Ryan Spadt

CS 499 Milestone Three Narrative

11/23/25

“The artifact selected for this enhancement was a mobile application created in CS 360 last term. We were given an option to create one of three application concepts. I elected to create a weight tracking mobile application. This application was created using Java to run an Android mobile application. That would allow a user to sign in or register an account. Once they are logged in they would be able to set a weight goal and input weight entries on a daily basis to track their progress towards that goal. The application also supported the ability to send SMS messages to the user when they accomplished their goal based on the setting inside of the application and phone permissions.

I selected this artifact because I found making mobile applications challenging, exciting, and rewarding. It’s something I’d actually like to do in my free time after I have graduated from SNHU. I also have been tracking my weight in a similar fashion inside of a Google Sheets spreadsheet and thought this would be a perfect opportunity to create something I could use. If I wanted to use this application I would need to write it for iOS and found Kotlin Multiplatform Mobile. So I wanted to learn Kotlin Multiplatform Mobile anyway. And with the enhancement requirements I thought it aligned perfectly to migrate it from Java (Android-only) to KMM and have support for Android and iOS.” (Spadt, Narrative Milestone Two).

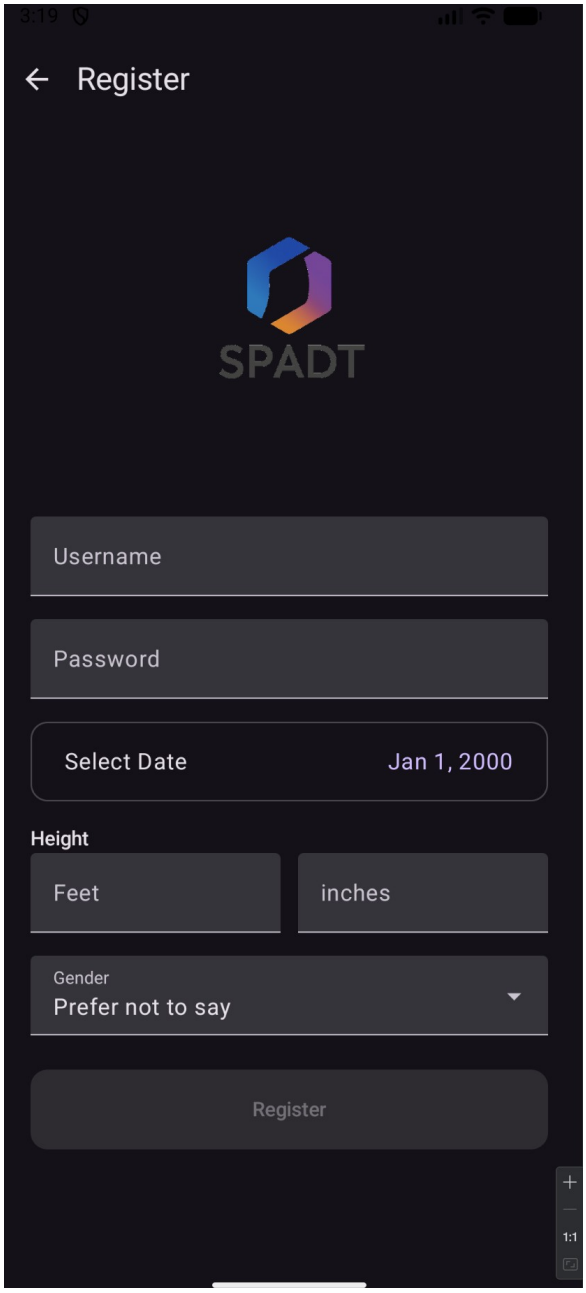
In all of my milestones, I have selected to complete enhancements on the same artifact. In this case, I improved upon the work that was completed in milestone 2. And the artifact has been

improved significantly in this enhancement. The following improvements were included in this enhancement:

1. “A new screen “Insights” that displays insights about the users entries such as confidence score, the trend of weight change per day, the estimated number of days until their goal is met, and how many outliers were detected in their entries. This screen was also added to the bottom navigation menu. I calculated these insights using a variety of formulas like linear regression, outlier detection using MAD-based Z-score, exponential smoothing for forecasting, and confidence scoring for goal estimation.

2. I added to the User object in the SQLDelight database date of birth, height, and gender.

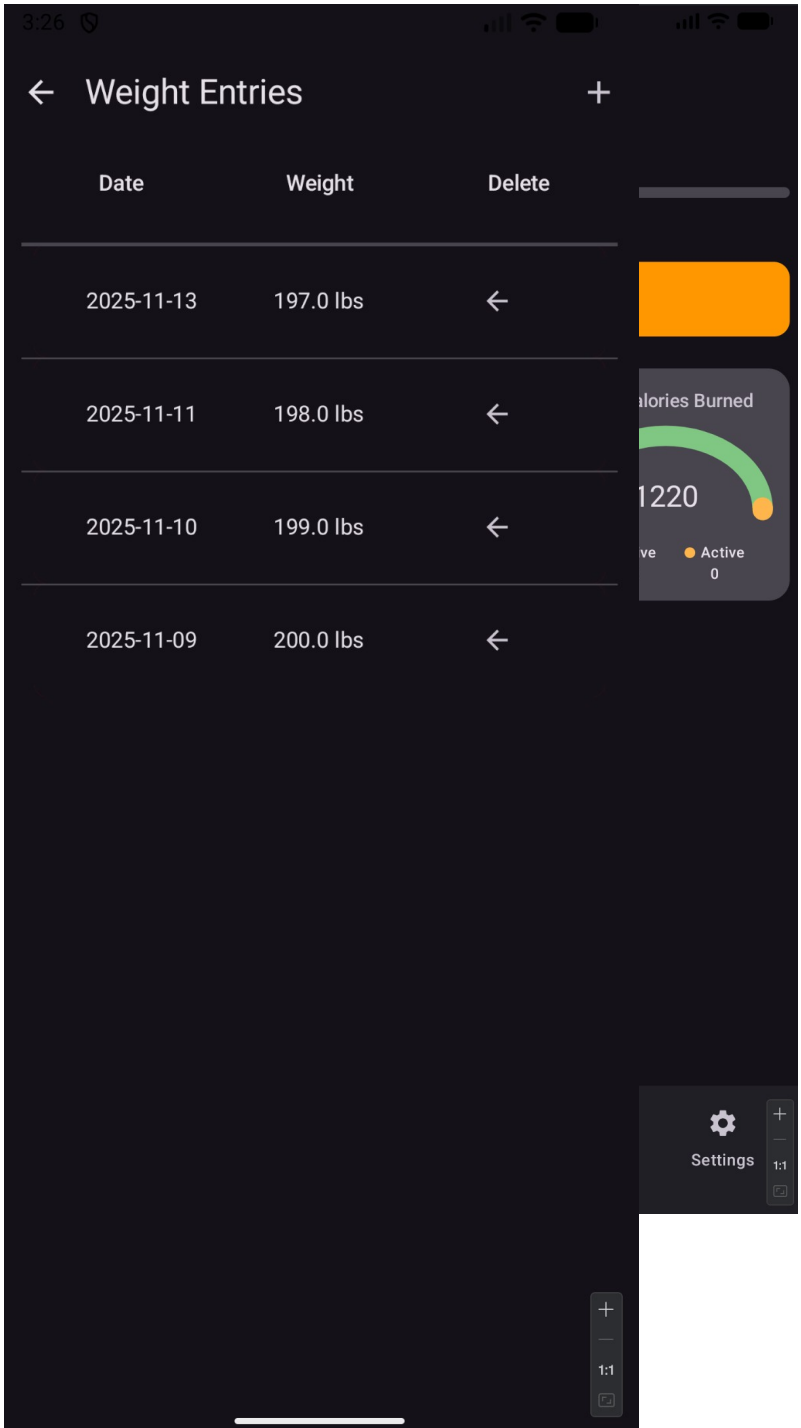
3. I modified the User registration screen to now capture those new fields at registration.



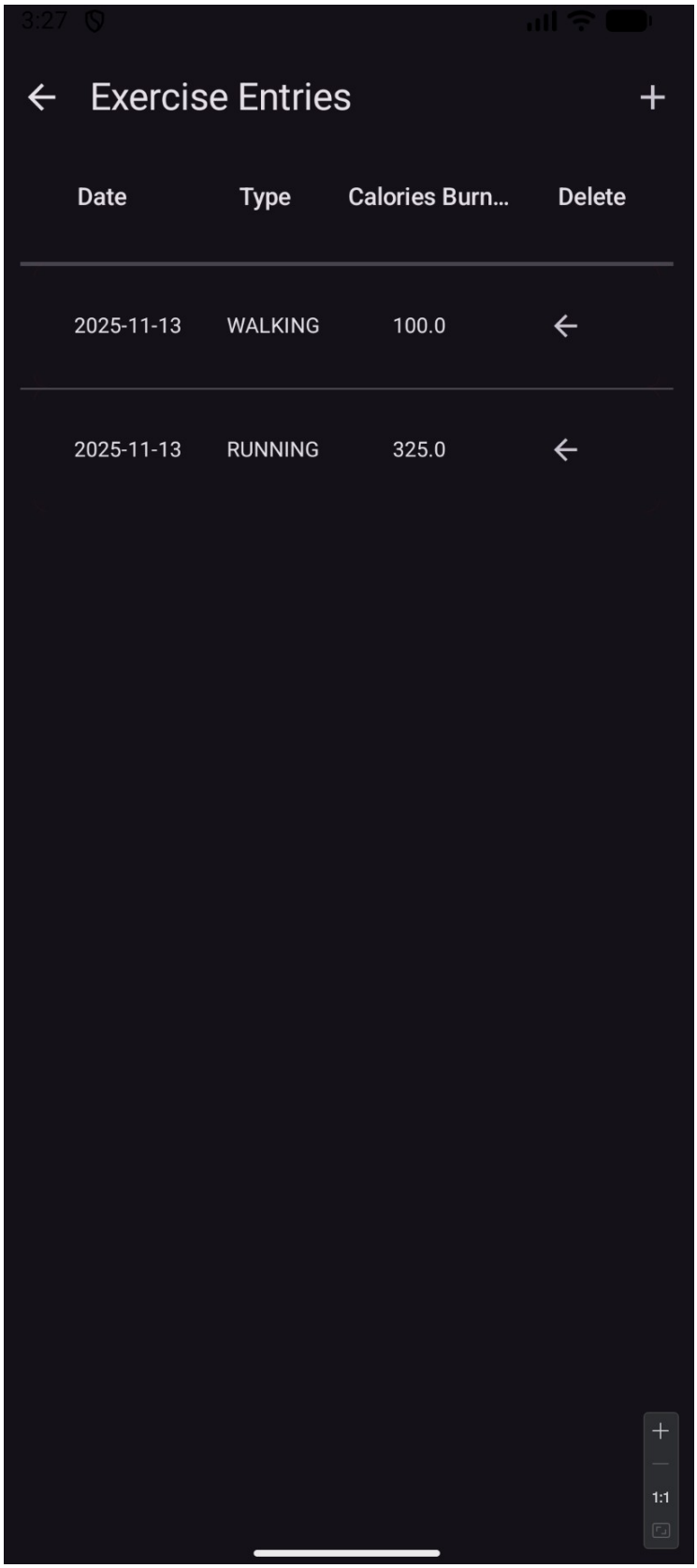
4. I also modified the validation for this form to ensure input is received. Height is stored as a calculation of the feet and inches of the input received and stored as inches.
5. Adapters were made for those fields so the database can link custom data types.
6. I fixed an API version issue that exists in DateUtils for the Clock library. I'm not sure what was wrong, it was working fine but the IDE was not satisfied.
7. I created BmiUtils for calculating the BMI of the user based on their height and weight.
8. I created a composable for a loading state to show a spinner and some text.

9. Instead of having the weight entries on the main home screen, I created re-usable configurable tiles for displaying information. These tiles can be used for different information to be displayed and when they are clicked they take you to another context screen like registration (where there is a back arrow). I created both a stat tile and a half moon chart tile. They are dynamically ½ width and 1/3 height of available screen space.

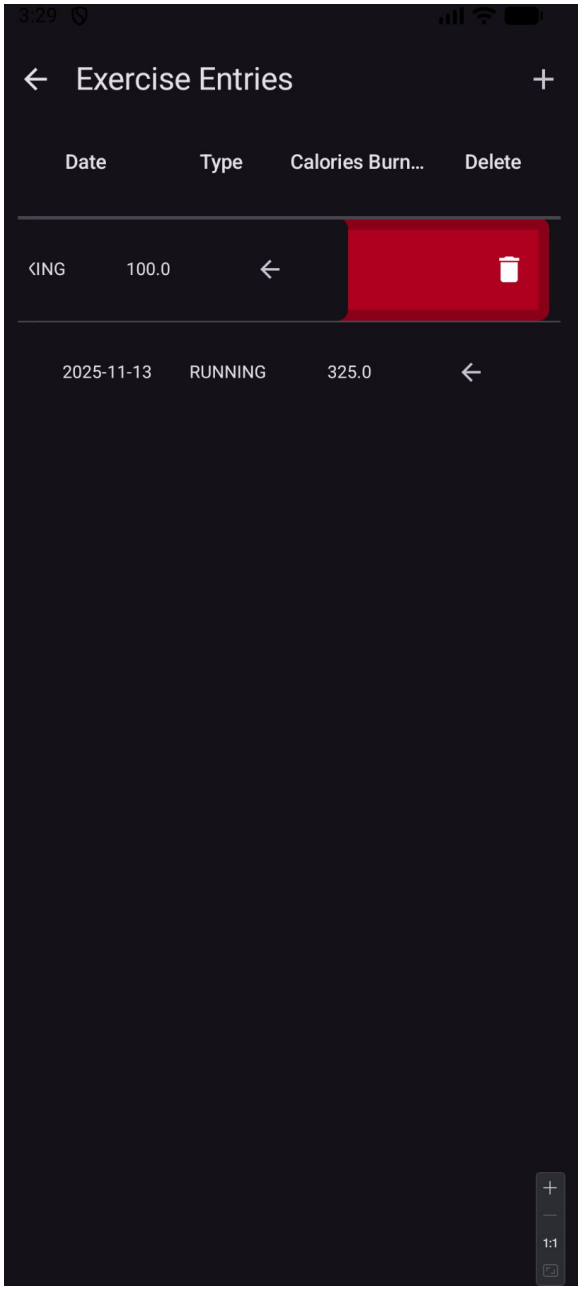
10. I created new screens for when you click on the tiles.
11. Which also includes a new table in the database for exercise objects. A new screen for them and a dialog for entering new exercise objects.



12. I made the WeightTable and WeightRow more generic so that I can re-use them for multiple object types instead of just weights.



13. I made the rows swipe to the left to delete instead of clicking a button which I think is more mobile application friendly.



14. I created new view models for both user objects and exercise objects to facilitate operations between the DB and UI layers.

15. I made goal progress round to two decimal places in the goal section, if the user input a decimal number it would have a lot of decimal places displayed. This makes it more concise.
16. The new tiles created an issue with DataScreen, where I needed to pass a lot of managers, daos, and view models into the data screen from the MainActivity and MainViewController. I did not like this because in the future I'd like users to be able to customize this page with tiles they want to have in there. If I wanted that functionality I would have had to pass the entire configuration state of the application to the data screen. To avoid this, I made tile factories and gave them ownership of the appropriate view models, daos, and managers. Then I passed a list of tile factories to the data screen.
17. I extracted the goal section into its own composable outside of the data screen.
18. I created constants for colors that I was using throughout the application to provide consistency in colors/themes.
19. For the new calories burned tile it shows passive calories burned and active calories burned. The passive calories burned are calculated as BMR using the Mifflin-St Jeor BMR formula. I added a refresh capability to the tile and made it have a fade-in-out glow effect on the graph when the value changes. So that the user can see up-to-date information in the tile." (Spadt Journal 3-1).

The expected course outcomes that were aligned with this enhancement were outcome 3 and 4. I believe I accomplished both of these outcomes in this enhancement. By applying best practices and principles in my improvements. Such as, making code into component-like or modular-based fashion so that I could re-use the code with configuration rather than writing duplicative code. This was demonstrated in my re-factoring/creation of my table, table row, StatTile, StatLine, and Half Moon Chart tile composables. I also was able to verify that the algorithms used to create the insights screen were aligned with best practices. They were efficient and completed calculations successfully quickly on both Android and iOS platforms. Additionally, outcome 4 was met because I created a solution that would realistically be used in a production ready application. The UI has significantly improved, the features are cool, interesting, and insightful, and I have diversified the application from passive weight-goal tracking to something that is able to provide dynamic feedback/insights.

Again, I learned a great deal completing this artifact. I faced several challenges with completing the different features of it. The initial enhancement of implementing an algorithm using linear regression, exponential smoothing, outlier detection, and confidence scoring was something I needed to re-learn. It also required a significant amount of testing and tuning with different inputs to verify that it was working properly. The other enhancements required significant re-work of my existing code to accommodate future enhancements. I wanted to make it so that my code was modular and could be configured in various ways to display different information. I would like, eventually, for the end user to be able to customize their home screen with those different tiles. Making them different screen widths ½ or full screen width and being able to add different tiles or remove them. To create something like that I knew that I needed to make them configurable.

I also knew that I wanted to display more data and make the app have more functionality, to be full-scoped on helping users meet their weight goals. Adding passive calorie calculation and being able to add active calories burned. Calorie in and out tracking will help the users meet their goals. Making different tiles to display that information became a great challenge and addition. With that addition, it created more challenges with how I designed the home screen though. I was passing in essentially all of the needed view models, DAOs, and other helpers into the data screen from the main app. It worked fine, but if I wanted to eventually have a lot of tiles, I would need to pass in the entire state of the application to this screen. This seemed like poor design, especially if a user could remove a tile and one of those resources were no longer needed. I needed to transfer ownership of those resources to the tiles themselves and pass in a list of tile factories to solve this problem.

Finally, I faced a challenge with implementing a screen with essentially the same desired functionality as weight entries for exercise entries. The table and table row composables I had created were specialized for weight objects. So I needed to redesign these composables to be able to be used for any object. It was a challenge to try and figure out how to do this but I managed to pull it off successfully. It’s not to where I would like it yet, and I would like to make the AlertDialogs more configurable as opposed to having them specialized as they are now. But it is definitely an improvement.

Spadt, Ryan. “CS 499 3-1 Journal Spadt.” *CS 499 3-1*, 14 Nov. 2025. Accessed 22 Nov. 2025.

Spadt, Ryan. “CS 499 Narrative for Milestone Two.” *CS 499 Narrative for Milestone Two*, 15 Nov. 2025. Accessed 22 Nov. 2025.