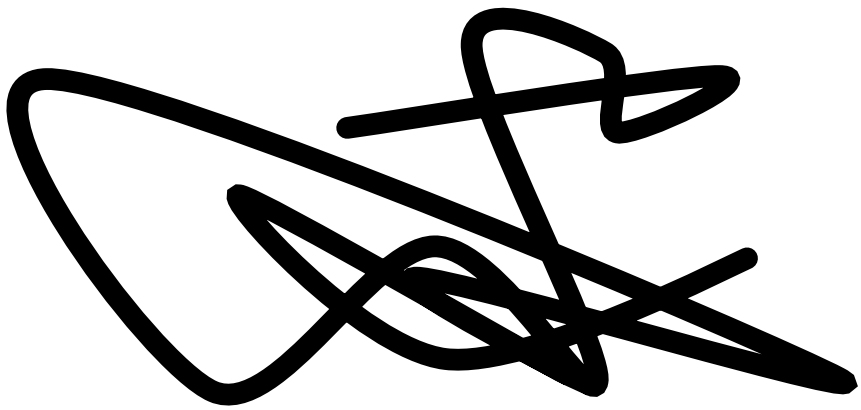
ECEN 5783 - Embedded Interface Design, Spring 2023

SquiggleDraw

Generative art prototyping with digital fabrication



Deanna Gelosi

ATLAS Institute, CU Boulder

https://github.com/deannagelosi/SquiggleDraw

Updated Work Breakdown Structure

An outline of planned work across three phases with a final due date of May 2.
The following color coding on the lefthand side of the page indicates with deliverables have been completed and which are left to do.

Complete To Do

Phase 1 (Project 6): Send SVG from phone to thermal printer (MVP), Design and User Testing

- Deliverable 1: Hardware
    - Task 1: Connect Raspberry Pi to thermal printer (S)
- Deliverable 2: Software and AWS
    - Task 1: Print to thermal printer from Raspberry Pi (S)
    - Task 2: Print a picture with thermal printer (S)
    - Task 3: Create print db on Raspberry Pi (M)
    - Task 4: AWS receives new squiggle
        - Sub Task 1: AWS DynamoDB stores squiggle data (S)
        - Sub Task 2: API Gateway to receive new squiggle (S)
        - Sub Task 3: Lambda function to insert new squiggle into DynamoDB (S)
    - Task 5: AWS sending squiggle to Raspberry Pi
        - Sub Task 1: Lambda function retrieves squiggle from DynamoDB (S)
        - Sub Task 2: AWS API Gateway to trigger Lambda (S)
        - Sub Task 3: Raspberry Pi uses API to retrieve new squiggle (M)
- Deliverable 4: Design and User Testing
    - Task 1: Architecture Diagram
        - Subtask 1: List major components of the overall system (S)
        - Subtask 2: Design diagram in Miro (S)
    - Task 2: Work Breakdown Structure
        - Subtask 1: Identify tasks to be completed sequentially (S)
        - Subtask 2: Write up in WBS format (S)
    - Task 3: Use Case Diagram using WAVE rule
        - Subtask 1: List specific tasks for users, including optional or follow-on required interactions (S)
        - Subtask 2: Design user case diagram in Miro (M)
    - Task 4: UI Wireframes
        - Subtask 1: Low fidelity wireframes for user interfaces (mobile, queue) (S)
        - Subtask 2: Medium fidelity wireframes in Balsamiq (M)
    - Task 5: Paper Prototype/Wizard of Oz Test
        - Subtask 1: Select user interface for paper prototype (S)
        - Subtask 2: Define three primary tasks for users to perform with the designed interface (S)

        - Subtask 3: Write a brief script to run the test (S)
        - Subtask 4: Recruit at least three “users” for testing (S)
        - Subtask 5: Test the three tasks on the paper prototype with each user (S)
    - Task 6: Project 6 Writeup
        - Subtask 1: Combine Tasks 1-5 into a PDF document (M)
        - Subtask 2: Unify formatting and page layout (S)

Phase 2 (Project 7): Mobile Interface and Messages

- Deliverable 1: AWS receives new squiggle
    - Task 1: Send from remote (Postman), not within AWS console (S)
    - Task 2: Send from website instead of Postman (S)
    - Task 3: Secure API access with event invite key with error handling (M)
    - Task 4: Mark incoming squiggles as “new” (S)
    - Task 5: When Raspberry Pi requests squiggles, only send new (S)
    - Task 6: Mark squiggles sent to Raspberry Pi as “sent” (S)
- Deliverable 2: Mobile UI - Basic
    - Task 1: Create basic mobile app (HTML client) to test parameters (S)
    - Task 2: Automatically print from Raspberry Pi db (S)
    - Task 3: Mark illustration as printed in queue when sent to printer (S)
- Deliverable 3: Thorough Component List
    - Task 1: Produce comprehensive list for super project (S)
- Deliverable 4: Updated Work Breakdown Structure
    - Task 1: Review Project 6 WBS to add any missing data (S)
    - Task 2: Remove and modify data from the WBS (S)
- Deliverable 5: Demonstration and Discussion
    - Task 1: Demonstrate basic operations of at least half of components (M)
    - Task 2: Discuss the remaining half, including technical work to be done (S)

Phase 3 (Project 8): Squiggle Generator code and Display

- Deliverable 1: Mobile UI Design
    - Task 1: Create mobile UI design in Figma (M)
    - Task 2: Export to HTML using Anima (S)
    - Task 3: CSS/JS changes to load SVG example (M)
    - Task 4: Send squiggle + parameters to AWS API Gateway (M)
- Deliverable 2: AxiDraw Output
    - Task 1: Hook up AxiDraw to Raspberry Pi and do test print (M)
    - Task 2: AxiDraw controller script to print from db (L)
- Deliverable 1: Raspberry Pi LCD UI
    - Task 1: Create Qt UI in Figma to display queue and control AxiDraw (M)
    - Task 2: Export UI using Qt Bridge (S)
    - Task 3: Python software to load and control Qt UI (L)

        - (Stretch) Task 4: User chooses boundaries (L)
        - (Stretch) Task 5: Multiple squiggles with user-selected origins (L)
        - (Stretch) Task 6: Saves SVG to S3 for user to retrieve later (M)
    - Deliverable 3: Thermal Receipt Paper Design
        - Task 1: Design layout for receipt paper in Figma (S)
        - Task 2: Program design into printer controller (M)

Notable Changes

- Removed
    - Enclosure: I removed the enclosure deliverable from Phase 3. While having an enclosure for the thermal printer is something I would like to see in the future, it's not part of the core functionality of the project.
- Moved
    - Mobile UI Design: Originally, this deliverable was bundled in with the Mobile UI deliverable in Phase 2. I moved it to Phase 3, which will primarily consist of design elements.
    - AxiDraw: I moved this deliverable to Phase 3 as well. I feel confident about controlling the AxiDraw with a Raspberry Pi because I've seen this done before and know there's documentation online to support this task.
- Added
    - AWS receives new squiggle: In Phase 2, I added more clarity around what needs to happen for AWS to receive new squiggles and mark sent squiggles as "sent." This also involved more work to secure the API access with an event invite key.

Component List and Updated System Diagram

An updated system diagram informed the creation of the following component list. Each of these system component are core to the functionality of the project.

- Phone to Web App
- Web app to API Gateway (POST)
- API Gateway (POST) to Lambda function (iot23_db_add_dege)
- Lambda function (iot23_db_add_dege) to DynamoDB
- DynamoDB to Lambda function (iot23_db_read_dege)
- Lambda function (iot23_db_read_dege) to API Gateway (GET)
- API Gateway (GET) to Python connection script (connect_AWS.py)
- Python connection script (connect_AWS.py) to PostgreSQL DB
- PostgreSQL DB to thermal printer script (print_controller.py)
    - TThermal printer script (print_controller.py) to Thermal Printer
- PostgreSQL DB to AxiDraw script (axi_controller.py)
    - AxiDraw script (axi_controller.py) to AxiDraw
- PostgreSQL DB to LCD display script (lcd_display.py)
    - LCD display script (lcd_display.py) to LCD Display

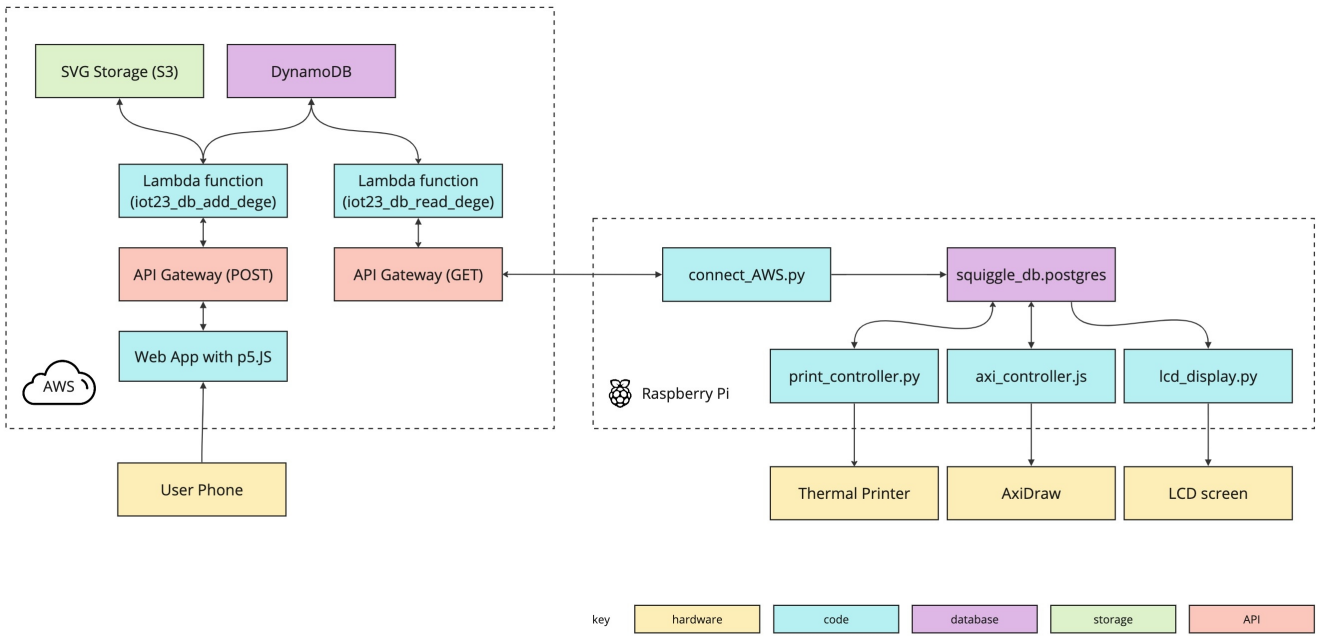


Fig1: Updated System Diagram.

Component or Partial Demonstrations

A live demo with Professor Spriggs of working components is scheduled for Friday, April 21 at 9 am.

Github Repo

https://github.com/deannagelosi/SquiggleDraw