:
                e["hunt id"] = e.key.id
```





ADAM SILVER
MARCELLA PETRUCCI

SILVERA@OREGONSTATE.EDU
PETRUCMA@OREGONSTATE.EDU

HTTPS://CS467-CAPSTONE.UW.R.APPSPOT.COM
HTTPS://GITHUB.COM/QSILVER7598/TREASURE_HUNT_APP

Oregon State University

DESCRIPTION

Treasure Hunt is a fun and interactive crowd sourced game that allows users to create a treasure hunt based on a historical area, local areas of cultural importance, or just a fun neighborhood themed adventure!

Users can create a hunt for others to enjoy, or simply play an already existing hunt in their area.

Creating a hunt is a simple process of entering the name and theme of your hunt, adding the text information and the location for each clue and the treasure.

To play a hunt, the user selects a hunt from the available hunts in their area. The “treasure map” and the first clue will be shown to the user. As the user approaches the solution location for the clue, they will receive the next clue.

A player completes the hunt by locating the treasure!

FEATURES

- Users create credentials and log in using Google as an authentication medium using Google Oauth 2.0 and JSON web Tokens.
- Treasure Hunt webapp uses the Google Maps Javascript API to render a map that users interact with to during game play. The API is also used to find and update the user’s location.
- The hunts and user information will be stored in a database on Google Cloud Platform.
- The server for the web app will be hosted on Google Cloud Platform as well.
- The interaction between the server and database will be done using a REST API from the front end by the user.