Joseph Gentile, Shi Jie Samuel Tan, and Isaac Wasserman
CS107
Instructor: Rajesh Kumar
December, 2019

Classroom Kit: A Journey and a Triumph

Our team set out to create a tool for keeping track of attendance and participation in the classroom. We wanted it to be easy for an instructor to configure once and use throughout the semester, and we wanted signing in and gaining points for participation to require little to no additional effort on students' behalf. We envisioned a classroom kit that would remove the bureaucracy and mundane tasks that stand in the way of productive interactions between instructors and students. By creating this classroom kit for the instructors in Haverford College, our team hopes to improve the quality of instruction in classrooms and promote a better learning environment.

Our Plan

To accomplish this, we landed on two means of data input: Bi-Co OneCards and Piazza, a forum based learning management system already used by students in a number of Haverford courses. Students would sign in to each class by simply tapping their OneCard on a device at the front of the room. The sign-in data would be stored in a database for the instructor to review in a GUI at a later date. On the participation side of things, the goal was to determine a score based on questions and answers found in Piazza. Each interaction with the forum would contribute to the score, and the weight of the interaction would be determined by the number of likes it had and the number of additional interactions that stemmed from it. Similar to attendance, these scores would be visible throughout the semester using a GUI of our creation.

Our plan required us to make use of three technologies no one on our team had any experience with. One of us would learn to build graphical user interfaces in Python; one of us would learn to interact with hardware components; and one of us would learn to scrape secure web application data. We acknowledged that the project had a lot of moving parts for something that had to be done in three weeks, but we decided to move ahead and cut elements where we saw fit as we neared the deadline.

Motivation

Computer Science professor Rajesh Kumar first voiced his desire for an alternative to the traditional circulating sign-in sheet mid-semester. Printing a class list, passing it around, collecting and transcribing it is a lot of work for such a procedural task. It takes up a disproportionate amount of time in an instructor's day for such a relatively inconsequential and mundane chore. Such time would likely be better spent planning lectures, grading assignments, or answering students' questions. Doing away with attendance tracking altogether, however, is not the answer. The results of a study performed at the University of Copenhagen shows students who attend class 100% of the time earn, on average, 10% higher grades than those who attend just 50-60% percent of the time.[1] Tracking participation the traditional way comes with its own issues. When class sizes hit a critical mass, it becomes difficult for instructors to keep track of students' names, let alone how many times they raised their hand in class. Participation grades are universally prone to imprecision and subjectivity, and they, too, occupy valuable time in an instructor's day. Similar to attendance, however, tracking participation is an effective method of

[1] Kassarnig, Valentin, Andreas Bjerre-Nielsen, Enys Mones, Sune Lehmann, and David Dreyer Lassen. "Class Attendance, Peer Similarity, and Academic Performance in a Large Field Study." *Plos One* 12, no. 11 (August 2017). https://doi.org/10.1371/journal.pone.0187078.

improving overall achievement in a class. A study from New York University found that high achieving students tended to be the ones who involved themselves more heavily in the class.[2] Our team went into this project acknowledging the critical importance of tracking attendance and participation, but in each instance, we knew there had to be a better way, a way that was easy, objective, sustainable, and extensible.

Accomplishments

We emerged from our journey with a self-contained device and ecosystem called Classroom Kit. Firstly, we built a database with methods tailored to our specific use case. With it, we can store daily attendance information on a file for an infinite number of courses as well as the information that allows us to access the class' page on Piazza. We then built a system to serialize and analyze data on each student found in Piazza. We applied an algorithm to this data that considers questions, likes, replies, and the endorsement of the professor to determine a student's participation. Lastly, we created a way to access and modify this data in the form of a GUI and an RFID scanner. With these tools, the user can configure their courses, allow students to sign in with their IDs, and visualize attendance and participation data in both a graph and a table format. Classroom Kit is a rough implementation of a tool with a real-world purpose that solves a real-world problem.

What Worked and What Didn't

From the very beginning, our team was certain that the number of moving pieces and aspects that had to work perfectly together for our project to come to fruition would pose

---

[2] Reddington, Cecelia, and Raymond Cañada. "How Does Student Participation Influence Student Achievement?" *Practitioner Action Research*, May 11, 2006. https://steinhardt.nyu.edu/departments/teaching-and-learning/research/practitioner-action-research/how-does-student.

significant problems. However, we saw this project as an opportunity to learn new techniques and practice what we had learned this semester, and whether we saw consistent success or faced setbacks along the way, we would have achieved our goal.

Before we commenced the project, we sat together to discuss our individual strengths and weaknesses. It was clear that we all had different amounts of programming experience and possessed different skillsets. While Isaac and Samuel both had a good amount of experience with developing web applications and GUIs, neither of them had experience with hardware or developing web scraping applications. Meanwhile, Joseph has had some experience with project conceptualization and user interactivity but has not developed GUIs or utilized data visualization tools. Hence, after balancing the priorities to achieve maximum learning and optimal performance, we settled on the following work distribution:

Joseph Gentile	Shi Jie Samuel Tan	Isaac Wasserman
GUI	Piazza Scraper	RFID Scanner
Data Visualization	Piazza Analysis and Score Generation	Database Class

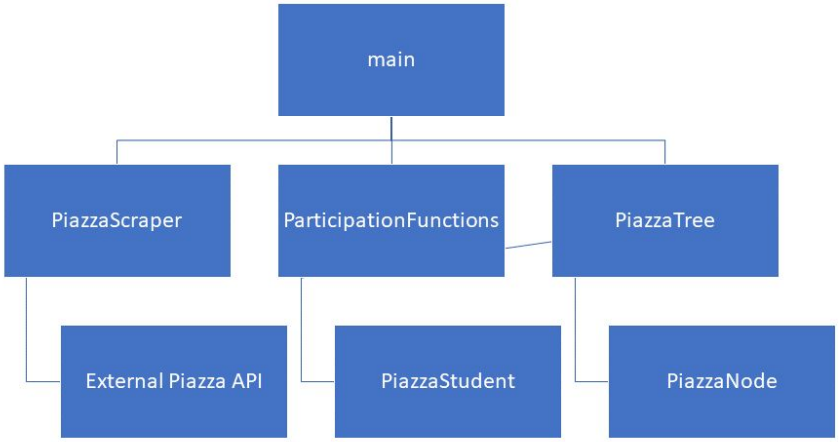
For the GUI, Joseph looked at several possible options before deciding to go with Tkinter (Python's de facto GUI). It was not only extremely compatible with Python but also very user-friendly. At the same time, Tkinter is widely used and has extensive documentation available online. For data visualization, Joseph decided to use Matplotlib, a plotting library for the Python programming language and its mathematics extension NumPy. Because it was Joseph's first time developing a GUI and using data visualization tools, we felt that it was sensible to use the packages which were most native to Python.

Joseph initially struggled with understanding how Python's Tkinter works. The online documentation was in itself self-explanatory, but utilized programming practices which were both different from what we learned in our classroom and antithetical to object-oriented programming practices. The templates and guides that were available online existed solely for the purpose of demonstration and not for large applications, and among other things often relied heavily on global variables, and when adapting portions of these templates into the GUI required reworking or otherwise finding ways to apply it without breaking it. Joseph's ultimate solution was to make heavy use of argument dictionaries most of which contain much of the data that was originally to be stored globally, or tying these values to the main GUI class. He also had to write a number of new methods when Tkinter's prebuilt methods were insufficient, like for updating the information on multiple pages, and had to extensively plan and consider how to organize the menus in such a way as to be intuitive and straightforward. The data visualization component was a lot more straightforward because we had already used it in our computer science lab work. Nonetheless, it was still a significant achievement for Joseph to be able to set up and utilize the libraries that are publicly available online.

Joseph's other challenge was communication. As the GUI exists as an interface between the user and the rest of the program, he needed to understand and be able to incorporate what both Samuel and Issac developed, and communicate which features or tweaks to their existing work will be necessary in order for the program to work as intended. It was ultimately a learning experience in programming at a wider project level, of combining all these disparate, independently produced parts of the project into a working whole, which necessitated extensive communication and planning.

For the Piazza Scraper, Samuel originally intended to build his own Piazza Scraper from scratch. Because he possessed some experience with sending and handling HTTP requests, he was fairly confident that he could set up a simple application that would be able to seek authorization from Piazza's endpoints and gain access to the HTML code from the Piazza webpages. From there, he would go through the HTML hierarchy to find the elements that contain relevant information for extraction. However, he realized that Piazza seems to be using a fairly primitive approach to handling HTTP requests. Instead of returning a 401 "Unauthorized" status code, Piazza simply returns a 200 "OK" status code even if the program has not gained authorized access.

After thinking about it and realizing that developing our own Piazza API is not the main focus of the project, Samuel decided to go online and find an existing Piazza API. Even though he was able to find an unofficial Piazza API made by another programmer as his own side project, the package had no documentation, and Samuel was forced to read through the codebase to learn to use the tool. Because the creator of the API wrote minimal comments, Samuel had to try out the different functions and deconstruct the person's program to understand how it works. He also had to ensure that no private information was being collected by the API to make sure that it was safe for use. After which, he built a custom PiazzaScraper class that modifies the Piazza API to fit the context of our application.



When it came to the Piazza analysis and scoring algorithm, Samuel decided to go with a customized node and tree data structure with level order traversal as its main means of traversal. A customized node (instead of the standard node we used in labs) was necessary because of the unique set of data that it contains as well as the one-to-many relationship it had with other nodes. A tree was also ideal because it gave us a structure as to how we could map out the relationships between the root, the nodes, and their respective children. Samuel decided to set up the tree recursively from the data that was returned by the Piazza Scraper. By laying the different threads out in the root of the tree and then recursively finding the children to each node and instantiating them, Samuel was able to set up the tree. Following which, the tree would be traversed using a queue and the level order traversal method to establish PiazzaStudent profiles that would contain the number of likes they had from instructors and students, the number of posts and comments their posts had spun off, and the number of posts and comments which they had written. This

collection of data allowed our Piazza algorithm to compute the class participation score. Samuel also developed a class participation report using a dictionary which would be returned from a modular function called from his partners' code.

For the RFID scanner, Isaac tried to tap onto our school resources by contacting VCAM to see if they have any existing breadboards, Raspberry PIs, or RFID readers and cards that we could borrow for this project. Unfortunately, they only had alternative microcontrollers that were not very suitable for our RFID scanner. At the same time, they did not have any RFID readers or cards that we could use. Hence, Isaac procured the Raspberry PI and the other components on his own. Because he was unaware that there are two variants to RFID readers (13.56 MHz and 125 kHz), Isaac ended up with the wrong RFID reader for our project (13.56 MHz instead of 125 kHz). Isaac decided to stick with the one he bought because of the time constraints that we were working with. By going through online video tutorials and manuals, Isaac was able to learn how to operate the Raspberry PI and solder the reader with the help of VCAM resources. It was a great experience for Isaac, who normally only works with software, to attempt to build the hardware for our project without much assistance.

For the database, Isaac was originally looking at Firebase and MySQL as possible options. However, after putting some thought into it, he figured that he wanted to implement a database by himself rather than use API functions to handle storage and retrieval of data. Hence, Isaac decided to store the data collected from the Piazza Scraper and the RFID scanner in a locally stored JSON file. He would convert the data from its JSON structure into a dictionary when the applications requested the data. Vice-versa, he would convert the dictionary data back

to JSON after it was mutated by the application. Isaac chose to use a dictionary because of its O(1) time to access any value, its mutability, and how it could be converted into JSON easily.

Other than our individual components, we also experienced some difficulties working in a team. Because of the different ways we type code, we struggled with understanding each other's code—especially when all of us use different naming conventions and structure our applications differently. However, we all had the good habit of commenting our code so it was ultimately not too challenging for us to interpret each other's code. Another challenge that we faced working together was how we failed to communicate some of the finer requirements that we had for each of our components. Joseph's Tkinter GUI could only run safely on a single thread, but Isaac's RFID scanning class was designed to operate on its own second thread. This problem only surfaced when we were trying to assemble all the components together. Even though it was not overly troublesome to reconfigure some of our methods and functions, the consequences could have been far worse if it had pertained to something more fundamental to our application. The moral of the story is to understand and communicate requirements effectively; it would have been better if the team had sat together to collectively define and adopt programming conventions before commencing any form of application development.

Conclusion: The Future of Classroom Kit

Our goal for Classroom Kit was to develop a proof of concept for a tool that handed off procedural and mundane tasks from instructors to computers that specialize in the procedural and mundane. In that sense, Classroom Kit was a success. In addition to polishing the toolkit and

installing a 125kHZ scanner for reading real OneCards, there are a number of improvements that we believe would improve its utility. The first of which is output capabilities. Having a spreadsheet or a bar graph is great for momentary visualization, but when it comes to entering data into a learning management system (LMS) or gradebook, Classroom Kit is currently no better than a sheet of paper. At the very least, we would like to add CSV exporting functionality, but to create a truly seamless workflow would require integrating the software with the LMS itself using an API. We also acknowledge that scraping Piazza is not the most comprehensive way to track class participation. The only way to get a true sense of who is contributing to the class is to track what happens in the classroom. In the future, we would like to research using visual or auditory cues to gather participation during class time. The final addition we would like to make is trend tracking, something especially useful in middle and high school settings. The idea behind it is that tracking participation shouldn't be solely about grades. Taking notice when students aren't contributing and encouraging them to ask questions and participate is an important part of creating a sustainable, productive, and inclusive learning environment, but instructors are only human and might not notice the student in the back who is completely lost. By tracking contributions, recognizing trends, and notifying instructors that a student might need some attention, Classroom Kit can help struggling students gain a better education. With Classroom Kit 1.0, we sought to relieve instructors of burdensome, objective, and procedural tasks to give them time to focus on the more important aspects of education, but in the future, we want Classroom Kit to help fill the gaps in the classroom that instructors might not even notice. We believe tools like this have a real future in classrooms of tomorrow.

Packages Used and Code Borrowed

- piazza-api
    - https://github.com/hfaran/piazza-api
- objgraph
    - https://github.com/mgedmin/objgraph
- matplotlib
    - https://github.com/matplotlib/matplotlib
- mfrc522-python
    - https://github.com/pimylifeup/MFRC522-python
- JSON validity checker: Eric Leschinski
    - https://stackoverflow.com/questions/5508509/how-do-i-check-if-a-string-is-valid-json-in-python
- How to switch frames: Bryan Oakley, @Baller, Terry Jan Reedy
    - https://stackoverflow.com/questions/7546050/switch-between-two-frames-in-tkinter