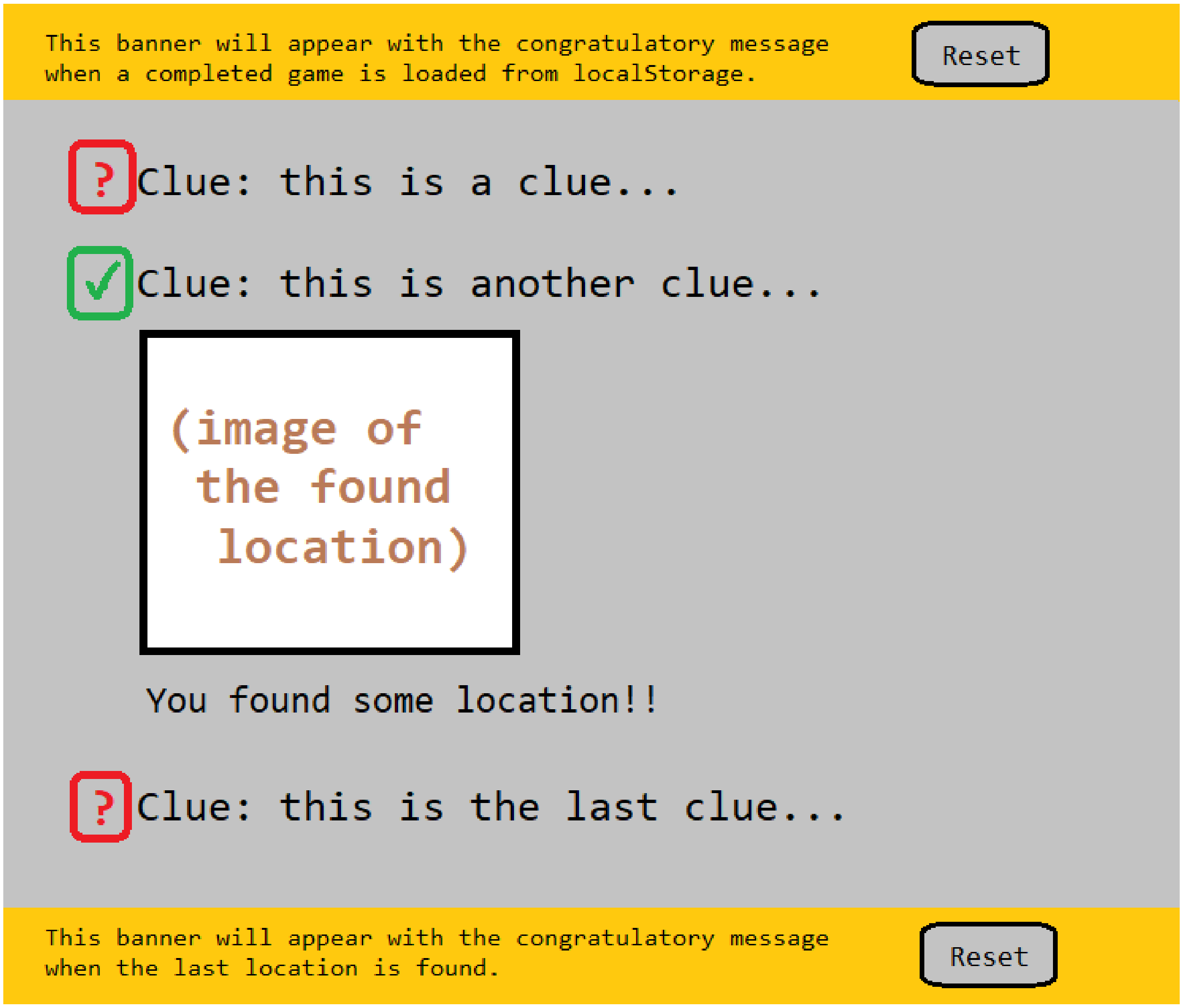
Pete Wurster;
CIS 228 – Spring 2020;
Lab 05 – Scavenger Hunt;

Layout:



This banner will appear with the congratulatory message when a completed game is loaded from localStorage.

? Clue: this is a clue...

✓ Clue: this is another clue...

You found some location!!

? Clue: this is the last clue...

This banner will appear with the congratulatory message when the last location is found.

Plan:

This game will use Classes for Location and GameState objects. A GameState object will contain the Locations as well as have functions for saving/loading the game from local storage or fetching location data from remote JSON as needed. Due to the use of Classes, this project will be divided into ECMA6 modules. The array methods map() and filter() will be used to the maximum extent throughout the project in order to practice using them. Constants will be used for strings needed in repetitive DOM tasks and to manage the precision of the geofence.

Class design:

```
Location {
    name: string,
    lat: number,
    lon: number,
    clue: string,
    id: string (“pN” where “N” is a sequential integer used later in
        arrays),
    isFound: bool
}

GameState {
    locations: array (of Location objects)
    pullLocations() -> promise (value is array of Location objects),
    save(data) -> void,
    load() -> promise (no value but used for ‘then’ chaining)
}
```

Various helper-functions will be used in an attempt to keep the main() function written at the highest level of abstraction possible.

```
Pseudocode for main() {

    let game = new GameState();
    game.load().then(
        check if game is completed;
        build DOM elements to hold location data;
        update DOM to appropriately display found locations;
        enable geolocation monitor;
    );
}
```