BoilerBoard.

Team 21 - Product Backlog

Arunav Sen Choudhuri, Arman Kumar, Soham Sharma, David Cox, Maximus Beato

Problem Statement

A prevalent issue among Purdue University students is the lack of a centralized platform for academic discussions, questions/answers, and resource sharing. This leads to fragmented information, making it challenging for students to find relevant academic help quickly. Our project, BoilerBoard, aims to address this by creating a Purdue-specific educational website that facilitates academic discussion and resource sharing for Purdue courses. Unlike existing services, BoilerBoard will be tailored to Purdue University's curriculum and community, providing unique features to students to enhance and personalize their student experience. Students will be able to create and join modules that correspond to Purdue courses and Instructors will be able to moderate these modules. These modules will consist of student-created learning materials, Q/A sessions, discussions, and voice-chat rooms.

Background Information:

Audience

As modern education shifts towards being more online, it makes sense to have an accessible studying platform to maximize studying efficiency for Purdue students and optimize student/instructor communication. Although there are several study resources available for Purdue students, there is not one platform that can fulfill every study need while also allowing instructors to communicate directly with their students in a public forum format.

Similar Platforms

There are a few websites that attempt to fulfill a similar niche as BoilerBoard such as Piazza, Ed Discussion, and Boiler Exams. Piazza and Ed Discussion aim to help Purdue students by facilitating classroom-wide discussions where TAs, instructors, and other students can reply to questions. BoilerBoard works to help students study by providing previous exams in an easy-to-use quiz format.

Limitations

These other services are useful; however, we believe they could be more beneficial if they were combined into one application. One of the limitations of Boiler Exams is that if you have a question about a specific problem there is no forum to ask questions and get help. Additionally, there is no way to expand Boiler Exams by adding new questions and keeping the courses up to date with recent material. We believe by combining a forum-based application with a service like Boiler Exams we can create a website that brings the best of both worlds.

Functional Requirements:

1. As a user, I would like to create an account with a Purdue email address.
2. As a user, I would like to log in to my account and manage my info.
3. As a user, I would like to be able to reset my password if I forget it.
4. As a user, I would like to be able to post learning materials for a course in the form of text and images.
5. As a user, I would like to make my own avatar for my profile.
6. As a user, I would like to be able to view my profile showing my account information and past posts that I have made.
7. As a user, I would like to be able to make practice quizzes that other users can take.
8. As a user, I would like to upvote certain posts and quizzes
9. As a user, I would like to have a scoring system based on the number of upvotes I get on my posts and practice quizzes and have other users view my score.
10. As a user, I would like to be able to sort discussions by the most recent or the most upvoted.
11. As a user, I would like to be able to edit and delete my own posts.
12. As a user, I would like to navigate from one page to another easily.
13. As a student, I would like to be able to easily join and leave courses.
14. As a student, I would like to be able to read and interact with other users' learning materials for courses I have joined.
15. As a student, I would like to take quizzes other users have made and immediately get the results after submitting them.
16. As a student, I would like to be able to view my quiz performance history to track my learning progress.
17. As a student, I would like to be able to make discussions in a specific course where other students in the course can respond to my question.
18. As a student, I would like to be able to mark a correct answer to my discussion question.
19. As a student, I would like to be able to create a voice chat that I can study and chat with fellow students.
20. As a student, I would like to be able to view voice chat rooms in session, and request to join.
21. As a student, I would like to be able to have access to a virtual whiteboard during voice chat sessions to enhance our collaboration experience.
22. As a student, I would like to view different modules within a course to better fit what I am learning.

23. As a student, I would like to be able to search course materials to find information that is relevant to me.
24. As a session leader, I would like to be able to change my session between private and public.
25. As a session leader, I would like to be able to invite students to my public or private session.
26. As a session leader, I would like to be able to remove students from my session.
27. As a session leader, I would like to be able to mute and unmute students in my session.
28. As a session leader, I would like to be able to have my sessions hidden from designated blacklisted users.
29. As a session leader, I would like to be able to clear the virtual whiteboard for all users in the session.
30. As a session leader, I would like to be able to remove whiteboard permissions from certain users in my session.
31. As a session leader, I would like to be able to schedule sessions in advance and send out invitations to participants.
32. As an admin, I would like to create courses and course modules.
33. As an admin, I would like to remove other users from a course in the event of repeated misbehavior.
34. As an admin, I would like to remove posts that are reported by other users or posts that can be considered as “unsafe”.
35. As an admin, I would like to be able to manage user roles and permissions within a course.
36. As an admin, I would like to be able to update course information and details.
37. As an instructor, I would like to be able to endorse posts and practice quizzes.
38. As an instructor, I would like to view my course’s discussion to ensure Purdue’s honor code is upheld.
39. As an instructor, I would like to remove any posts that could be considered academic dishonesty.
40. As a user, I would like to receive notifications when someone responds to my posts or discussions.
41. As a user, I would like to be able to follow specific courses and receive updates about new posts and discussions.
42. As a user, I would like to be able to report inappropriate content to the moderators.
43. As a user, I would like to be able to bookmark certain posts or discussions for easy access.
44. As an instructor, I would like to be able to post announcements to all students in a course
45. As an instructor, I would like to be able to provide official answers to student questions in discussions.

46. As an instructor, I would like to be able to assign badges or rewards to students based on their participation or performance.
47. As a user, I would like to be able to provide feedback on the platform's features and report any issues or bugs I encounter.

Nonfunctional Requirements:

Architecture and Performance

BoilerBoard will be structured into two main components: the front end and the back end. The backend will be written in Django which is a Python web framework. Django features rapid development capabilities, security features, and scalability which will allow our web application to be robust. Django also features robust security features out of the box, helping us to avoid common security mistakes such as SQL injection and clickjacking.
We will be using PostgreSQL for our database needs. PostgreSQL is a powerful relational database system that uses the SQL language combined with many features that can safely store and scale complicated data workloads.

The frontend will be developed using React which is a JavaScript framework for building user interfaces. React allows us to create reusable UI components, significantly speeding up our development process.

The combination of Django, PostgreSQL and React will provide a robust, scalable, and high performance solution for both the frontend and backend of BoilerBoard. This tech stack will ensure that we can effectively meet the needs of our users and provide them with a reliable experience when using our web application.

With this in mind, we have some performance goals. As a developer,

a. I would like this application to run smoothly without crashing.
b. I would like this application to be launched in 5 seconds.
c. I would like this application to support 20,000 users.
d. I would like requests to be handled within 500 ms.

Security

Having good security measures is important for BoilerBoard, as there is sensitive information tied to users' accounts. The Django framework that we will be using comes with several security features to prevent common exploits such as cross-site request forgery (CSRF) protection and SQL injection protection. We will also implement different roles for users and a permissions system to ensure users can only access functions in which they should be able to. All requests to APIs will need to be authenticated to prevent abuse.

Usability

The User Interface (UI) will be easy to use and navigate. The extent of our features requires the website to have an easy-to-understand interface that any user can use. Similar platforms have completely separate functionality and interfaces, but since this will be a central platform, we have to emphasize usability even further than our competitors for students, instructors, and administrators. Instructors and administrators will have a similar interface to the students but with additional capabilities. We want to ensure that our interface is easy to use on all resolutions, screens, monitors, and browsers for a universal experience.

Hosting/Deployment

Our front and back ends will be able to be deployed and updated separately. As we progress through the development process, we can deploy the back end to a DigitalOcean server to go through testing. The front end can be hosted for free on a service like Vercel.