

Group 29 – Microbiology Lab Information Management and Visualization System (GraphKey)

Team Members: Benjamin Vogel, Brittany McPeck,
Samuel Jungman, Rob Reinhard, Kyle Gansen, Ben
Alexander

Technical Advisor: Thomas Daniels
Client: Karrie Daniels

Email: sddec20-29@iastate.edu or bavogel@iastate.edu

Problem Statement

- › Many scientists and researchers dedicate large amounts of time towards organizing, maintaining, and visualizing the data they collect.
- › The solution should be able to automate the process of organizing, maintaining, and visualizing data.

Project Goals Recap

- › Intuitive Graphical User Interface
- › Support importation and parsing of large amounts of data
- › Generate Graphs and Visualizations of the data
 - › Generate large combinations of data into many graphs
 - › Allow user to customize the generated graphs
 - › Perform some statistical analysis on the data
- › Support exportation of the generated graphs to Google Drive and the local machine
- › Ensure the system could be maintained by one or two people

Project Progress

- › One unified application in development
 - › No more branching prototypes, everything works off of one UI
 - › Backend is abstracted in a format that makes it easier for the UI to be changed without affecting the actual business logic
 - › Has core functionality while also leaving hooks in for expansion
 - › Can be packaged and distributed as a single “executable”
- › Mass graph generation
 - › User can now select different points of data and have multiple different graphs generate at the same time
- › Data Saving
 - › Data created by projects, graphs, or user preferences persists between sessions

Reworking the Backend

- › Previously, all our projects worked, but were independent of each other
- › Needed to unify graph generation across all projects regardless of if it was a new experimental GUI or or the old one, the graph generation would stay the same
- › Needed to allow for expansion of new graphs if the client desired
- › Needed to support graph customization (colors, shapes, different data sets, names, titles) while also allowing that customization to be saved

Reworking the Backend - Approach

- › Figure - Abstract class each graph function will implement
- › Individual graphs - Implement a `construct_figure()` method
 - › User passes a dictionary that holds parameters such as names, data, colors, shapes, etc.) into the method
- › Factory method - GUI calls the `FigureFactory` with their desired figure and gets a `Figure` object
 - › No more recursive changes throughout the GUI, only in the `Factory` class

```
class Figure(abc.ABC):
    """
    This is an abstract container class that holds the Plotly figure object. All that gets passed is the pandas
    DataFrame and a config dictionary or JSON object that holds all the information. Plotly then generates the figure
    and is held here.

    This is also where the other graphing figures inherit from (eg. Boxplot, Barchart, Scatter, etc.)
    """
    @abc.abstractmethod
    def __init__(self, df: pandas.DataFrame):
        pass

    @abc.abstractmethod
    def construct_figure(self, config: dict):
        pass

    @abc.abstractmethod
    def get_figure(self) -> go.Figure:
        pass

    @staticmethod
    def eliminate_nones(config: dict):
        fig_config = config
        for key, value in fig_config:
            if value is None or value == "None":
                del fig_config[key]
```

Reworking the Frontend

- › We previously had several different prototypes of the UI with different features implemented on each one
 - › Disjointed UI windows from prototypes are streamlined and managed by window manager; additionally, a standardized menu bar also toggles actions
 - › Needed to rework the UI so it isn't so cluttered
- › We also wanted the user to have the ability to create projects
 - › So the data and graphs generated for one experiment wouldn't get mixed up with another experiment
 - › So the user could save a project and then open it back up later

Reworking the Frontend - Result

Microbiology Lab Information Management and Visualization System

Project File Demos sddec20_29

Select Workbook
CleanedExperimentData

Select Workbook Edition
1

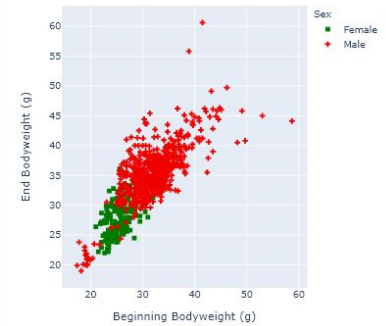
Select Excel Sheet
Behavioral Data

Sheet Graphs

- Bar x-Sex y-Beginning Bodyweight (g)
- Box Plot x-Experiment Type y-End Bod
- Box Plot x-Sex y-End Bodyweight (g)
- Scatter x-Beginning Bodyweight (g) y-
- Scatter x-Beginning Bodyweight (g) y-

Graph Type
Scatter Plot

Generate Graphs



Graph Title

Size

X-Axis Title

Y-Axis Title

Enable logarithmic x-axis

Enable logarithmic y-axis

Save Changes

X-Variable

- Date of Experiment
- Animal Number
- Beginning Bodyweight (g)
- End Bodyweight (g)
- Bodyweight: % Difference
- Corticosterone Values (pg/ml)
- EPM: Average Speed (m/s)
- EPM: Total Distance Travelled (m)
- EPM: Longest Visit to Open Arm
- EPM: Average Speed in Open Arm
- EPM: Distance Travelled in Open Arm
- EPM: Latency to First Entry to Open Arm
- EPM: # of Entries to Open Arm
- EPM: Time Spent in Open Arm (s)

Y-Variable

- Date of Experiment
- Animal Number
- Beginning Bodyweight (g)
- End Bodyweight (g)
- Bodyweight: % Difference
- Corticosterone Values (pg/ml)
- EPM: Average Speed (m/s)
- EPM: Total Distance Travelled (m)
- EPM: Longest Visit to Open Arm
- EPM: Average Speed in Open Arm
- EPM: Distance Travelled in Open Arm
- EPM: Latency to First Entry to Open Arm
- EPM: # of Entries to Open Arm
- EPM: Time Spent in Open Arm (s)

Color Grouping Variables

- Experiment Type
- Sex
- Breed
- Length Of Experiment
- Housing (2/or 4/cage)
- Diet
- Location
- Experimental or Control

Marker Grouping Variables

- Experiment Type
- Sex
- Breed
- Length Of Experiment
- Housing (2/or 4/cage)
- Diet
- Location
- Experimental or Control

Technical Challenges

- › Speed of the Program
 - › Running off of a compressed Python Zip File works for an executable, but has a longer boot time than running from source
 - › Importing LARGE excel files can make the program hang while it imports and categorizes the data
- › Customization
 - › To speed up multiple graph generation, want to save user's preferences from session to session
 - › How do we persistently save information in a way that's safe yet also space-sensitive?

What's Next

- › Wrap up tweaks, customization, saving templates
- › Documentation (user guides, README, etc.)
- › Final Report
- › Final Poster
- › Bug-fixes and small features that can be added within the next few days/week

Email Address

sddec20-29@iastate.edu
bavogel@iastate.edu