Li Lin
Ryan O’Flaherty
Nicholas Maresco
John Knollmeyer

Project Description: *Learning Analytics*

Vision and Goals Of the Project:

The Learning Analytics project aims to create an online platform which will perform data analytics on programming assignments useful for both students and teachers.

High level goals of this project include:

- Turning large amounts of programming submission data into useful statistics and information. (e.g. determining the different ways students approached and solved a problem)
- Effectively visualizing the analytics that are generated.
- Creating two separate views for with analytics built for instructors as well as students.
- Designing the application such that it is easy to integrate with.

Users/Personas Of The Project:

We are aimed to offer an online service to the academic community. Most assignment turn in portals offer very little beyond whether the submission is correct or incorrect. We want to tailor fit a platform for two different end users: instructors and students.

Students want feedback for their program submissions. On the other hand, teachers are an integral component of Learning Analytics. We want teachers to be able to ultimately gauge where their class is struggling and where they excel.

Ideally, Learning Analytics will be highly integratable with the ability to utilize our application in spaces outside the Learning Analytics web app. In addition, we will supply a highly sleek and user-friendly UI for both types of target users. It is also worth noting that we do not grade these assignments, but rather output a variety of analytic solutions.

Scope and Features Of The Project:

Functional:

User Interface

- Instructor View
    - The instructor view will consist of data analytics collected and computed for all student's submissions for each programming assignment given
    - Analytic dimensions
        - Time/Assignments
            - The instructor will be able to view the entire suite of visuals and analytics for any single student's progress throughout the entire semester. Each data point in time will be the assignment that was handed out for that week for that particular student.
        - Students
            - The instructor will be able to view the entire class' performance on any single assignment given. Each data point will correspond to each student in the course.
- Student View
    - The student view will essentially produce analytics and visuals performed on a subset of the entire dataset, specifically, the assignment data for that particular student. The dataset will consist of all programming assignments which the student completed.
    - Analytic Dimensions
        - Time/Assignments
            - The student will be able to view the entire suite of visuals and analysis for every assignment completed in that course. Each data point in time will be the assignment that was handed out for that week for that student.

Data Analytics and Visualization

Metrics / Patterns

These metrics will be compartmentalized into separate graphs in order to increase readability. The metrics we are going to begin with are as follows:

- Space Complexity
  We need to develop a space complexity estimation tool that allows users to gauge how much space it takes up when their code executes comparatively to their classmates.
- Data Structuring
  We want a way to figure out how are these students structuring their data. Are they using a heavy data structure involving multiple stores of the same information unnecessarily?

- Loop Counter
  Could this question be done with less loops?
- Nest Loop Count
  Some questions could be completed with less nesting. We would like to gauge whether the optimal solution is for example O(nlog(n)) vs O(n^2) if feasible.
- Time Complexity
  We need to develop a space complexity estimation tool that allows users to gauge how long it takes their code to execute comparatively to their classmates.
- Class Rank
  Using the analytic tests we devise, we can give weighted scoring, and ultimately creating a ranking system for each problem. We want users to be able to see their own performance analytic on an assignment level, and ideally we would like to support analytics on a course-level as well.
- Content analysis:
  - Attempt count
  - Comment count
- Comments vs Code
  Ratio of code to comment
- Statistics
  For more advanced analysis we can use probabilistic constructs. For example if user X did Y on problem A then they are likely to perform Z on problem B.
  - Conditional Probabilities
  - Probability Distributions
  - Probability Density Curves
- Clusters
  - How many different ways can we solve this problem statement (Solution Clustering)
  - What was the optimal solutions comparatively to the instructor solution
  - Are all students struggling in a similar area
- Popular functions in problem (trending functions)
  - Click on chart to see solutions
  - Be able to navigate to those solutions
- Activity Chart
  How many times did the student submit his or her code?
  - Time based activity and metrics
  - Dynamic / build in this versus time

Tracking progress over time

The tool will have the flexibility to track students' submissions over time in order to present metrics describing their progress throughout the course.

Real Time Updates

The analysis of each submission will be performed as each submission is entered into the system, providing real-time analysis and visualizations

Nonfunctional:
The design and architecture of the Learning Analytics application is guided by the following nonfunctional requirements. These requirements are the most critical in the success of the project.

Interoperability - The success of this project will rely on its ability to integrate with other systems.
- *Metric*: Develop a useful interface that allows for other software to easily integrate with the Learning Analytics application
- *Measure*: REST API architecture (client-server) will allow us to build a robust interface between the backend and many front end applications.

Scalability - The project needs to be scalable at both the application level to support more traffic as well as assignment level to handle increasing code submissions for each question.
- *Metric (application level)*: Create a backend architecture that will scale with increased traffic.
- *Measure 1*: Utilize AWS for scalability and load balancing. Use Express.js + Node.js to handle a large volume (50) of concurrent users. This number was generated to estimate an average class size.
- *Metric (assignment level)*: Handle increased submissions on each assignment as more students submit answers to the same problem.
- *Measure 2*: Use SQL to scale the data storage needed in the project and support the average number of submissions on a Cody assignment.

Usability - The final result of this project is intended to make learning more efficient, therefore ease of use is paramount to the success of the project.
- *Metric*: Graphs and figures must effectively display data and be understood when a user first interacts with them.
- *Measure*: Most (90%) of users should be able to understand what each graph and visualization is showing on the first viewing.

Expandability - The project is sponsored by an employee from Mathworks, and the codebase should be built with the likelihood of adding new features in mind.
- *Metric*: Support the ability to add new features to the application.
- *Measure*: Setup the Node.js architecture to support functionality for new modules. Design the API to support new modules without having to modify the API structure (e.g. part of the API call would be the module name).

Solution Concept:
Global Architecture Structure of Project

Front End:

We are going to generate a typical use case for our instructor and student users. Our app's frontend presents what our API would look like if integrated into external service.

Dimple.js/D3.js

We aim to utilize this javascript visualization library because it is powerful, flexible, and lightweight. The visualizations are truly the core of our application, and we want our product to be user-friendly, visually stunning, and inviting users of all types.

Bootstrap

We want to create dynamic and responsive front-end compatibles for users viewing from mobile, tablets, laptops, and desktops. Bootstrap is an incredible html, css, and js framework which will give us the power to do so.

React.js

We have decided to use react.js as our data rendering library, because we will be building a large application which needs to be capable of easy scaling over time as the number of problem statements, students, and submissions increases. We will also use the JQuery library for AJAX.

Back End:

Node.js Express Server

The backend server will provide a RESTful API for transferring relevant data stored in the database to the frontend application, as well as perform calculations for analytics which would be taxing on web clients. Express was chosen because

MySQL Database

A MySQL database will be used for storing the data regarding submissions, and potentially for caching analytics data so they do not need to be recalculated often. MySQL was chosen because its easy setup and popularity would make it easy for us as students to learn to use and deploy it, and we predict would fit well in the relational database schema of SQL.

Acceptance Criteria:

Bare minimum:

Utilize a given data set, we should be able to provide feedback analytically in the form of a teacher view and a student view. The flow of the website will be a home page containing all pre-loaded problem sets, and a search option if the number of data sets is very large. Off of the home page will be a problem set page specific to the problem clicked on the home page. The problem set page will supply two different views for the same problem. The student view and the teacher view. Imagine the same page with two tabs which offer two different perspectives on the same data visualization.

Essentially we have a web app given a large set of data, we need to analyze this data statically initially and then offer a teacher’s perspective or a student's view on that same data.

Unit testing:

We will conduct unit testing ideally, which involves compartmentalizing our application into small chunks and test each unit as a standalone component in order to strengthen the foundation our Learning Analytics as a whole.

Release Planning:

- **Demo 1 (2/9)**
  Basic Layout/UI is available, with different pages for users and instructors

  Instructor View: Users go to instructor endpoint, can see UI with list of assignments and open them
  Student View: Same as above, except with the student endpoint
  Assignment View: Opening an assignment shows an assignment UI with stubs in place of where the actual analytics displays would be

- **Demo 2 (2/23)**
  Visualizations and analytics for individual assignments are present on the pages using data hard-coded on the frontend (no communication with server yet)
  ...
- **Demo 3 (3/15)**
  The existing analytics are created using data which is loaded from the server
  ...
- **Demo 4 (3/29)**
  Add course analytics which are visible to the instructors
  ...
- **Demo 5 (4/12)**
  The server and database can be updated while running to identify changes over time and give real time analytics

  [Stretch Goal] Transfer database to Amazon DynamoDB and Express server to AWS Elastic Beanstalk