BoileResources

Sprint 1 Planning

CS 307: Team 25

Created by Arya Shukla, Suhesh Venkatesh, Pranavi Chaganti, Zain Sohail

Sprint Overview

In this sprint, we are aiming to implement core features for our application. This includes user registration, authentication, password reset, and profile management. Also, we aim to implement class-related functionalities like adding, editing, deleting, and viewing class details. Additionally, various UI/UX customizations like theme toggling and marking classes as completed are also included.

Scrum Master:
Suhesh Venkatesh

Meeting Plan:
Monday 10:30am-11:30am with Project Coordinator
Monday 1:30pm - 2:30pm
Tuesday 3:00pm - 4:00pm
Thursday 1:30pm - 2:30pm

Risks and Challenges:

For our sprint, the biggest challenge is getting the project up and running, especially with our limited experience with MongoDB and Node.js for backend development. This sprint provides a significant learning curve. Since there's also a deadline with the work to be done, we need to manage our time well and ensure that our work is completed. This is especially important in this sprint, as we are just starting to build our project and may not be able to anticipate what roadblocks will come up.

Current Sprint Detail:

1. As a user, I would like to be able to register for a BoileResources account.

#	Description	Estimated Time	Owner:
1	Create MongoDB user schema	0.5 hour	Suhesh
2	Build registration form UI with React	2.5 hours	Suhesh
3	Implement form validation (frontend)	1.5 hours	Suhesh
4	Create backend API endpoint for registration	2.5 hours	Suhesh
5	Implement password hashing with bcrypt	1.5 hours	Suhesh
6	Add email verification	4 hours	Suhesh
7	Error handling and user feedback	1 hours	Suhesh
8	Integration Testing	1.5 hours	Suhesh

Total: ~15 hours

Acceptance Criteria:

1. Given a new user is on the registration page, when they fill in valid registration details and submit, then their account is created and a verification email is sent/
2. Given a user has received a verification email, when they click the verification link, then their account is verified and they can log in.
3. Given a user attempts to register with an existing email, when they submit the registration form, then they receive an error message indicating the email is already in use.
4. Given a user enters mismatched passwords, when they attempt to submit the registration form, then they receive an error message and cannot proceed.
5. Given a user enters an invalid email format, when they attempt to submit the registration form, then they receive an error message about invalid email format.
6. Given a user has logged in after signing up, they will be taken to the user dashboard/profile.

2. As a user, I would like to be able to log in to my account.

#	Description	Estimated Time	Owner
1	Create login form UI	3.5 hours	Zain Sohail
2	Implement Google OAuth authentication	1 hour	Zain Sohail
3	Create login API endpoint	2.5 hours	Zain Sohail
4	Add session management	2.5 hours	Zain Sohail
5	Implement "Remember Me" functionality	2 hours	Zain Sohail
6	Error handling and user feedback	1.5 hours	Zain Sohail
7	Testing authentication flow	2 hours	Zain Sohail

Total: ~15 hours

Acceptance Criteria:

1. Given a verified user enters correct credentials when they click the login button, they are successfully logged in and redirected to their dashboard.
2. Given a user enters incorrect credentials when they attempt to login, they receive an appropriate error message.
3. Given a user selects “Log in through Google” when they are on the login screen, they are prompted to selected a Gmail account to log in with as opposed to logging in directly with their email.

4. Given a user is logged in, when they close the browser and reopen it, then they remain logged in if "Remember Me" was selected.
5. Given a user is logged in on multiple devices, when they make changes on one device, then the changes are reflected across all devices.

3. As a user, I would like to be able to reset my password if I forget it.

#	Description	Estimated Time	Owner
1	Create forgot and reset password form UI	2 hours	Arya
3	Implement email service integration	1 hour	Arya
4	Create password reset API endpoints	2.5 hours	Arya
5	Add reset token generation and validation	2.5 hours	Arya
6	Error handling and user feedback	1.5 hours	Arya
7	Testing reset flow	2 hours	Arya

Total: ~12 hours

Acceptance Criteria:

1. Given a user clicks "Forgot Password", when they enter their email address, then they receive a password reset link via email.
2. Given a user clicks the reset link, when they enter a new password, then their password is updated and they can log in with it.
3. Given a user enters mismatched passwords in the reset form, when they attempt to submit, then they receive an error message.
4. Given a password reset link has expired, when a user attempts to use it, then they receive an error message about link expiration.
5. Given a user changes their password, when they attempt to log in with the old password, then access is denied.

4. As a user, I would like to manage my account to change personal details (position, grade, major, etc.).

#	Description	Estimated Time	Owner
1	Create profile UI with editable fields	3 hours	Pranavi
2	Implement form validation	1 hour	Pranavi
3	Create profile update API endpoint	0.5 hours	Pranavi
4	Add data validation on backend	1 hour	Pranavi
5	Implement success/error notifications	0.5 hours	Pranavi
6	Testing Profile Updates	1 hours	Pranavi

Total: ~7 hours

Acceptance Criteria:

1. Given a logged-in user is on their profile page, when they update their personal information, then the changes are saved and displayed.
2. Given a user modifies their profile, when they navigate away and return, then their changes persist.
3. Given a user makes profile changes, when they click cancel, then the changes are discarded.
4. Given a user modifies their profile, when they click save, then they receive a notification that tells them if their changes were successfully saved or not.

5. As a user, I would like to switch between dark and light mode.

#	Description	Estimated Time	Owner
1	Set up theme context in React	1 hour	Pranavi
2	Create theme toggle component	1 hour	Pranavi
3	Implement theme storage in localStorage	1.5 hours	Pranavi
4	Design and implement light/dark CSS variables	3 hours	Pranavi
5	Add theme transitions	1.5 hours	Pranavi
6	Testing theme persistence	2 hours	Pranavi

Total: ~10 hours

Acceptance Criteria:

1. Given a user is on any page, when they click the theme toggle, then the theme transitions and changes accordingly.
2. Given a user sets their preferred theme, when they close and reopen the browser, then their theme preference persists.
3. Given a user changes theme, when they navigate between pages, then the theme remains consistent.
4. Given a user sets theme preference, when they log in on another device, then their theme preference is applied.

6. As a user, I would like to be able to add the classes that I am signed up for.

#	Description	Estimated Time	Owner
1	Create class schema in MongoDB	0.5 hours	Arya
2	Build class search/selection UI	4 hours	Arya
3	Implement class search functionality	2 hours	Arya
4	Create API endpoint for adding classes	1.5 hours	Arya
5	Populate class database	3 hours	Arya
6	Implement success/error notifications	1.5 Hours	Arya
7	Testing class addition flow	2 Hours	Arya

Total: ~15 hours

Acceptance Criteria:

1. Given a user searches for a class, when they select it from search results, then it's added to their class list.
2. Given a user attempts to add a duplicate class, when they submit, then they receive an “already enrolled” message.
3. Given a user adds a class, when they view their dashboard, then the new class appears in their class list.
4. Given a user adds a class, when they log out and back in, then their added classes persist.
5. Given a user reaches the maximum credit limit of 23 credits, when they try to add another class, then they receive an error message.

7. As a user, I would like to be able to edit and delete the classes that I am signed up for.

#	Description	Estimated Time	Owner
1	Create delete class UI	1 hours	Suhesh
2	Implement delete confirmation modal	0.5 hours	Suhesh
3	Create update/delete API endpoints	2 hours	Suhesh
5	Implement success/error notifications	1.5 hours	Suhesh
6	Testing modification and deletion	1 hours	Suhesh

Total: ~6 hours

Acceptance Criteria:

1. Given a user attempts to delete a class, when they confirm deletion, then the class is removed from their list.
2. Given a user tries to delete a class and it goes below the minimum credit hour threshold, then they receive appropriate error messages.
3. Given a user starts deleting a class, when they don't confirm the operation,

then no changes are saved.

4. Given a user deletes a class, when they refresh the page, then the class remains deleted.

8. As a user, I can mark classes that I've already taken and this is updated in my profile.

#	Description	Estimated Time	Owner
1	Add completion status to class schema	1 hour	Pranavi
2	Create UI for marking classes complete	0.5 hours	Pranavi
3	Implement status update API endpoint	2 hour	Pranavi
4	Add visual indicators for completed classes	2 hour	Pranavi
5	Testing completion status	1.5 hour	Pranavi

Total: ~7 hours

Acceptance Criteria:

1. Given a user views their class list, when they mark a class as completed, then

the class shows a completed status.
2. Given a class is marked complete, when the user views their profile, then the class appears in their completed courses.
3. Given a completed class, when a user unmarked it as complete, then the status is updated accordingly.
4. Given a user marks a class complete, when they log out and back in, then the completion status persists.
5. Given a user has completed classes, when they view their course list, then all completed classes are listed.

9. As a user, I would like to be able to see information about class timings and class information.

#	Description	Estimated Time	Owner
1	Create class details schema	1 hour	Zain
2	Design and implement class information UI	4 hours	Zain
3	Implement class details API endpoint	2 hours	Zain
4	Web Scraping and organizing Class Information (Subject Professor, Description, Class size range, pre-requisites)	3.5 hours	Zain
5	Add loading states and error handling	2 hours	Zain
6	Testing information	2 hours	Zain

	display		

Total: ~15 hours

Acceptance Criteria:

1. Given a user selects a specific class when they are on the “Courses to Add” page, then complete class information, including time, professor, and description, is displayed.
2. Given a user selects a specific class when they are on their individual courses page, they can click on a course to view its complete class information, including time, professor, and description.
3. Given a class has schedule information when a user views the class, then accurate timing information is shown.
4. Given a user views class capacity when they check class details, then the current class capacity is displayed.
5. Given a user views prerequisites, when they check class details then all required courses are listed.

10. As a user, I can add, modify, or delete my profile picture.

#	Description	Estimated Time	Owner
1	Create image upload UI	3 hours	Pranavi
2	Implement image preview and cropping	3 hours	Pranavi
3	Set up cloud storage (e.g., AWS S3 or Cloudinary)	1 hour	Arya
4	Create image upload API endpoint	1 hour	Arya
5	Testing image upload flow	2 hours	Arya

Total: ~10 hours

Acceptance Criteria:

- Given a user is on their profile when they upload a valid image file, then the image is displayed as their profile picture.
- Given a user uploads an invalid file type, when they attempt to save, then they receive an error message.
- Given a user uploads a file larger than 5MB, when they attempt to save, then they receive a file size error message.
- Given a user deletes their profile picture, when they confirm deletion, then the default avatar is displayed.
- Given a user crops their profile picture, when they save changes, then the cropped version is displayed.

11. As a user, I can filter classes I search for depending on subject, credit, or class type

#	Description	Estimated Time	Owner
1	Implement filter logic on frontend	3 hours	Suhesh
2	Create filtered search API endpoint	1 hour	Suhesh
3	Add filter state management	2 hours	Suhesh
4	Testing filter combinations	2 hours	Suhesh

Total: ~8 hours

Acceptance Criteria:

- Given a user applies subject filters, when they search for classes, then only classes matching the criteria appear.
- Given a user sets multiple filters, when they search, then results match all selected criteria.
- Given a user clears all filters, when they search, then all available classes are shown.
- Given a user saves filter preferences, when they return to the page, then their preferred filters are pre-selected/
- Given a user applies invalid filter combinations, when they search, then they receive appropriate feedback.

Person	Time
Suhesh	29
Pranavi	30
Zain	30
Arya	30

Remaining Backlog

Functional Requirements:

1. ~~As a user, I would like to be able to register for a BoileResources account.~~
2. ~~As a user, I would like to be able to log in to my account.~~
3. ~~As a user, I would like to manage my account to change the username and other personal details (position, grade, major, etc.).~~
4. ~~As a user, I would like to be able to reset my password if I forget it.~~
5. ~~As a user, I would like to be able to add the classes that I am signed up for.~~
6. ~~As a user, I would like to be able to edit and delete the classes that I am signed up for.~~
7. As a user, I would like to create study groups for classes that I am in.
8. As a user, I would like to join study groups of classes that I am in.
9. ~~As a user, I can mark classes that I've already taken and this is updated in my profile.~~
10. As a user, I would like to be able to communicate to my study group via an in-app chat.
11. As a user, I would like to receive email notifications for chat groups I am in.
12. As a user, I would like to receive email notifications when the page for a class I am in is updated.
13. As a user, I would like to receive recommended study resources that scrape Google for each class that I signed up for.
14. As a user, I would like to be redirected to a Reddit search on the specific class through a button.
15. As a user, I would like to be redirected to a RateMyProfessor search on the specific class through a button.
16. ~~As a user, I would like to be able to see information about class timings and class information.~~
17. As a user, I would like to be able to see information about professors (classes taught, contact information, and typical class size for classes taught), for the classes I signed up for.

18. As a user, I would like to be able to receive grade distributions and class descriptions about the classes I signed up for.
19. As a user, I can post resources for classes taken already/are taking for other students to see.
20. As a user, I can bookmark posted resources for each individual added class that can be easily viewed for future reference.
~~21. As a user, I can add, modify, or delete my profile picture.~~
~~22. As a user, I can filter classes I search for depending on subject, credit, or class type~~
23. As a user, I would like to be able to view my course calendar.
24. As a user, I would like to be able to export my course calendar to Google calendar.
~~25. As a user, I would like to switch between dark and light mode.~~
26. As a user, I can comment and view comments on posted resources..
27. As a user, I can edit my posted resources and comments.
28. As a user, I can share posted resources to others via a share button that copies a link to clipboard.
29. As a user, I can view and respond to a feedback form for comments/questions about the site.
30. As a user, I can report inappropriate content/behavior on the platform.
31. As a user, I would like to filter recommended resources by type (e.g., PDFs, lecture slides, YouTube videos, textbooks).
32. As a user, I would like to get emails about new resources regarding the classes I signed up for.
33. As a user, I would like to react to posted resources by upvoting/downvoting them for visibility.
34. As a user, I would like to react to comments on posted resources by upvoting/downvoting them for visibility.
35. If time allows, as a user, I would like to receive youtube video recommendations for the course I am taking.
36. If time allows, as a user, I can customize the layout of my dashboard to prioritize the classes and resources most relevant to me.
37. If time allows, as a user, I can view my exam timings for a certain class on my calendar.
38. If time allows, as a user, I can view the past exam averages for the exams of a certain class.
39. If time allows, as a user, I can input my courses into a scheduling assistant that will tell me what my week would look like based on the courses that I added.
40. If time allows, as a user, I can add tasks to a personal planner.

Non-Functional Requirements:

Architecture and Performance Requirements

- The application will have a completely separate frontend and backend to make collaboration and compatibility easier.
- Node.js and Express will be used to build the backend, providing a RESTful API.
- MongoDB will handle data storage.
- The frontend will be developed using React, fetching data via API requests
- Cloud storage and automated management features will also be provided by the application.

Security Requirements

- The application must protect sensitive user information and scheduling data from unauthorized access.
- The application must validate all user input to prevent injection attacks.
- The application must implement rate limiting to mitigate brute-force attacks, limiting users to a maximum of 100 requests per 5 minutes.
- Users must not be able to view other users' account details or scheduled classes unless explicitly permitted.
- The application must enforce secure login and registration processes, including email verification and strong password policies.
- The application must provide a user reporting and flagging system to identify and address potential security concerns.

Usability Requirements

- The application must feature a responsive design compatible with all modern devices and browsers.
- The interface must be intuitive and easy to navigate for all users.
- Each user must have a centralized profile dashboard displaying their activity on the platform.
- All students must have access to the same layout, but past students and TAs must have additional privileges to endorse posted resources.

Hosting/Deployment Requirements

- The backend and frontend must be deployed separately to allow independent modifications and scalability.

- The frontend must be deployed on Vercel.
- Deployment configurations must be managed through Vercel's dashboard to ensure streamlined updates and maintenance.