## 

## Team 22 - Product Backlog

*Mahad Faruqi, Justin Lai, Henry Mayer, Nicholas Powers, Zachary Sy, Joshua Yang*

## Problem Statement

Students at Purdue University struggle to find the optimal schedule that balances the various factors of their day. BoilerTime eases this by ingesting user preferences and matching it with a variety of data sources to find the optimal class sessions based on student feedback, distance between buildings, bus times, previous grade distributions, and friends’ schedules. As students already tend to closely reference individual services separately such as Rate My Professors, Google Maps, and BoilerGrades during class registration, this service combines them all into one application interface and provides a detailed analysis to recommend a complete schedule.

## Background Information

### Target Audience

Students at Purdue University need a single platform they can use to build their perfect schedule. Existing options are limited, with no single platform providing all the necessary information in one place to create the best schedule possible. All students at Purdue can benefit from using this service.

### 

### Similar Platforms

There exist individual platforms to help schedule classes like Rate My Professor, Boilergrades, and UniTime. These platforms can help students decide which sections of their classes to enroll in, for example Boilergrades can give an overall gpa of a professor. We aim to combine these platforms and create a software that can optimize a student’s class schedule without struggling to search all these sites themselves.

### Limitations

Existing services are each limited because they only provide a small amount of data. RateMyProfessor is limited because its reviews shed little insight on the actual class grade outcomes, BoilerGrades is limited because it provides no context for the grades that we given to students in the past, Google Maps is limited because it’s difficult to make a quick map of where all classes are and consider how much time it will take to get between them, and UniTime is limited because it provides no context to the class options that it presents. BoilerTime fixes all of these issues by creating a single application that unites the various data sources together and uses an algorithm to analyze the available options and make informed schedule recommendations to Purdue students.

## Functional Requirements

1. As a student, I would like to register for a BoilerTime account using my *@purdue.edu* email.
2. As a guest, I would like to be able to use limited features without providing my *@purdue.edu* email.
3. As a user, I would like to ensure my email address is verified before using the website.
4. As a student, I would like to be able to reset my password if I forget my login details.
5. As a student, I would like to be able to create a profile that contains my name, profile picture, year at Purdue, and other information about me
6. As a student, I would like to input my class choices and categorize them into required/major and elective/flex/interest classes.
7. As a student, I would like to be presented a filtering by tags or categories feature for the classes.
8. As a student, I would like to be presented with a live search by keywords feature for the classes.
9. As a student, I would like to be notified if my required/major classes outright do not work together.
10. As a student, I would like my schedule to contain all of my required class choices.
11. As a student, I would like classes that are not already full.
12. As a student, I would like to bookmark certain classes.
13. As a student, I would like to be notified through email if a slot opens in one of the classes I hoped to take.
14. As a student, I would like to see a calendar view of my schedule.
15. As a student, I would like to see a list view of my schedule.
16. As a student, I would like to switch between a calendar and list view of my schedule.
17. As a student, I would like to be able to find a schedule that considers all of my preferences in one.
18. As a student, I would like to be able to prioritize my classes based on the time of day.
19. As a student, I would like to see the past average GPAs of classes based professors.
20. As a student, I would like to see the peer ratings of each of my professors.
21. As a student, I would like to see the route between each of my classes.
22. As a student, I would like to create a schedule that prioritizes minimum walking distance.
23. As a student, I would like to create a schedule that prioritizes overall professor rating.
24. As a student, I would like to be able to specify my residence hall or bus route.
25. As a student, I would like to plan around other meetings, or TA sessions outside of my Purdue class schedule.
26. As a student, I would like to be able to see rankings of different schedule options.
27. As a student, I would like to export my schedule to external calendar apps.
28. As a student, I would like to be able to export my schedule as a photo or PDF.
29. As a student, I would like to be able to interact with my schedule on the calendar.
30. As a student, I would like to see the class distances visually through a map with locations of classes marked.
31. As a student, I would like to be able to view classroom ratings and teaching assistants.
32. As a student, I would like to be able to write classroom ratings based on the convenience of access, quality of seating, and availability of technology in the classroom
33. As a user, I would like to be able to rate a class, based on the strictness of prerequisite requirements, the pace of materials covered, the depth of the materials covered,
34. As a student, I would like to be able to edit and delete my past ratings.
35. As a student, I would like to be able to flag and report malicious ratings.
36. As a student, I would like to be able to write ratings of teaching assistants based on helpfulness of answering questions, responsiveness, and fairness of grading.
37. As a friend, I would like to join a group so that I can have classes with friends.
38. As a user, I would like an easy way to create a group for my friends through unique links.
39. As a friend, I would like my group’s schedule to match with mine for certain classes.
40. As a friend, I would like to be able to see all of my friend’s schedules on one screen.
41. As a user, I would like to be able to share a created BoilerTime group .
42. As a guest, I would like to have my schedule saved on my browser, even if I am not logged in.
43. As a user, I would like to be able to find other BoilerTime users in my classes if they consent (i.e. the “discover” functionality on social media platforms).
44. As a user, I would like to see an introduction page to the website so that I can learn about the special functionalities of the website.
45. As a user, I would like to be able to access my collected data so that I can track my digital footprint.
46. As a user, I would like to be able to change my password from my account settings.
47. As a user, I would like to be able to customize the accent and themes of the website.
48. As a user, I would like to be able to have two-factor authentication on my account.
49. As a user, I would like to be able to create unique links to share my BoilerTime schedule.
50. As a professor, I would like to be able to see feedback from my students.
51. As a user, I would like to be able to provide feedback on my generated schedule and errors that I see in it.
52. As a student at another university, I would like to be able to port the BoilerTime service to my university and host it on my own infrastructure.
53. As a user, I would like to see recommended classes that people have taken together.
54. As a user, I would like to see a list of trending and popular classes.
55. As a user, I would like to import and view my previous schedules in my profile.
56. As a user, I would like a limited functionality version of the platform for use on a mobile or smaller-screen device.
57. As a user, I would like to be able to set preconditions of times that I need to reserve, like religious prayers (ex. Friday Prayers for Muslims has two times on Friday).

## Non-Functional Requirements

### Architecture and Performance

1. A Java algorithm that analyzes schedule options to create recommendations
2. A Vue.js front end to dynamically render a data-driven user interface on the client side
3. A middleware built on Node.js to handle request-response cycles and dependencies
4. Integration of a cloud-based analytics tool to understand user behavior
5. Separation of the front end and back end for modularity and maintainability
6. Data storage and retrieval through various API endpoints
7. Use of caching strategies to minimize redundant requests
8. Use of load-balancing strategies to ensure support for concurrent requests and at least ~1,000 simultaneous users
9. Use of optimization strategies to ensure average initial website load time across devices is less than 5 seconds
10. Use of prerendering strategies to ensure time to first byte when navigating across functionality is optimized for network and device conditions

### Security

1. Implementation of rate limiting to prevent excessive or malicious API requests
2. Requiring CAPTCHA verification for guest users
3. Automated logging and flagging of user requests and session details (i.e. user agents, IP addresses)
4. Compliance with data privacy regulations
5. Requiring a client generated JSON web token (JWT) to access the back end’s API endpoints
6. Ensuring that user data is encrypted both in transit and at rest

### Hosting/Deployment

1. Deployment of the back end to a cloud infrastructure for close to 24/7 end-user availability and easy future scalability
2. Integration of a cloud-based monitoring tool for resource utilization and uptime to allow quick response to performance or security issues
3. Regular backups of user data to protect against data loss and corruption
4. Use of version control to manage the platform’s code changes
5. Proper configuration of a firewall on the back end to restrict access

### Usability

The interface should be clear to understand and simple. Having the options to navigate the website while also minimizing the amount of pages required to use the website. When creating a schedule, users should have clear indications of the selections they have made before and after constructing the calendar so that it can be easily referred to. Being able to refer back to one’s schedule should also be a possibility through saving the schedule and associating it with the user’s login will make ease of access and avoid having to recreate the schedule each time.

1. Responsiveness for use on different devices and screen sizes
2. Adherence to web accessibility standards
3. Average interface response times of non-algorithmic dependent functionality should be around 200 milliseconds and not exceeding 1 second