1. find missing assets on css sheet
2. all strong on home page should be a link --> everything else clickable if possible

Orchard Queries

to add to orchard test

The following fields are empty and need to be filled

1. panel.description
2. panel.synonym
3. panel.estimatedCost
4. panel.alternateId1
5. ...

notes

if panel.name == paneltest.name => test is not a panel.

current page TODO:

home

- □ "community first values" -> add link to news eve
- □ download missing image
- □ utmost convenience in diagnostic testing -> link to schedule test
- □ world-class service to our community -> link to contact us **(think about adding a chat for sales or reps)**

about-us

- □ lab locations -> click takes to directions to or map with phone number.

lab-services

- □ fix two broken COVID-19 links.
- □ make lab locations link to modal with google maps and contact info -> build this modal

contact us

- After logins are created, make client contact us option for chat???
- ? route to correct phone and etc info based on ip and geospatial browser / device data

careers

- □ make for job openings. Filtering for type and location.
- □ Create application form.
- □ resume upload -> parse and auto fill application form.

un-built

Client

1. Order supplies
2. request phleb / schedule collection
3. request pickup
4. ? chat

Employee

1. editable page/sections for blog type entry on certain sections/pages.
2. calender -> incorporate outlook calender -> HOW???
3. problem reporting. Problem log moved to digital. a;; problems are reported. System will alert pertinent employees -> **email or text** <- to check their problem log.

    ###### PROBLEM LOG

    problems sorted by:
    - problem reported by:
        1. location -> ip
        2. time
        3. department -> ip
        4. position/role -> ip + login info + time
        5. client
    - problem entry:
        - problem entry: client -> *search autofill* <-, other required fields: -> *auto-populate fields based on ip and typical department/section problems* <-, problem class -> *dropdown* <-.
        - notes.
        - system queries local db for info:
            - client -> reps, sales, region, facility type.
            - problem class -> description.
            - time of problem
            - location of problem
            - department(s) of problem
            - reported by.

roles:

1. employee
    1. departments : [("department", "role")]
    2. locations: [("location", "departments")]
    3. roles: ['tech', 'supervisor', 'collector', 'team-lead', 'accessioner', 'manager', 'sales', 'rep', 'logistics', 'inventory', 'admin']
    4. onDuty: Boolean *(look at replacing the timeclock, to only allow access to info when employee is clocked-in or on-call)*
2. client
    1. location(s)
    2. access_level/rights.