Prototypes Report

Product Name: All Aboard Attendance
Team Name: The Bolts
Date: July 21, 2019
Team Members: Donghyun Kim, Matt Jacobs, Suleyman Saib, Trevor Weger, Wesley Smith

Known Problems and Bugs

1.Test server(localhost) sometimes has a bug at the admin page. It occurs when we move from homepage to admin page, and error's name is ConnectionAbortedError. The document says this error is due to a data transmission time-out or protocol error. Also, because this error happens in only one person's laptop, we think this is Django issue or problem with our local machine more than implementation fault.

2. URL address issue in instructor page when not logged in. Clicking Course and Admin button on navbar should redirect to localhost:8000/course and localhost:8000/admin, but our URL looks like this: "localhost:8000/accounts/login/?next=/course" and same happens for admin. It doesn't violate our functionality since it leads to the correct page, but the URL isn't generated in the way we thought. I think it's due to our views.py and urls.py has some errors in handling those requests, and fixing those can be a solution.

3. Long lecture titles can occur overflow in the lecture index page. Because we can add a lecture by typing their from our web page, a string that has too long length can destroy the web page's shape or alignment. This can be fixed by limiting the length of the lecture's name when handling the request.

4. Student login is not secure. For instructors, we used Django's built-in functionality for authentication and blocked access for not logged in users. But for students, we just check whether their CruzID is in the course's student list when they scan the QR code. This may be problematic if some student tries to cheat the attendance system. We should devise an advanced authentication system for students to solve this problem.

5. Due to the 15min time limit starts from every time we create a new lecture, there are some limitations in creating and handling attendances. The instructor has to create lecture at right at the beginning of each lecture, and there cannot be a late attendance for students. To solve this, we could set date and time for our timer to start, or add an auxiliary feature to add late attendance manually.

6. Timezone field defaults throw warnings on LocalServer execution. Python datetime assigns a static value - a dynamic date system should be used in future.

7. Admin page occasionally hangs from an attempt to query a null input (NoneType error). This may be caused by the development server package provided by Django. It does not seem to affect the webapp's function beyond requiring a forced refresh of the local server. The cause of this error is not fully diagnosed, and if it is found to be the development server package, moving to Apache or a production server package will resolve it.

8. The lecture informatic page does not display the correct URL for the development server sign-on. The correct URL varies by machine, but is generally a localhost URL tree. Hosting from a production server with a legitimate domain and matching the displayed URL will fix this.

9. The student sign-in page prompts the user for a CruzID, but Student creation does not necessarily require a CruzID be used. In the future, the sign-in page prompt could match the institution's student ID context. For instance, if they use a numerical ID, the prompt would reflect this and direct the user to where they might find the number on their ID card.

10. Well-connected networks of large size produce illegible graphs. In the future, offer to export the NetworkX data for the user to use with more advanced visualization packages. Additionally, implementing a network visualization library such as GraphViz will allow more sophisticated visualizations within the webpage.

11. Graphs are rendered too many times, putting excess stress on the server. To fix this, rendered graphs should only be produced on request by an Instructor. Currently, they are re-rendered every time a student joins the network.

12. Student connections do not update without a manual refresh. Implementing a live, continuous feed of connections should be investigated. Similar continuous feeds could be applied elsewhere in the website.

13. Editing the instructor from the admin page does not successfully change the alteration. This is likely caused by a misapplied "Editable=false" tag, but must be investigated further.

14. Deleting a previously opened course does not affect rendered images related to that course. In the future, consider compressing the lecture graphs and data from closed courses into archives to limit storage blowup.

15. CSRF tokens are sometimes lost when new lectures from different courses are opened. If a student attempts to log another student's ID when this error occurs, they will receive a 404 instead of a successful refresh of the sign-in page. The cause of this requires further investigation, but it may be caused by conflicting paths in the URL namespace.

16. Identical student IDs may be provided to the database for the same course. In the future, student IDs should be tagged as unique, and a fitting error thrown if this condition is violated.

17. Some integrity errors are mishandled. In the admin screen, creating courses with the same title will cause a full redirect to an error page, as opposed to a more suitable form error. This problem exists in any context with a unique field constraint, but no matching form constraint.

18. Creating lecture titles containing only spaces creates a Reverse Match error. This will effectively break the lecture list page for a course, as a Reverse Match error will be thrown whenever the lecture with page attempts to display the errant title's null slug. Either implementing a form error or rewriting the lecture_title_slug generation to accommodate non-null all-space titles will resolve this.

19. The session state field does not contain relevant information on the lecture's status. To fix this, the lecture should be shown as "closed" if student sign-in form submissions are no longer accepted and "open" if students can still sign in.

20. Runtime blowup occurs during creation of lectures for courses containing more than 10,000 students. This is a result of naive collision checking during temporary ID generation. In the future, efficient hash tables should be implemented instead of iterative code generation-and-checking.

21. Runtime blowup occurs when students in large classes log one another's ID at once. In general, the smaller the lecture (<100) the more students can sign-in at once (20+) without noticeable slowdown. For larger lectures, the slowdown grows exponentially. This can be resolved by relocating the graph render function call to a location where it is not run excessively.

22. The network render does not scale with the number of students. The current canvas size tolerates signed-in student quantities of ~150 students well, but for significantly less or more the canvas is inappropriately scaled. In the future, render DPI and frame size should be adjusted dynamically in response to the number of students signed in.

23. Edge weights are currently not assigned in the network renders. It would be wise to derive an edge's weight from the time the edge formed, or the frequency with which the edge forms in other lectures.

User Stories/Acceptance Criteria

- As a student, I want to scan a projected QR code so that I can quickly access the session attendance webpage and log in.
    1. QR code is generated and displayed for each lecture.
    2. QR code correctly redirects student into signup page for lecture.
- As a student, I want to exchange a temporary ID with other nearby students so that I can be marked present.
    1. ID is uniquely generated and given to the student
    2. Field to input neighbors ID
    3. Confirmation message for successful student attendant

- As an instructor, I want to be able to host a unique attendance session for each lecture so that I can track attendance throughout the quarter.
    1. Button to create a new lecture instance for a course
    2. Newly created lectures are added to course's list of lectures
    3. Clicking on a lecture redirects to the lecture page with QR code
- As an instructor, I want a simple list of all students marked present for a session.
    1. Lecture detail web page with all students in the course.
    2. Separated tables for student's status(present, absent)
- As a professor, I want a page to manage and create courses so that attendance is organized by each course
    1. Admin page to manage courses created by an instructor.
    2. Button to add/remove courses.
- As a professor, I want a page for each course to manage and create lecture sessions so that I can take attendance or see past attendance.
    1. Web page for listing lectures associated with a specific course.
    2. Button to create lecture session.
    3. Lecture detail page for viewing attendance records in that lecture.
- As a researcher or instructor, I want to click a button to generate a graph based on a course's lecture data
    1. Graph that displays connections between students
    2. Useful information on graph on connection strengths and time created
    3. Button on lecture page to generate graph that can be reused to update information
- As an administrator, I want to be able to create courses, teacher profiles, and be able to associate them together so that I can organize my school.
    1. Teacher creation implemented through admin
    2. course creation implemented through admin
    3. Field to associate a teacher while creating a new course
- As faculty, I want to log in and see a list of my courses so that I can manage them.
    1. Web page displaying list of courses.
    2. Feature to filter courses by its instructor that only instructor-specific courses are displayed.
    3. Page for adding and removing courses.
- As a student, I want to be able to follow a link to sign into my class, allowing me to be marked as "present."
    1. Unique link is generated for each lecture
    2. Link, or QR code that contains link sends student to signup page.
    3. After signing up with CruzID, student is marked as "present".
- As any user of the app, I want a clean and consistent UI so that I may be able to more easily use the app.
    1. Every page has a navigation bar on top of the page.
    2. Color and shapes of the button is unified.
    3. Blocks are correctly aligned.