



Application Development Strategies (Building the Solution)

Blaise Hanagami
Mililani High School



Overview

- HACC Timeline
- Brainstorm: Solve The Problem
 - How can you help your team navigate the HACC?
- Application Development
 - Platform Choices
 - Technology Choices
 - Team (Knowledge and Skills)
- Questions/ Concerns



DISCLAIMER: I AM NOT A MASTER AT HACCC!

- Lend perspective from participating in HACCC 3+ years
- Provide helpful strategies that my team used in the past
- Warn you of the mistakes we made so you do not have to make them yourselves!



Timeline: Keep Your Crew on Time (1/2)

- Time management = most difficult aspect of HACC
 - Scale expectations and complexity of the program to the timeframe!
- General timeline:
 - 2 days prior to start: Project Postings
 - Look over options, have students develop questions to ask presenters at Kickoff
 - Kickoff (1 day):
 - Have students if possible divide and conquer
 - Ask questions and look for pros/ cons and feasibility
 - Ask what technologies client would like to integrate
 - Brainstorming (3-5 days):
 - Have students independently discuss projects that interest them
 - When project is selected have students draft an entire plan of development (more on this later)



Timeline: Keep Your Crew on Time (2/2)

- Development (14 days):
 - Have students develop their portion of applications based on Dev Teams
 - Have students keep notes of dev experience (useful for presentation Q&A Preparation)
- Video of Application and Technical Review Prep (1 day in parallel with Development)
 - Have a few students comfortable with live demo and answer questions
- Presentation Development (2 days)
- Presentation/ Ceremony (1 day)



Uh... Wait... You Expect Me To Do All Of That In 3 Weeks?!

- Prepare beforehand!
 - HACC has all of their previous proposed projects online!
 - Use Projects to guide instruction to use in classroom
 - Build Units and lessons throughout the year to facilitate application building process

<https://hacc.hawaii.gov/past-event-2019/>

- Past Winner's projects posted to use as ideas

<https://hacc.hawaii.gov/challenges/>

- Under Construction, but usually contains the raw proposals from State Departments



Brainstorm: How to Solve The Problem (1/4)

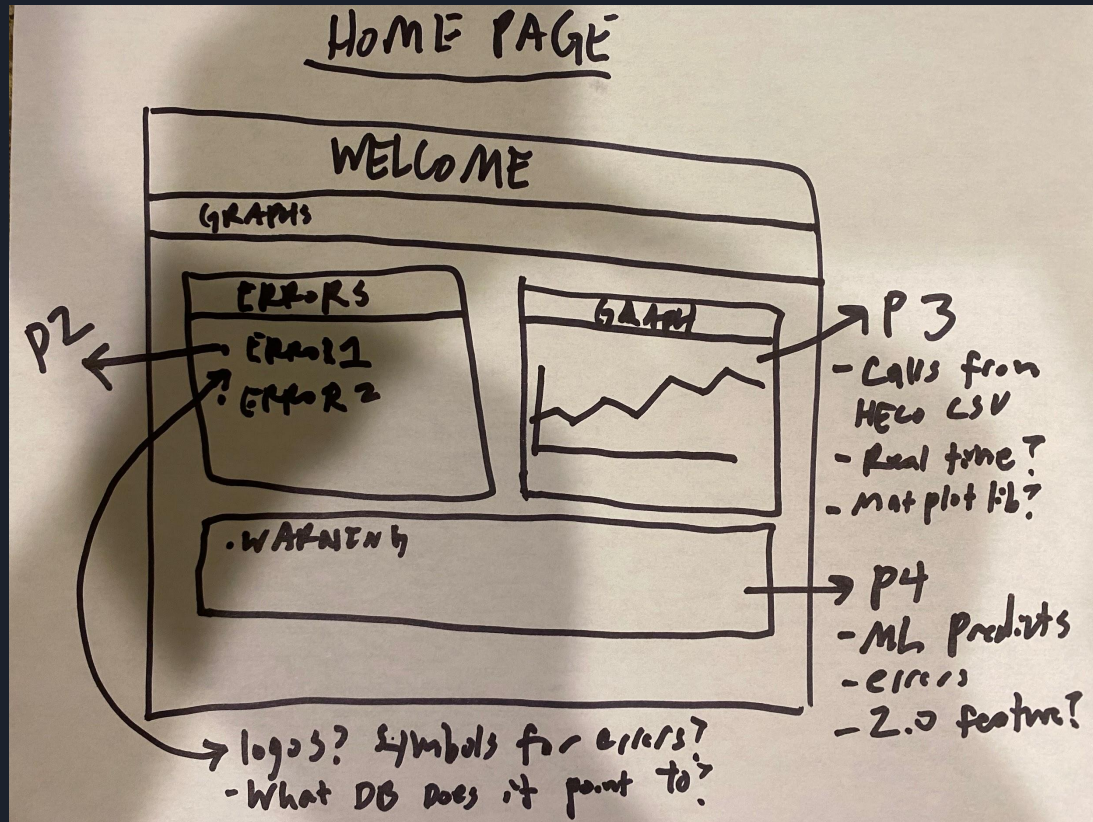
- Instructor's Primary Job: Manage expectations!
 - HACC's dev window is very short
 - Encourages rapid development and prototyping
 - Help students find the perfect balance between interest, ability and time
- Step 1: Independent Selection of Projects:
 - Have students rank projects based on interest level
 - Limit interaction between students to ensure genuine feedback
- Step 2: Pitch For Project Selection:
 - Have students present to the group proposing the pros/ cons of their top projects based upon
 - interest
 - Ability
 - Time



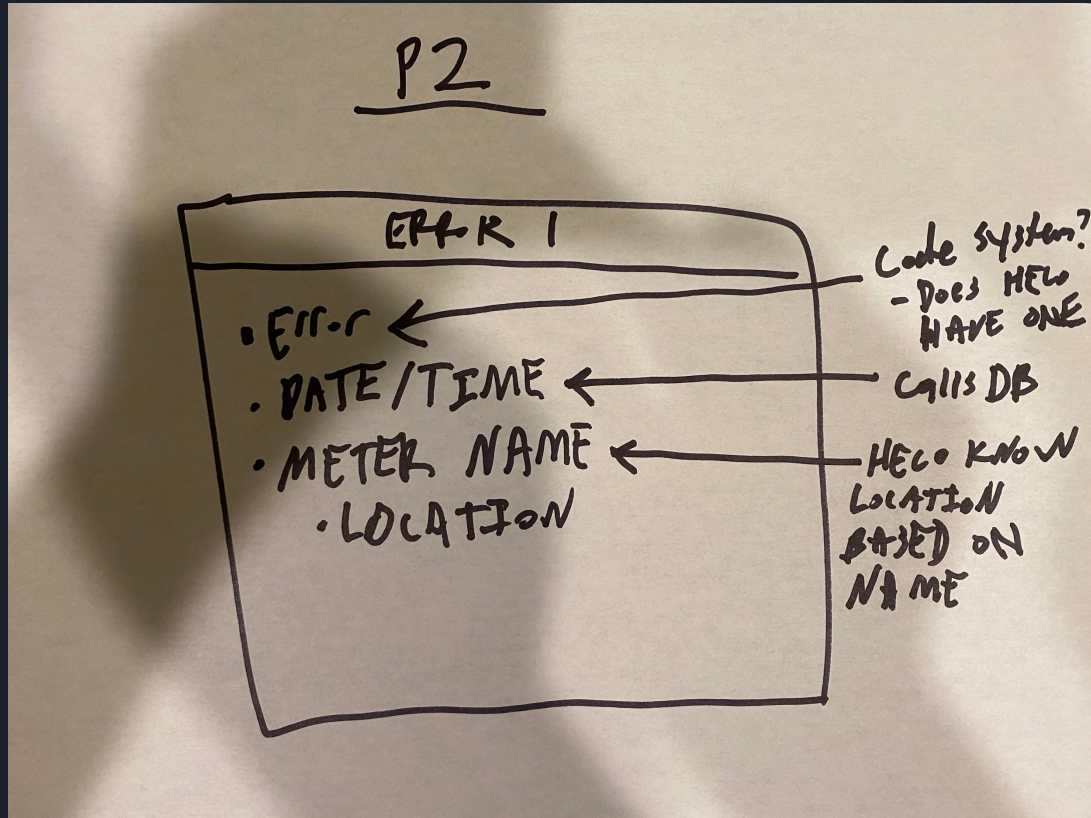
Brainstorm (2/4)

- Step 3: Vote
 - Project selection will not be unanimous
- Step 4: Create a Goal/ Mission Statement & Plan App Solution Based on UI
 - Seems backwards, but this has been highly effective for my students
 - Have students draw out what the app GUI will look like
 - If a page links to another, show the GUI of the next page
 - After GUI has been developed, break down the back end technology/ coding that will be involved to make the application work
 - Ex. if there is input from the user, it should connect to a database
 - If there is a graphing component, what API should be used to make that feature happen?
 - Students may not have detailed answers, but general terms like database, graphing, machine learning will help

Example Of GUI Planning



Example of GUI Planning





Brainstorm (3/4)

- Step 6: Pitch Ideas
 - Have students present their plan to the group
 - Facilitate discussion to ensure that all ideas are explained in a safe learning environment
- Step 7: Feedback & Vote
 - Have students ask questions of each other's plans based on IAT
- Step 8: Deeper Dive Into Planning:
 - Have the entire group look at the selected UI and suggest changes
 - Have the group then think about the back end things that will need to be built
 - Teacher: manage expectations by having students decide what features are needed for version 1.0
- Step 9: Labor Distribution
 - Split students into development teams with as specific jobs as possible (Front end, Database, Framework etc.)
 - Appoint/ vote for head of each dev team
 - Explain to students that labor may need to shift during the project based on need



Brainstorm (4/4)

- Step 10: Dev Teams Plan Their Part
 - Have students logic map out code that will need to be developed
 - Enables all members to have a common document to refer to

IMPORTANT: Teacher needs to monitor all groups at all times!

- Establish good soft skills
 - Look for teachable moments
- Ensure plans are feasible and realistic




Application Development: Platform Choices

- Before HACC, a general platform that is used in class should be used
 - This will help students have perspective on what is feasible
- Criteria for a good platform:
 - Flexibility
 - Language
 - RESOURCES:
 - Tutorials
 - Documentation
 - Curriculum/ Lessons
 - Is the platform used by industry?
 - Gives students valuable experience and skills that are not HACC specific



Application Development: Technology Choices

- Assess what types of functionality your project requires
- Solutions do not need to be expensive!
 - You do not need the top of the line computers to code
 - Open Source resources
 - Paid software usually has an Open Source equivalent
 - Free opportunities for education
 - Google gives free server time for educational purposes
- Hardware driven solutions can be sometimes emulated
 - If hardware solutions are not available, challenge students to think of software-based solutions
- Talk to your tech team before starting
 - programs / plugins/ libraries may need admin rights to install
 - Leverage IDE Virtual Environments



Application Development: Team Knowledge and Skill

- All students need to know how to code
 - Highly specialized skill sets (Public Speaker, the “Video Guy”) are tertiary to actual skill in coding
 - Members of team who only have specialized skills can cause morale to fall
- Teacher needs to act as the “Taskmaster”
 - Constantly check in with teams to see what they are working on, where they are at
 - Sometimes a solution to one group’s problems is already with another
 - Build a working environment that encourages dev teams to talk to each other constantly
 - Helps to ensure naming conventions, consistent variable names etc
- All members need to be trained in all skills
 - Students may have their “favorite thing” to do, but some areas will need to be done no matter what
 - Ex. Students love to work with Machine Learning, but no one wants to do the boring database stuff!
 - Labor will need to be shifted to meet needs of team
 - Ex. If the HTML Design team is done, they might need to help the framework team troubleshoot an issue or type out code.
 - Establish the rule: NO PERSON IS ABOVE ANY JOB!



Problems & Pitfalls (1/2)

- Students need to communicate constantly
 - Have students make a Slack or Discord
- Always Revisit the Mission/ Goal Statement
- Check in with Point of Contact Regularly
 - Verify that you are meeting Client's needs
 - Might lead to answers or resources that expedite development
- Accountability is HUGE!
 - If a person does not do their part, the whole project can be in jeopardy
 - Constantly ask students for proof of progress
- You will not know the answer to everything, and that is ok!
 - Students should research solutions, but they should have some guidance from YOU



Problems & Pitfalls (2/2)

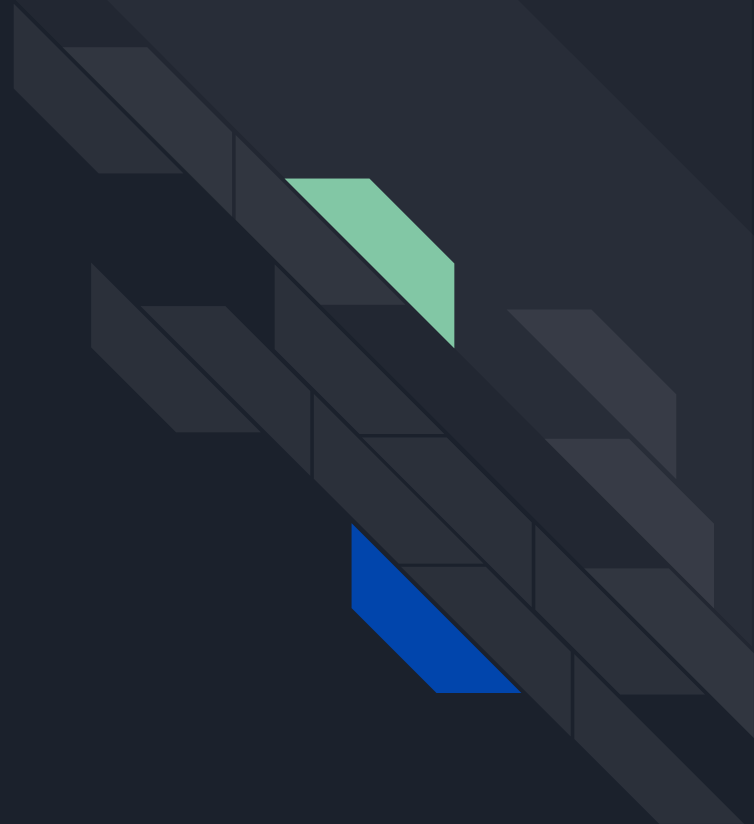
- Application development is sequential
 - Certain jobs cannot be started until a previous component is complete
 - Delegate labor to meet needs
- Students lack a realistic concept of time
 - Ask students probing questions about their code, how much time they think it will take and adjust their expectations accordingly
- Have people outside of the dev group test the application
- Connecting everything together takes the most time!
 - Things work in isolation, but are hard to connect in the end
 - Ex. Getting your database to connect to the graphing function on the site



Activity: Let's Do a UI Brainstorm!

- Using the strategies we discussed before, let us do a run through
- Goal: Develop a web application that enables users (Botanists) to upload pictures of Native Hawaiian Plants, their location, and the date. Collected data should populate over a map of Hawaii with search functions that enable the user to filter output.

Questions? Concerns?



Thank You!

