Dithi Saxena
Catherine Penquite
Abhishek Gunasekar
Christopher Yu
Vivek Nair
Team 6

Sprint 1 Retrospective

15th March 2021

Werk It

TABLE OF CONTENTS

What went well?

In general, we developed the fundamental functionality of both the web app and mobile app, enabling user sign up and login authentication with the two respective clients for our project. We also implemented a common backend route on the server side that communicates with the mobile client and web client simultaneously, thereby enforcing the cross-platform nature of our project.

User Story #1

As a user, I would like to be able to easily access the *Werk It* landing page.

Completed:

The work landing page has a simple and elegant user interface that illustrates the application's purpose, features and provides access to all the important pages such as the sign up and login page. Additionally, the user can choose to return to the landing page at any time using the shortcut icon.

User Story #2

As a user, I would like to be able to register for a *Werk It* account on the web app.

Completed:

After the user enters all the required fields within the sign up page, there is a successful redirection to the welcome screen. More importantly, each user entry is stored accurately within the MongoDB cluster and the backend route between the server and the database is strongly established.

User Story #3

As a user, I would like to be able to login to my *Werk It* account on the web app.

Completed:

After the user enters all the required fields within the login page, there is a successful redirection to the dashboard page. More importantly, the backend route correctly identifies the respective user by accessing the requested entries within the database.

User Story #4

As a developer, I need to display an error message if at least one of the user's credentials already exist in the database when creating an account.

Completed:

The web page and mobile app both successfully display an error message when a user tries to login with a pre existing username. Not only this, but the web app and mobile app will also display an error message if a user attempts to sign up with a pre existing account.

User Story #5

As a developer, I need to display an error message if at least one of the user’s credentials is incorrect.

Completed:

On login, the username and password given by the user input is checked with the data stored in the database. If the username is nonexistent, if password is nonexistent, or if the username and password do not match an existing profile in the database, then an error alert is shown to the client and the user is not logged in. This functionality works on both the web and mobile client.

User Story #6

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

Completed:

The reset password page is accessible from both the web and mobile client, and when the required fields are filled out correctly (username, new password, confirm new password) then the new password is saved to the existing user's profile in the database, the old one is removed, and the client receives a success message. When filling out the required fields, if the username does not exist or the new password fields do not match, then an error alert is shown and the password is not updated.

User Story #11

As a user, I would like to be able to sign up on the mobile app.

Completed:

The create account form is accessible from the login screen on the mobile application. In addition, the mobile application accurately communicates user data to the MongoDB database in order to create a new account.

User Story #12

As a user, I would like to be able to login to my Werk It account on the mobile app.

Completed:

The login form is functional and is displayed upon opening the mobile application. Additionally, the mobile application accurately communicates with the MongoDB database to ensure that user credentials are valid before logging in.

User Story #13

As a user, I would like to be able to easily navigate the dashboard of the mobile app.

Completed:

The dashboard screen is accessible from the login screen and displays a list of previously created workouts along with a create workout button. Any of these workouts or the button can be tapped to pull up the workout editor screen.

User Story #14

As a user, I would like to be able to easily access the workout editor screen on the mobile app.

Completed:

The workout editor screen is accessible from the dashboard screen when the user clicks the new workout button. Furthermore, the workout editor screen prompts the user to add exercises to the workout, showing relevant input fields.

What did not go well?

In general, we were unable to implement features such as displaying a loading symbol, setting a profile picture, making the mobile device remember a user login, and having touch ID and face ID login. While this did not create an obstacle for basic functionality of the app, it is important we incorporate these in the next sprint as they are essential features most exercise applications include.

User Story #8

As a developer, I need to display a loading symbol if requests take longer than a second so that the user does not think the app has frozen.

1	Design loading symbol UI for web app if requests to the server take longer than one second.	3 hrs	Vivek

2	Design loading symbol UI for mobile app if requests to the server take longer than one second.	3 hrs	Vivek

Not completed:

The loading symbol was not displayed on the web or mobile app if the elapsed time since the button click was more than 1 second. It was also not displayed while the client was waiting for a response or if there is an error with the request.

User Story #9

As a user, I would like to be able to set my profile picture.

1	Add profile pic feature on the web app.	6 hrs	Vivek

2	Add profile pic feature on the mobile app.	6 hrs	Vivek

Not completed:

The user could not set or update their profile picture on the web or mobile app if the user clicks the profile image. The user was also not able to choose to take a picture or upload an existing picture for both the web and mobile app, and the profile picture was not displayed in the top left corner of the profile and dashboard pages.

User Story #10

As a user, I would like my device to remember me until the next time I log out on the mobile app.

1	Create a token corresponding to the login credentials that persists even when the app is closed.	2 hrs	Vivek

2	Access the token when the user boots up the mobile app to automatically log in.	2 hrs	Vivek

Not completed:

The mobile app did not remember the user credentials. The mobile app did not implement the “Keep me signed in" option, nor did create a token in a supported package (ex. MMKV) that stores the user’s login credentials. When a user signed out/in, the user did not have a login token created for their mobile app.

User Story #15

As a user, I would like to have face ID login once my credentials are saved.

1	Check whether device is capable of supporting face ID login	3 hrs	Vivek

2	Create key in Info.plist file	4 hrs	Vivek

3	Implement authentication handling in the backend to validate face ID login	2 hrs	Dithi

Not completed:

The mobile application did not listen for a face ID input. The mobile application was not able to check whether a device is capable of supporting face ID nor was it able to detect a correct face ID when they login.

User Story #16

As a user, I would like to have touch ID login once my credentials are saved.

1	Check whether device is capable of supporting touch ID login	3 hrs	Vivek

2	Create string written in code for touch ID	4 hrs	Vivek

3	Implement authentication handling in the backend to validate touch ID login	2 hrs	Dithi

Not completed:

The mobile application did not listen for a touch ID input. The mobile application was not able to check whether a device is capable of supporting touch ID nor was it able to detect a correct touch ID when they login.

How should you improve?

There are many lessons our team learned from this first sprint. First, we could improve by going through our team's practice sprint review the weekend before the sprint review (so at least 5 days ahead). For sprint 1, we practiced our sprint 1 review presentation the day before the actual sprint review, which exposed some changes that needed to be made. However, it was stressful to make those changes with the minimal amount of time before the actual sprint review. In addition, some changes made now need to be refactored in sprint 2, as the quickest method to fix them was not the best in terms of code architecture. Having our practice sprint review multiple days in advance will allow us to make the necessary changes correctly before the actual sprint review with our grader.

Furthermore, over the course of this first sprint, we realized the importance of communicating with everyone on the team. This includes responding with available times for meetings, as well as just general updates on how everyone is doing with their work and whether they need help or not. During Sprint 1, while communication was good overall, sometimes things were misunderstood or poorly communicated, and that led to inefficient meetings and a general waste of what could have been productive time. In order to improve for this next sprint, we will try to be more vocal about any issues we might be having, as well as notifying the team if situations arise that conflict with meetings.

Moreover, we learned that at the very least, pull from the main branch at least every week. One team member switched to their own branch at the beginning of Sprint 1, and pulled from another teammate's branch. They did all of their work on this version of the repository, and kept making pulls without realizing that they were pulling from their own branch. This resulted in this team member having challenges when other teammates were testing their work, as the team's changes were not compatible with this team member's latest changes (integrated with the main codebase). In order to improve this for the next sprint, by default, a pull will be made from the "main branch" at both the beginning and end of a week, and every time a group member integrates their branch with main (at the very least).

Another issue we faced during this sprint is abiding by the acceptance criteria we made. At the beginning of the sprint, when we made our sprint planning document, we naively put very specific acceptance criteria that made it difficult to implement and integrate multiple features. There was also a problem with our acceptance criteria being very poorly worded or repeated, and during our sprint review we had to clarify a lot of this to our grader. To solve this problem, we are going to more carefully word our acceptance criteria and make sure it is implementable by each person.

Finally, we can improve on planning and scheduling our tasks much ahead of time rather than postponing it. Moreover, by coordinating with other team members during week two of the sprint would be much more effective instead of waiting until the last week to do so. For example, the members responsible for the UI and front-end parts of the application should work incrementally with members responsible for the backend components of the application, thereby preventing any unprecedented connectivity issues between the front-end and back-end.