BoilerScout Sprint 1 Planning Document

Team 15

Baris Dingil, Hardy Montoya, Jacob Miecznikowski, Selin Ovali, Terry Lam

Sprint Overview

During this sprint, our team's goal is to implement the signup, login, and authentication portion of our application. The frontend team will be creating the views and hooking up routes while our backend will be in charge of writing the functionality that interfaces with our database and handles email verification. By the end of this sprint, users should be able to sign up and create an account with our application, and verify their accounts via their Purdue email addresses.

Scrum Master

Terry Lam

Meeting Schedule

Standup meetings weekly on Tuesdays and Wednesdays (with Project Coordinator), longer group meetings on Sundays.

Risks/Challenges

This is the first time many of members are working together on a real project, meaning that there is a lot of inexperience when it comes to different frameworks and technologies. For this first sprint, some time will need to be spent getting up to speed so that those members can contribute effectively. Another challenge we face is ensuring that all production level code is the same quality, and that everybody is doing their due diligence when reviewing code and pull requests. Communication will be essential between the backend and frontend to make sure both parts can integrate together during this first sprint.

Sprint Detail

User Story #1

As a user I would like to sign up with my Purdue email and a unique password, so that I can have a set of credentials to use the application.

#	Task Description	Estimated Time	Owner
1	Set up database tables	3	Terry
2	Write API logic to add users to database	5	Terry
3	Implement hashing functionality to apply on user passwords	3	Terry
4	Setup signup endpoint	3	Terry
5	Creating email verification method	3	Hardy
6	Redirect verification email to login page and update database with user’s email status upon completion of verification	3	Baris
7	Write logic to remove user credentials from database if their email hasn’t been verified after a certain amount of time	5	Hardy
8	Implement the ability for people who receive a confirmation email for an account they didn’t sign up for to cancel their credentials (`Didn’t sign up`).	5	Baris
9	Create home view	5	Selin
10	Create sign up view	5	Selin

Acceptance Criteria

- ➔ Given that sign-up logic is implemented successfully, when a user signs up, then their credentials will be added to our database.
- ➔ Given that sign-up logic is implemented successfully, when a user signs up with invalid input (malformed or non-Purdue email), they will be met with an error.
- ➔ Given that sign-up logic is implemented successfully, when the user correctly signs up, before being allowed to further use the web app, they will be prompted to confirm their email via the verification method implemented.
- ➔ Given that both sign-up logic and the verification method are implemented successfully, once the user confirms his email, they should be allowed to freely use the app without any more barriers.
- ➔ Given that a user has signed up, when their email hasn't been verified after a certain time, then erase user data and account to prevent false account creations or malicious attempts.

Testing
To test this module, we will be calling our API via Postman with different inputs in our requests. Proper and malformed input will test to see if our sign up flow is robust enough, and ensures that our error handling is correct. Results can simply be examined through entries in our database as well as the response object. We will also be testing email verification by sending off requests to our own personal emails to confirm that it is working as intended.

User Story #2
As a user, I would like to login my Purdue email and password that I created, so that I can access and use the application.

#	Task Description	Estimated Time	Owner
1	Write query logic to validate a user's credentials upon login	5	Terry
2	Write logic to handle access and expiration tokens	10	Terry
3	Setup login endpoint	3	Terry
4	Create login view	5	Jacob

Acceptance Criteria

➔ Given that login logic is successfully implemented, when a user attempts to login with correct and valid credentials, then they will be authenticated successfully.
➔ Given that a user's email has not been verified yet, when a user attempts to login, then they will be presented with an error and asked to verify their email.
➔ Given that access token validation has been successfully implemented, when a user logs in, they will be assigned an access token as well as an expiration token.
➔ Given that a user's access token has become stale, when a user navigates onto our application, they will be redirected to our login page.
➔ Given that a user has signed up, when a user attempts to login with invalid credentials, they will be presented with an error.

Testing
This module will also be tested via Postman. We will be providing invalid and valid credentials to our login API and examine results through our database and response objects.

User Story #3
As a user, I would like to be able to reset my password via email if I forget it.

#	Task Description	Estimated Time	Owner
1	Send a randomly generated temporary new password via email	5	Baris
2	Implement functionality to add newly generated passwords to the appropriate database table	3	Baris
3	Setup reset password endpoint	3	Baris
4	Create reset password view	5	Jacob

Acceptance Criteria
➔ Given that the reset password endpoint is successfully implemented, users should have the option to reset their password on the login page that directs them to the reset password view.
➔ Given that the reset password view is finished, users should be presented with the password view prompting them to enter their email.
➔ Given that the reset password feature is successfully implemented, only emails registered to an existing account should be able to successfully use this feature.
➔ Given that the reset password feature is successfully implemented, user should receive an email with a randomly generated temporary new password.
➔ Given that the database table should be updated with the new password, user should be able to sign into their accounts with the new password they received via email.

Testing
To test this module, we are going to create an dummy account, which has its own password. After we created this account we are going to go to our website and go to login page. In login page we will click on “forget my password”. This will send a randomly generated temporary new password to our email. We will use that new password as our password; if we can login with it, it means that our “forget my password” works. After we login with that password, we will change the password than open the login page again. If we can login with the password we created, it means that our system works.

User Story #4
As a user, I would like to be able to resend the confirmation email.

#	Task Description	Estimated Time	Owner
1	Setup resend confirmation endpoint	3	Hardy
2	Request another verification email from the API	5	Hardy
3	If email is not confirmed after a set period of time, erase users data	5	Hardy
4	Create resend confirmation view	5	Selin

Acceptance Criteria

- ➔ Given that the resend confirmation endpoint is successfully implemented, users should have the option to request a new confirmation email from the corresponding page.
- ➔ Given that a new confirmation email is requested, the first one should be made invalid to prevent any problem that could arise from having two functioning confirmation channels.
- ➔ Given that the request a new confirmation email view is finished, users should be presented with a page where they can enter their email.
- ➔ Given that a resend of the confirmation email is requested, guarantee that the user will receive it.
- ➔ Given that malicious use of the web app could lead to an account being created using someone else's email, when a user receives a confirmation email for an account they did not create themselves, then an option to mark the email and the account as not requested would erase the data related to saidl email address.

Testing
To test this module, we will be using a dummy account that will ignore the first confirmation email, and instead request a new one. The first email sent should then be rendered useless, and we will attempt to confirm via it to ensure this is working properly. We will then attempt to confirm via the second email sent, which should work just as smoothly as trying to confirm with a single initial mail.

User Story #5
As a user, I would like to update my password via my account settings on web.

#	Task Description	Estimated Time	Owner
1	Write database query logic to update password	3	Baris
2	Setup update password endpoint	3	Hardy
3	Create update password view	5	Jacob

Acceptance Criteria

- ➔ Given that the logic for updating password is successfully implemented, when a user chooses to update their password, they should enter their old password to for security and only successfully changed their passwords if the current password entered was correct.
- ➔ Given that the logic for updating password is successfully implemented, when a user successfully updates their password, then the database should update to reflect their new password.
- ➔ Given that the update password view is finished, users should be presented with a page where they can enter a new password.
- ➔ Given that a user has updated their password, when they login again, then they should be able to authenticate successfully.
- ➔ Given that a user updates his password, when they try to change it to their current one, then they will be prompted with an error.

Testing
Testing of this module will simply include passing a user's id and previous password, as well as input (valid and invalid) for their new password to the API endpoint. By examining the response objects and database entry for that user, we will be able to see if our logic is successfully implemented.

User Story #6
As a user, I want to be directed to the scout page when I log in the web application.

#	Task Description	Estimated Time	Owner
1	Create top navigation bar	4	Selin
2	Setup scout endpoint	3	Baris
3	Create scout view	5	Selin
4	Create advanced filters view	5	Jacob

Acceptance Criteria

- ➔ Given that the main landing endpoint is successfully implemented, when a user logs in, they should land on the home scout page with a top navigation bar.
- ➔ Given that the top navigation bar is successfully implemented, users should be presented with links to different pages of the application.
- ➔ Given that the scout view is finished, users should be presented with a page where they can search for other users.
- ➔ Given that the advanced filters feature is successfully implemented, users should be able to access a page with specific filters.

Testing
To test this module, we will simply log in and make sure that the user lands initially on the "scout" page. We will try to make a search with random query. As we won't have the results page done in Sprint 1, we only expect the search to go through but no results to be presented. We will also check if advanced filters view has the correct fields and options for user as the search criterias change.

Remaining Backlog

1) As a user, I would like to create a profile that contains a short biography, classes taken and my personal skills, so that other user's can see if I meet their needs.
2) As a user, I would like to be able update my profile with new information and skills, so that my relevant classes and skills are as recent as possible.
3) As a user, I would like to have a basic search function for names, specific skills, or courses and have a list of users that match be returned.
4) As a user, I would like to have advanced filters in addition to regular search such as year standing of students or their majors, so I can make my search as narrow or as wide as I want.
5) As a user, I would like to be able to send another user an email through a link on their profile.
6) As a user, I would like to view a forum containing different posts from other users, so that I can see other questions that Purdue students have posted.
7) As a user, I would like a a multiple variety of forums that contain relevant threads, so that there are multiple topics of discussion (such as Mentor Search, Class Help, etc).
8) As a user, I would like to search through the forum with keywords and tags, so I can look for posts that are relevant to me.
9) As a user, I would like to make posts on the forum, so that I can ask questions or post issues I have that other Purdue students may be able to help me with.
10) As a user, I would like the ability to sort threads by most recent or oldest, in order to refine my searching.
11) As a user, I would like to be able to save certain forums for quick access. (if time permits)
12) As a user, I would like to comment on other people's posts, so that I can provide my expertise in an area if it calls for it.
13) As a user, I would like to view the post history of another user. (if time permits)
14) As a user, I would like to privately message other users. (if time permits).
15) As a user, I would like to have an inbox where I can receive messages. (if time permits)
16) As a user, I would like to delete read or unneeded messages from my inbox. (if time permits)
17) As a user, I would like the application to search and show me the results in a relatively short time, so that it isn't tedious to run many searches one after the other.