Spartan Score - Project Report

Naya Singhania
Department of Computer Science
San José State University
San José, USA
naya.singhania@sjsu.edu

Anushka Dhole
Department of Computer Science
San José State University
San José, USA
anushka.dhole@sjsu.edu

***Abstract*—This report explores the development of a grade calculator that utilizes the Tesseract Optical Character Recognition (OCR) library combined with a React front end in order to calculate a user's GPA from a user-provided screenshot of their grades. It will explore the reasons behind creating this grade calculator, technical development, challenges, and utilization of the object-oriented programming paradigm. Future development on this grade calculator will also be explored.**

***Index Terms*—React, Tesseract, Canvas, Instructure, LMS, Grades, GPA**

I. Introduction

The Canvas LMS, created by Instructure, is one of the most common LMS applications used across educational systems. It has a US market share of approximately 20% and is especially common in higher education [1]. This LMS currently lacks any sort of comprehensive grade calculation tool for students, making it complicated and tedious for students to keep track of their GPA throughout the semester. Current grade calculators have dated UIs and work off manual entering of grades, which is slow and error prone. As a result, students find it harder to maintain their grades and plan their semester, and this can become a major stressor for many students.

Spartan Score aims to be an easy way for students to calculate and track their GPAs efficiently by simply uploading a screenshot of their Canvas grades. We designed Spartan Score specifically for SJSU students to ensure our calculations are in line with the way grades are calculated at SJSU. We designed Spartan Score with industry standards libraries and frameworks, and ensured privacy by never storing uploaded grade screenshots.

II. Technical Aspects

A. React

React is a modern web library used to design the user interface and experience in a clean, modular manner, and especially excels in single page applications, such as Spartan Score. It also provides a way to implement much of the functionality of an app, as it uses application states and hooks to conditionally display content and perform functions. We used React as part of the Next.js web framework to ensure that time would not have to be spent on manually setting up build tooling, page routing, or SEO. Next.js not only supports React server components, but provides the ability to create server-side API routes, which would be very useful in implementing additional features in the future, such as tracking of student grades over time.

As Next.js is designed to work with Tailwind CSS, we decided to use the Shadcn UI library, which provides reusable and customizable React components styled with Tailwind's classes. This allowed us to quickly design our UI with prebuilt components and implement a color scheme that matches that of SJSU's color scheme. Our modular UI design also ensures easy maintenance of our codebase and gives us the ability to incrementally change or add features over time.

B. Tesseract

Tesseract is an industry standard OCR library that we utilized as a Node.js package in order to provide the ability to pull grades and calculate GPA from a screenshot. Tesseract was first developed by HP in 1984 and uses a traditional pipeline that was seen as unconventional at the time of its initial creation. Because of its great ability to work with well structured text with high contrast, we found it to be a great and effective option for our use case [2]. We implemented Tesseract on the client side to prevent the need to send potentially sensitive grade data to a server, and to ensure quick results.

C. Programming Paradigms

Spartan Score was created not only using the functional programming paradigm, but an object-oriented programming paradigm as well. While React tends to be more object-oriented, its close relation to JavaScript and its state management inherently makes it a functional library. State management in React happens from the component itself, and the inherent modular design of React ensures consistency and immutability across our code, which is closely aligned with the principles of functional programming [3].

We used a more object-oriented approach to processing outputted text from Tesseract, treating the output as an object and transforming it into a hierarchical object of grades. This object was then passed into a function that used the grade data to determine the overall GPA. While we could have taken a more functional approach to displaying grade info and calculating overall GPA, our object-oriented approach creates simplicity and allows for future developments.

III. Challenges

We faced a couple of big challenges while building Spartan Score, but have figured out solutions for these issues.

A. Pivoting

We were unable to carry out our original plan of utilizing the Canvas LMS API, as SJSU IT administrators were unable to provide us an API key due to security reasons. As a result, we were forced to pivot, but were luckily able to come up with an alternative plan.

B. OCR

We also had some issues with setting up the Tesseract OCR library at first, but were able to quickly resolve the issue and make full use of it.

C. Grade Weighting

One final challenge that we are still facing is that because Canvas does not store the unit weightings of classes, GPA calculations may be slightly inaccurate as we assume equal weighting of all courses. We hope to solve this by creating a scraper that will scrape unit weighting data of courses and return the unit weights to the front end. Then, we would be able to factor in the weightings into our calculations to ensure that our GPA calculation is correct.

IV. Impact

A. Efficiency

Using Spartan Score, students can calculate their GPA up to 90% faster than manual GPA calculation methods, enabling students to quickly check their GPA without losing valuable study time.

B. Accuracy

With Spartan Score, GPA calculation errors will be minimized, ensuring that students always have an accurate snapshot of their GPA.

C. Engagement

Giving students a simple and efficient way to calculate GPA ensures students are engaged with maintaining their grades. As a result, students will be able to focus on utilizing resources to ensure academic success.

V. Future Development

While we have been able to successfully implement the main feature of our project, we hope to further Spartan Score by adding a few features to enhance our main functionality.

A. Unit Weighting

As discussed in the previous section, we hope to solve one of our challenges by utilizing SJSU's course catalog. The course catalog has information on how many units each course is worth. By using a scraper to pull this data, we can then cross-reference it with the course that the student is in (via the screenshot of their grades they submit) to calculate overall GPA without assuming that all courses are weighted equally.

B. Tracking Over Time

We hope to implement a feature that allows authenticated users to track the changes in their grades and GPA over time by saving the student's grade data every time they check their grades. By doing this, we could also use the RAG features of an LLM to create personalized study plans for students based on classes that they are excelling or struggling in.

C. Browser Extension

Given that Spartan Score is currently a separate website and works on a user-inputted image, there is definitely a large amount of friction in using it. By integrating Spartan Score's functionality as a browser extension, we would be able to provide students with a calculated GPA with no manual input required for the student. This would make it much quicker for a student to check their grades without any extra steps.

References

[1] A. Aldiab, H. Chowdhury, A. Kootsookos, F. Alam, and H. Allhibi, "Utilization of Learning Management Systems (LMSs) in higher education system: A case review for Saudi Arabia," *Energy Procedia*, vol. 160, pp. 731–737, Feb. 2019, doi: 10.1016/j.egypro.2019.02.186.

[2] R. W. Smith, "History of the Tesseract OCR engine: what worked and what didn't," in *Document Recognition and Retrieval XX*, SPIE, Feb. 2013, p. 865802. doi: 10.1117/12.2010051.

[3] A. Banks and E. Porcello, *Learning React: Functional Web Development with React and Redux*, O'Reilly Media, Inc., 2017.