Module 7-1: Final Project: Enhancement Three: Databases

Stephanie S. McClurkin

Southern New Hampshire University

CS-499: Computer Science Capstone

Professor Federico Bermudez

February 19th, 2025

Briefly describe the artifact. What is it? When was it created?

The artifact I chose to enhance is my weight-tracking mobile application, originally developed in Android Studio during my CS-360 Mobile Architect & Programming course. The app allows users to log their weight, track progress, and set personal weight goals. The original version stored data using SQLite, requiring users to manually enter weight entries that were displayed in a RecyclerView list. For this enhancement, I focused on improving the app's database functionality by integrating Firebase authentication and Firestore database support. This upgrade allows users to securely log in and store their weight data in the cloud, making it accessible across multiple devices while ensuring data security and reliability.

Justify the inclusion of the artifact in your ePortfolio. Why did you select this item? What specific components of the artifact showcase your skills and abilities in software development? How was the artifact improved?

I chose to enhance my weight-tracking application for this project because I genuinely enjoyed working on it during my CS-360 Mobile Architect & Programming course and always wanted to explore ways to improve it. The original app provided a straightforward and functional way to log and track weight entries, but it relied on a simple local SQLite database with shared storage, which came with some clear limitations. If a user's data was lost or corrupted, there was no way to recover it, and using the app across multiple devices was impractical since weight entries were tied to a single device. Given these challenges, integrating cloud storage, authentication, and an import/export feature felt like a reasonable next step to improve both functionality and user experience.

To enhance the app's database capabilities, I integrated Firebase Firestore and authentication, allowing users to securely log in, store their weight data in the cloud, and access

it from any device. Now, data is no longer at risk of permanent loss, and users can transition between devices without losing progress. The import/export functionality also gives users greater control over their data, letting them manually backup and restore it as needed - something that wasn't possible in the original version. This enhancement demonstrates my ability to work with cloud-based databases, authentication systems, and real-time data synchronization, all while keeping the app user-friendly and reliable. Through this process, I've strengthened my skills in database integration, debugging, and UI refinement, making this artifact a strong addition to my ePortfolio.

Did you meet the course outcomes you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?

Course Outcome 2: Design, develop, and deliver professional-quality oral, written, and visual communications that are coherent, technically sound, and appropriately adapted to specific audiences and contexts.

For this enhancement, I wanted to make sure that my implementation was not just functional but also well-documented and easy to follow. Since Firebase is more complex than local storage, I knew I needed to document things carefully so future developers (or even my future self) could easily understand how everything works. To do this, I added numerous comments throughout my code, explaining how user authentication is handled, how weight data is stored, and how data syncs between Firebase and the local SQLite database. This further satisfies the second course outcome. Beyond making the code easier to understand for the developer, I also focused on making things more intuitive for the user. One way I did this was by improving error messages and feedback notifications. Before, if something went wrong, users might not know why. Now, the user will receive an error message letting them know what

happened and how to proceed. By improving both technical documentation and user communication, I made this enhancement easier to maintain and more user-friendly. The code comments make Firebase integration much simpler to work with, and the error messages help users navigate the app with ease. These efforts directly support the second course outcome by demonstrating my ability to write clear documentation, communicate technical processes effectively, and improve the overall user experience.

Course Outcome 4: Demonstrate an ability to use well-founded and innovative techniques, skills, and tools in computing practices for implementing solutions that deliver value and accomplish industry-specific goals.

This enhancement took my weight-tracking app from a basic local-storage application to a cloud-integrated solution, making it more reliable and scalable. Instead of users being stuck with data saved only on their device, they can now log in securely with Firebase authentication and access their weight history from any device. Moving to cloud storage shows that I can work with industry-standard tools to improve both functionality and user experience, satisfying the fourth course outcome. By successfully integrating Firestore and Firebase authentication, I demonstrated my ability to work with different database architectures and build scalable, cloud-based solutions.

Another major improvement was adding import/export functionality, giving users the ability to back up and restore their weight data. Before this, there was no way to recover lost data or transfer weight entries between devices. Now, users have full control over their progress, which is an expected feature in most professional applications. This update shows that I can design features that align with user needs while following best practices, further highlighting my ability to deliver modern solutions that meet industry-specific goals. This enhancement directly supports

the fourth course outcome by showing that I can use innovative tools and computing techniques to create practical, scalable solutions that add real value to users.

Course Outcome 5: Develop a security mindset that anticipates adversarial exploits in software architecture and designs to expose potential vulnerabilities, mitigate design flaws, and ensure privacy and enhanced security of data and resources.

Security was a major focus of this enhancement. Before this update, all user data was stored locally on the device, meaning anyone with access to the phone could view or modify weight entries. The login system wasn't very secure, there was no way to protect or recover user data. To fix this, I implemented Firebase authentication so that users must log in to access their weight history. This greatly increased the security of the application and its data, further promoting the fifth course outcome. Beyond improving the login process, Firestore security rules were utilized to protect user data from unauthorized access. These rules made sure that only users can read and write their own data, preventing others from accessing or tampering with their weight history. You can see the process for setting these rules in the screenshot below.

```
rules_version = '2';

service cloud.firestore {
  match /databases/{database}/documents {

    // Allow users to access only their own main document
    match /users/{userId} {
      allow read, write: if request.auth != null && request.auth.uid == userId;
    }

    // Allow users to read and write to their own weight data stored in a subcollection
    match /users/{userId}/weights/{weightId} {
      allow read, write: if request.auth != null && request.auth.uid == userId;
    }
  }
}
```

By adding cloud authentication and security rules, I significantly improved data privacy and protection in my app. This enhancement directly supports the fifth course outcome by

demonstrating my ability to identify potential security risks, implement authentication safeguards, and design a system that prioritizes user privacy and data security.

Reflect on the process of enhancing and modifying the artifact. What did you learn as you were creating it and improving it? What challenges did you face?

This enhancement was my first real experience working with databases on my own, and diving into Firebase was both exciting and more challenging than I expected. Compared to using local storage with SQLite, Firebase required a completely different approach, and I quickly realized how much more complex cloud-based databases can be. At times, it felt like fixing one bug created three more, which really drove home the importance of being mindful with every change; a small mistake can have a domino effect across the entire system and I'm totally ashamed of how long it took me to find some deceptively simple bugs or mistakes! Despite the challenges, there was a huge sense of accomplishment every time I overcame an issue and got something working. I also gained a better understanding about cloud databases and why they are such a strong choice; they allow for better data security, accessibility across devices, and real-time updates, all of which make my app far more reliable than before. It gives users much more control over their data, which is always a good thing. This experience has definitely strengthened my understanding of database management, debugging, and cloud integration, and while I know there's still more to learn, I now feel much more confident working with Firebase and similar technologies in the future.

Enhancement Summary

Enhancement 1: Conversion from Java to Kotlin (Software Design & Engineering)

- Converted the entire codebase from Java to Kotlin to align with modern Android development best practices.

- Improved code maintainability and readability using Kotlin's null safety, use function, and short syntax.
- Updated dependencies, Gradle configurations, and the compile SDK to API level 35 to ensure compatibility with the latest Android versions.
- Final testing confirmed the conversion was successful, with no errors or issues.

Enhancement 2: Predictive Weight Trend Algorithm (Algorithms & Data Structures)

- Implemented an algorithm to estimate future weight trends based on historical data, providing users with text-based predictions.
- Integrated MPAndroidChart to visually display only recorded weight value in a line graph.
- Added a custom date entry option for weight logs instead of restricting users to the current date.
- Resolved previous graphing issues that prevented accurate trend visualization.
- Removed predicted weight line trend from the graph; shelved for a future update due to visualization issues.
- Final testing confirmed that predictions are now accurate, and all features function correctly.

Enhancement 3: Import/Export Feature (Database & Cloud Integration)

- Integrated Firebase Firestore functionality to allow users to import and export weight data.
- Added Firebase authentication to securely manage user accounts and access control.
- Fixed UI refresh delay issues that previously caused imported data to not display immediately.
- Final testing confirmed that all import/export functionality works as expected, with no major issues remaining.

How to Run the Project

GitHub Repository & Branch Information

The latest version of my project - with the weight prediction algorithm enhancement - is located on the enhancement3 branch of the GitHub repository.

- Repository Name: CS499weighttrackingapp
- Repository Link: https://github.com/smcclurkin0312/CS499weighttrackingapp
- Latest Version (Enhancement 3): enhancement3 branch

The latest version of my project can be ran using the following Git commands:

- git clone https://github.com/smcclurkin0312/CS499weighttrackingapp.git
- cd CS499weighttrackingapp
- git checkout enhancement3

System Requirements

- Android Studio: 2024.2.2
- Kotlin Version: 2.0.21
- Gradle Version: 8.10.2
- Minimum SDK: 31 (Android 12)
- Target SDK: 35 (Android 14)
- MPAndroidChart 3.1.0
- Firebase Firestore & Authentication

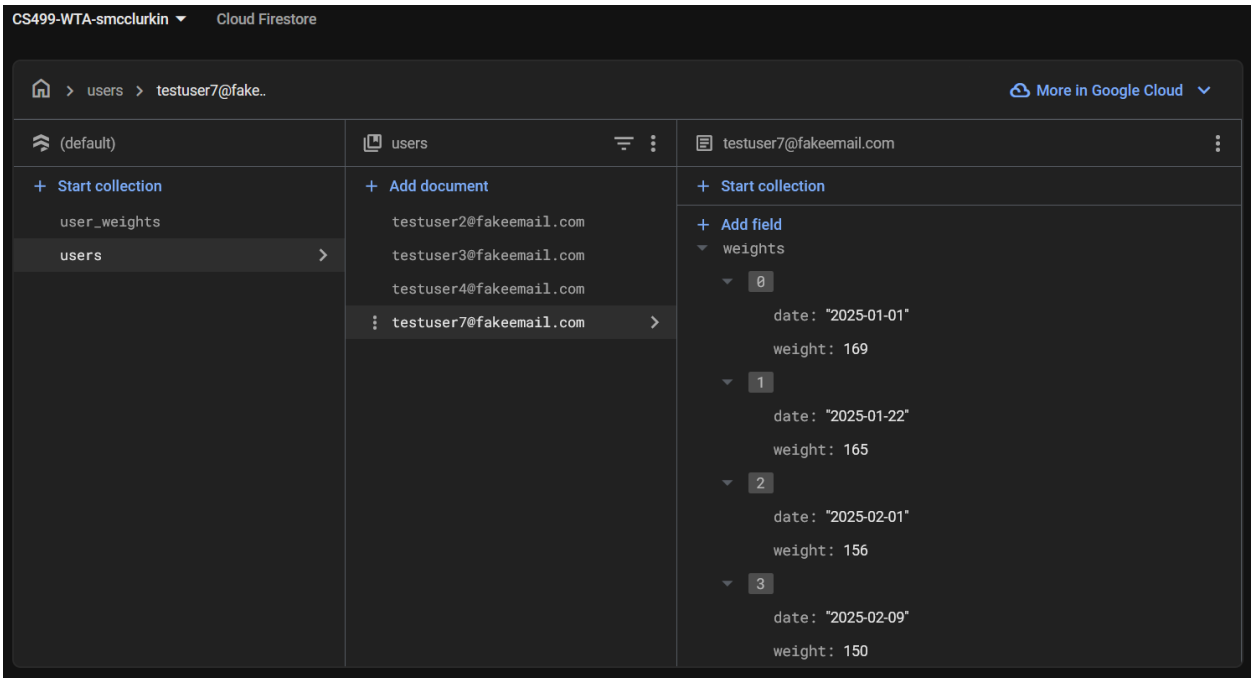
Android Studio

- Open Android Studio and choose to "Open an Existing Project."
- Open the cloned project folder.
- Sync Gradle by selecting Sync Project with Gradle Files from the top navigation bar in Android Studio.

- Click Run to launch the app on Android Emulator (Pixel 6, API 35).

Testing & Evaluation

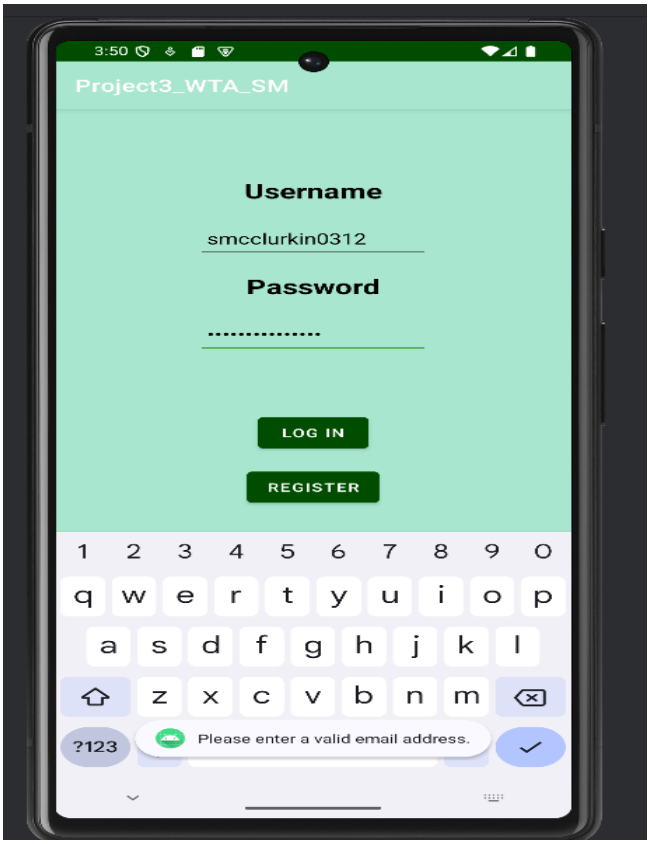
To make sure Firebase authentication and Firestore integration worked correctly, I focused on testing to check functionality, security, and data handling. The main goal was to confirm that users could securely log in, access their weight history, and sync their data between Firestore and the local database without any issues. I also tested different scenarios to see how well the app handled errors and edge cases. First, I focused on testing the core functionality of the app. It was important to verify that users could log in, register, and retrieve their weight data from Firestore without issues. I also tested the import/export feature by backing up weight entries, deleting weight data, and confirming that the data could be successfully imported and synced across devices. As shown in the screenshot below, I tested these features across multiple user accounts and with multiple weight entries to ensure everything worked as expected.



Next, I focused on testing the security of the application. First, I confirmed that Firebase authentication worked correctly and that users had to be logged in to access the application and

their data. I also reviewed my Firestore security rules to ensure they were set up to prevent unauthorized access, meaning that each user should only be able to read and write their own data; you can see these rules again in the screenshot located in the fifth course outcome section above. While I did not attempt to bypass these rules manually, I verified that authentication was required to access Firestore and that login credentials were handled securely.

I made sure that Firestore synced correctly with SQLite and tested multiple weight entries and datasets to make sure Firestore queries were working as intended. Finally, I tested error handling to make sure the app responded properly to incorrect inputs and unexpected situations. When entering invalid login credentials, the app displayed an appropriate error message instead of crashing. You can see a screenshot of one of these tests below. Tests like these confirmed that the app provided clear feedback to users while preventing unexpected behavior.



After running all these tests, everything worked as expected. Firebase authentication ensured that only authorized users could log in, Firestore correctly stored and retrieved weight logs, and security rules were in place to protect user data. The import/export feature allowed users to back up and restore their data successfully, and the app remained stable and user-friendly. These tests confirmed that the enhancement significantly improved security, data accessibility, and reliability.

Sources

Delaney, J. (2018, February 25). *The ultimate beginner's guide to Firebase.* Fireship. Retrieved February 10, 2025, from https://fireship.io/lessons/the-ultimate-beginners-guide-to-firebase/

Google. (n.d.). Firebase guides. *Firebase Documentation.* Retrieved February 10, 2025, from https://firebase.google.com/docs/guides