IRS.v2.a

By: Jessica Bishop, Sukhmai Kapur, Prem Sakala, Michael Keohane

Introduction

Front-End

Jessica Bishop

3rd year Computer Science

2nd Semester on IRS

Mike Keohane

2nd year Computer Science

1st Semester on IRS

Backend

Sukhmai Kapur

2nd year Computer Science

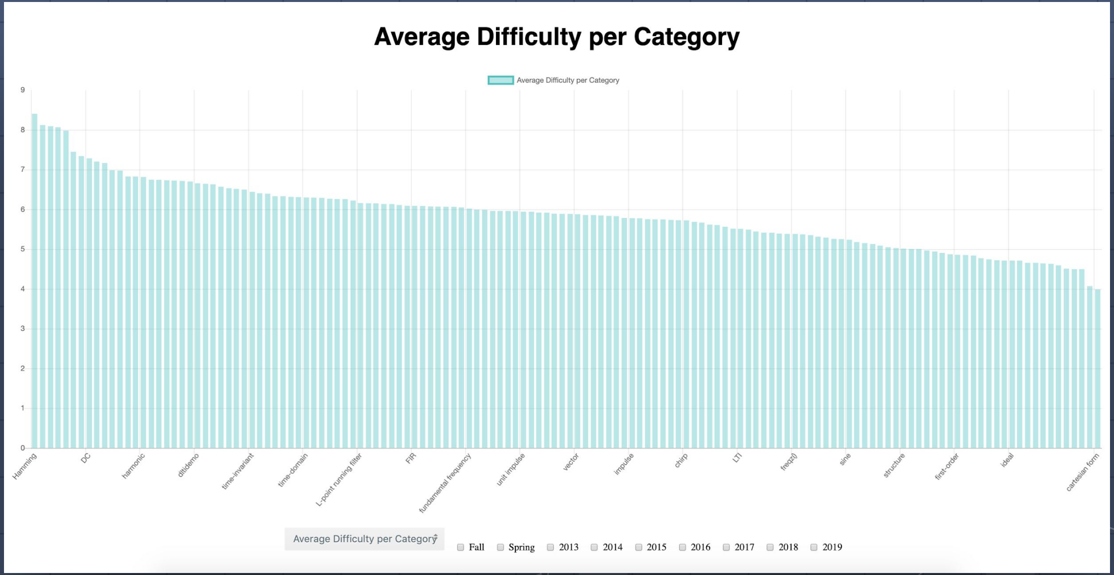
2nd Semester on IRS

Prem Sakala

3rd year Computer Science

1st Semester on IRS

Motivation



Major Flaws in Old Design:

1. Too many data points displayed
2. Limited Filters
3. Only one “view” for all types of users
4. Way too slow
5. All calculations during runtime

Goals

Backend

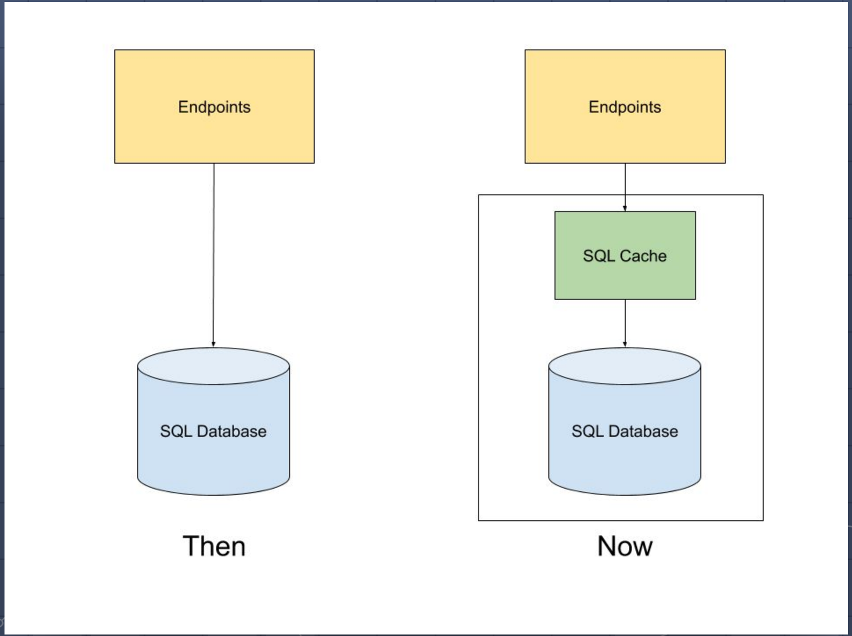
- Increase speed
- Lower complexity of SQL querying
- Decrease runtime processing

Frontend

- Create multiple views for different users
- Improve filtering with a greater range of filters
- More user-friendly graphs and UI

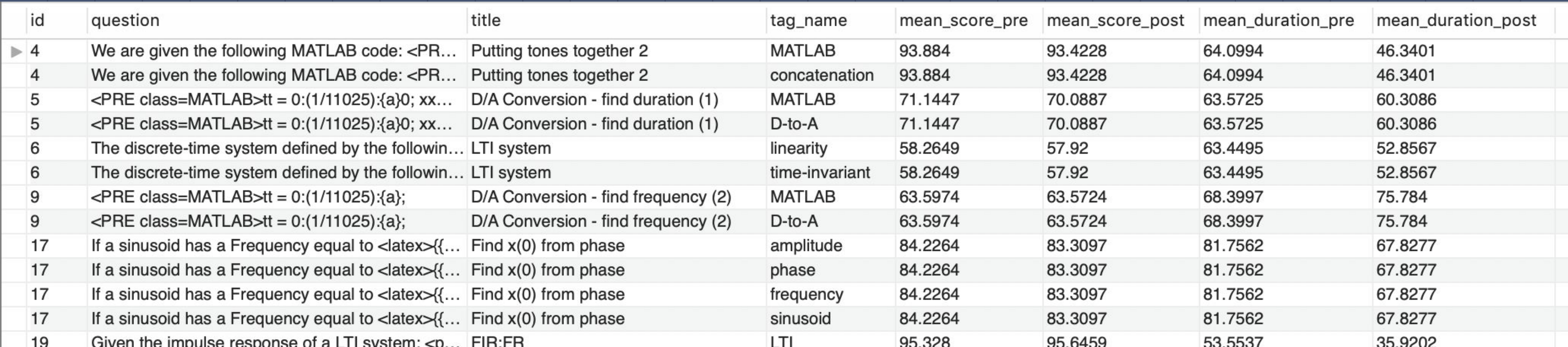
Backend Approach

- Do less calculations at runtime
- Cache Results
- Multiple Quick Endpoints
- Heavily Parameterized
- Simplified Queries



Caching

- New tables
- Stores statistics
- Requires initial run
- Increases runtime speed
- Wrapper methods

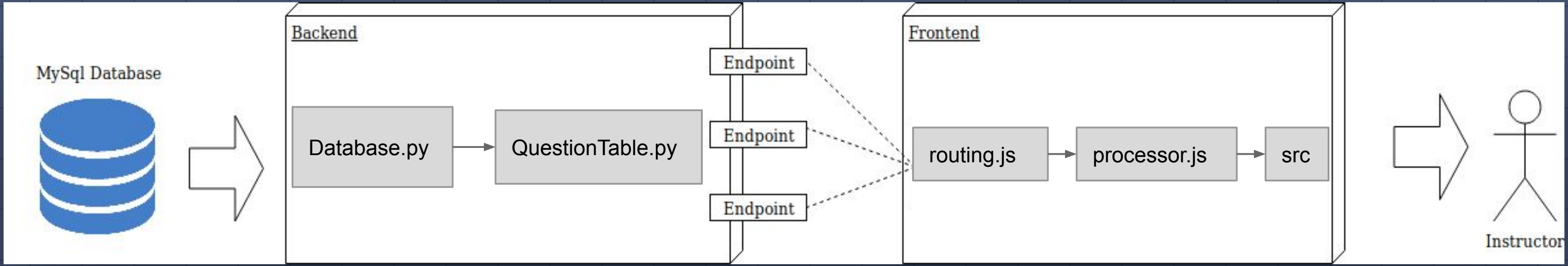


id	question	title	tag_name	mean_score_pre	mean_score_post	mean_duration_pre	mean_duration_post
4	We are given the following MATLAB code: <PR…	Putting tones together 2	MATLAB	93.884	93.4228	64.0994	46.3401
4	We are given the following MATLAB code: <PR…	Putting tones together 2	concatenation	93.884	93.4228	64.0994	46.3401
5	<PRE class=MATLAB>tt = 0:(1/11025):{a}0; xx…	D/A Conversion - find duration (1)	MATLAB	71.1447	70.0887	63.5725	60.3086
5	<PRE class=MATLAB>tt = 0:(1/11025):{a}0; xx…	D/A Conversion - find duration (1)	D-to-A	71.1447	70.0887	63.5725	60.3086
6	The discrete-time system defined by the followin…	LTI system	linearity	58.2649	57.92	63.4495	52.8567
6	The discrete-time system defined by the followin…	LTI system	time-invariant	58.2649	57.92	63.4495	52.8567
9	<PRE class=MATLAB>tt = 0:(1/11025):{a};	D/A Conversion - find frequency (2)	MATLAB	63.5974	63.5724	68.3997	75.784
9	<PRE class=MATLAB>tt = 0:(1/11025):{a};	D/A Conversion - find frequency (2)	D-to-A	63.5974	63.5724	68.3997	75.784
17	If a sinusoid has a Frequency equal to <latex>{{…	Find x(0) from phase	amplitude	84.2264	83.3097	81.7562	67.8277
17	If a sinusoid has a Frequency equal to <latex>{{…	Find x(0) from phase	phase	84.2264	83.3097	81.7562	67.8277
17	If a sinusoid has a Frequency equal to <latex>{{…	Find x(0) from phase	frequency	84.2264	83.3097	81.7562	67.8277
17	If a sinusoid has a Frequency equal to <latex>{{…	Find x(0) from phase	sinusoid	84.2264	83.3097	81.7562	67.8277
19	Given the impulse response of a LTI system: <p…	FIR;FR	LTI	95.328	95.6459	53.5537	35.9202

Backend Structure & Implementation

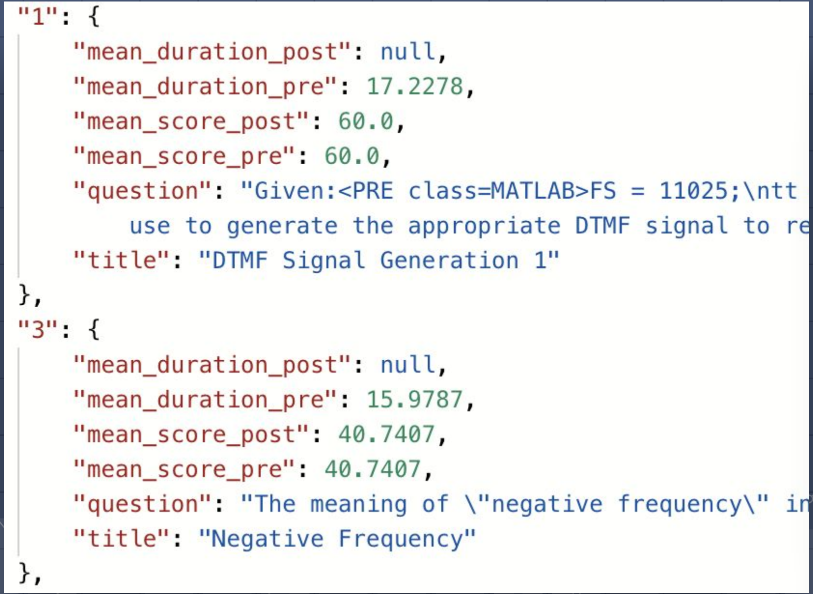
- QuestionTable.py
  - Caches data into multiple compiled, comprehensive tables
- Analysis.py
  - Queries database for summary statistics
- Database.py
  - SQLAlchemy wrapper methods
- Main.py
  - Endpoints
- **Rationale?**
  - Speed & Efficiency

Endpoints


MySql Database
Backend
Database.py
QuestionTable.py
Endpoint
Endpoint
Endpoint
Frontend
routing.js
processor.js
src
Instructor

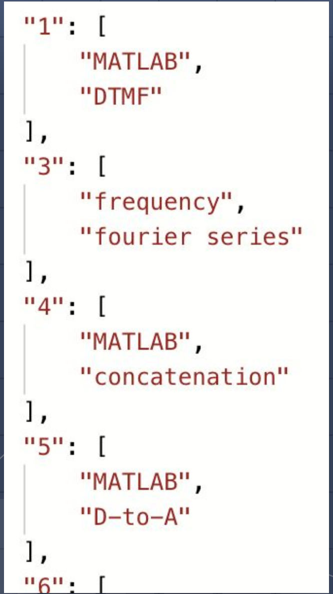
Endpoints (Postman Visualization)

/get_question_table



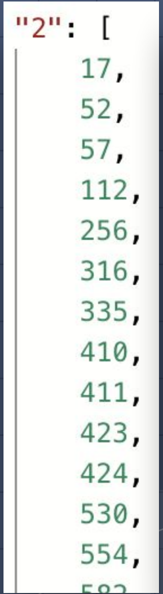
```
"1": {
    "mean_duration_post": null,
    "mean_duration_pre": 17.2278,
    "mean_score_post": 60.0,
    "mean_score_pre": 60.0,
    "question": "Given:<PRE class=MATLAB>FS = 11025;\ntt
        use to generate the appropriate DTMF signal to re
    "title": "DTMF Signal Generation 1"
},
"3": {
    "mean_duration_post": null,
    "mean_duration_pre": 15.9787,
    "mean_score_post": 40.7407,
    "mean_score_pre": 40.7407,
    "question": "The meaning of \"negative frequency\" in
    "title": "Negative Frequency"
},
```

/get_categories

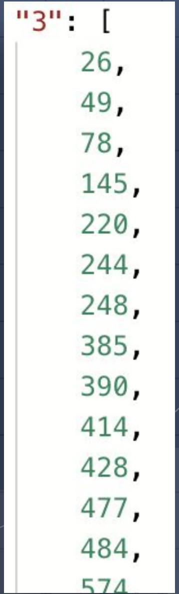


```
"1": [
    "MATLAB",
    "DTMF"
],
"3": [
    "frequency",
    "fourier series"
],
"4": [
    "MATLAB",
    "concatenation"
],
"5": [
    "MATLAB",
    "D-to-A"
],
"6": [
```

/get_questions_per_chapter?chapters=2,3



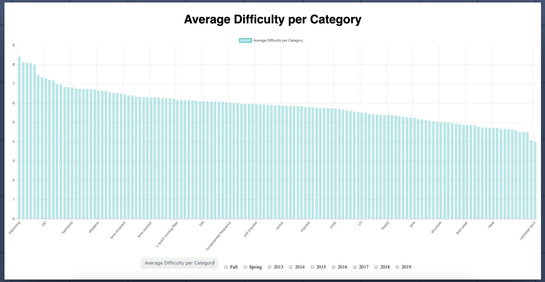
```
"2": [
    17,
    52,
    57,
    112,
    256,
    316,
    335,
    410,
    411,
    423,
    424,
    530,
    554,
```

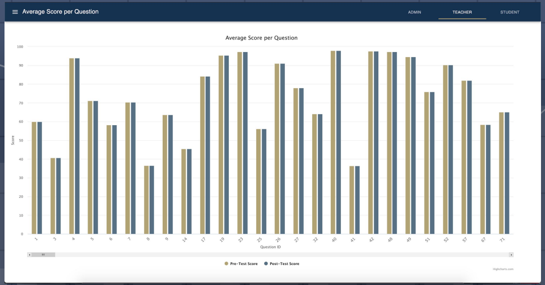


```
"3": [
    26,
    49,
    78,
    145,
    220,
    244,
    248,
    385,
    390,
    414,
    428,
    477,
    484,
    574
```

Front-End Design: Visual Overhaul

- In order to improve the user experience we revamped the UI for the entire application using:
  - *Material UI*
  - *Highcharts*


Average Difficulty per Category
Average Difficulty per Category
Fall
Spring
2013
2014
2015
2016
2017
2018
2019

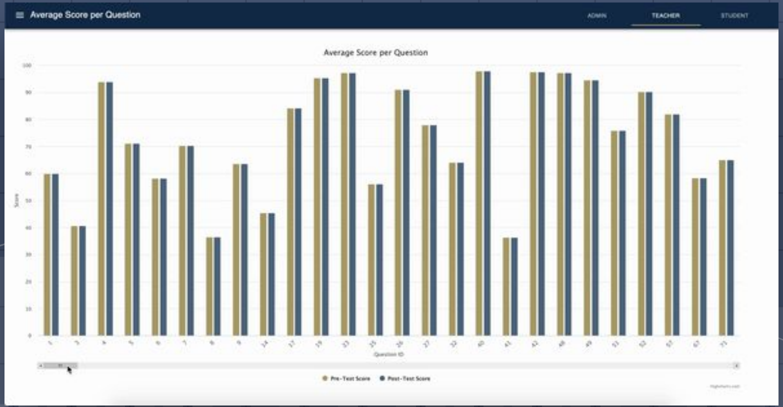


Front-End Design - Highcharts

- Graph Library created for large datasets
- NPM package that acted as a wrapper for easy usage as a React component

Benefits

- Scrollable graphs reduce clutter
- Tooltips provide more insight into each question
- Can zoom in to view smaller samples



Front-End Design - Material UI

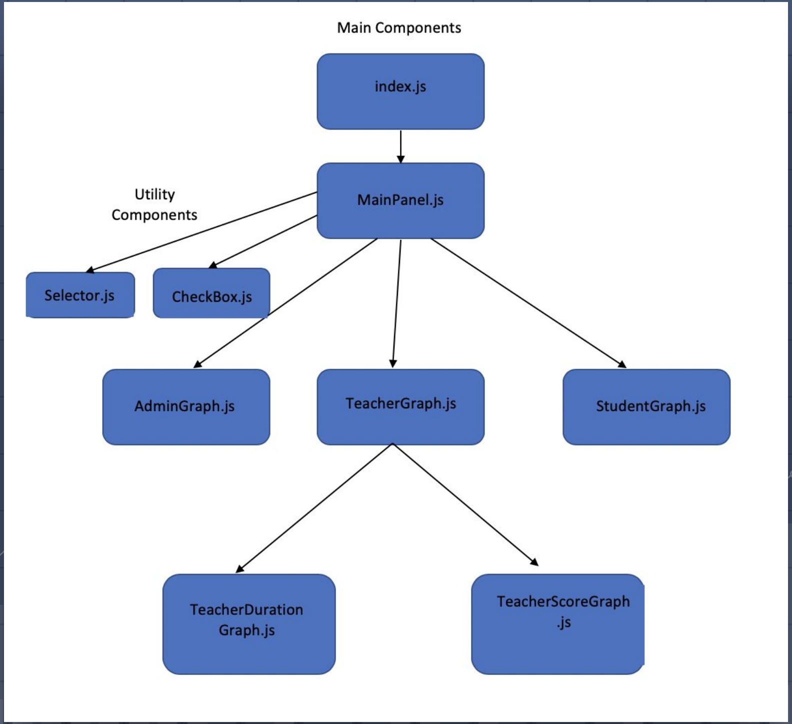
- Design Libraries offer pre-made working components for React
- Think modular bootstrap pieces that are React components

Our project used:

- AppBar
- CircularProgress (Loading circle)
- Grid (Layout)
- SelectorMenu
- CheckBox
- Tabs

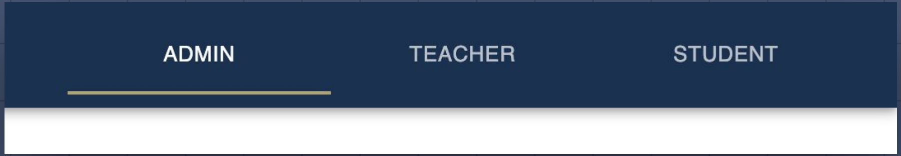
Front-End Implementation: Redesigning Structure

- Utilizing React Components
  - Separating different tabs and graphs
  - Utility Components



Front-End Implementation: Tabs

- Admin, Student, and Teacher
- Eventually the tabs will offer different graphs and options
- Set up for future progress and implementation by adding routing

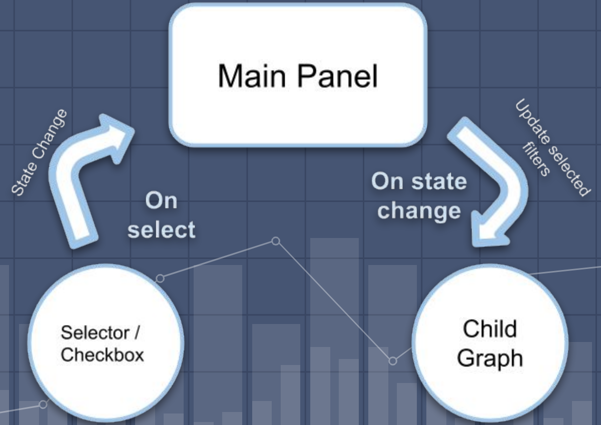


http://localhost:8080/#/Admin

Front-End Implementation: Filtering

Filters:

- Pre-Test Chapters
- Post-Test Chapters
- Semester
- Pre/Post Test



Filters declared and rendered in MainPanel.js

↓

When a filter is pressed and changed Selectors.js or Checkbox.js triggers a state change in MainPanel.js. This state change records the currently selected filters

↓

MainPanel renders the appropriate graph and sends the currently selected filters as a prop to the child graph component

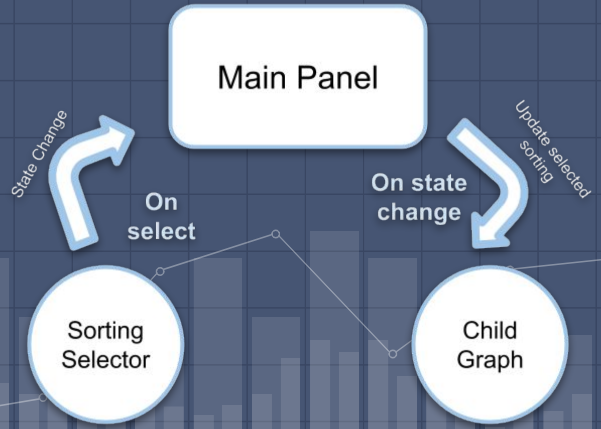
↓

The child graph component sends the selected filters to the API to receive appropriate data

Front-End Implementation: Sorting

Sort by:

- ID
- Ascending
- Descending



Sorting declared and rendered in MainPanel.js

↓

When a sort is pressed and changed Selectors.js triggers a state change in MainPanel.js. This state change records the currently selected filters

↓

MainPanel renders the appropriate graph and sends the currently selected sorting method as a prop to the child graph component

↓

The child graph component acknowledges the sort method and sorts the data itself. It then gives it to Highcharts to be displayed

Progression

BACKEND

September

Research on methods for processing data. Initial setup for caching tables.

October

Made methods to create tables to store statistics. Made endpoints to get table data.

November

Created filtering parameters: semesters, assignments and categories.

FRONTEND

September

Researched design and chart libraries and began structuring react components

October

Implemented Material UI, Highcharts to improve visuals. Added tab bar for separate views

November

Added new graphs from data retrieved from Backend. Added filtering and sorting functionality

Conclusion

- Successfully sped up system multifold
- Added several filter options
- More user-friendly graphs
- Fluid system
- Multi level filtering (e.g assignment + semester)

Next Steps

- Admin Tab - Database query speeds
- Student Tab - Student specific statistics
- Add more tab functionality & login feature
- More ways to filter
- More Graphs
  - Box Whisker Plot
  - Gaussian Distributions
    - Mean + Standard Deviation
  - Difficulty, Avg. Skips

Demo