3-2 MILESTONE

3-2 Milestone Two Enhancement One Software Design and Engineering

CS-499-12473-M01 Computer Science Capstone 2024 C-5 (Sept-Oct)

Michael Zietz

Southern New Hampshire University

Sept 22, 2024

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

I have selected a previous project I created in Python independent of my courses at SNHU to enhance. This artifact served as my first dive into studying Python before returning to higher education and completing my bachelor’s in computer science. The project spans three files with a main class and two subclasses. The current work can replicate a simplified version of classic table-top gaming combat within a Python-based terminal. It records user input to create one or more characters and weapons with unique properties and simulates the combination of character and weapon damage during their turn using rule systems inspired by Pathfinder by Paizo.

* 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?

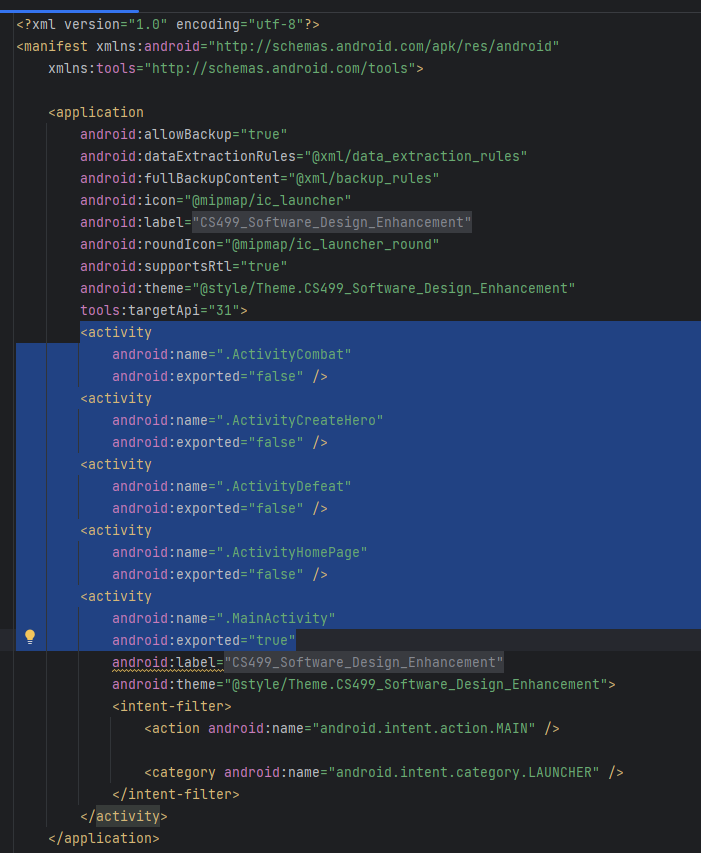
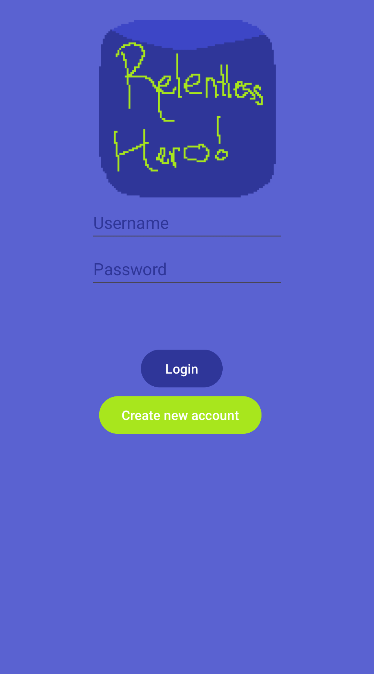
I have recreated this Python project as a Java mobile app using Android Studio to improve the artifact’s base functionality through implementing new features, correcting known issues, and improving both the error handling and comments. New feature additions include opponents, with their own sets of properties, for the characters to simulate combat, and a rounds system to challenge the user’s character with procedurally more difficult enemies to face. Between each round, the user will have the option to fight or flee, and once the user is defeated in combat or flees their first time the simulation ends, showing a collection of statistics their character was able to achieve. The final improvement I seek to add is creating a permanent record outside the application’s run instance that saves the user’s highest recorded attempts.

My intention for enhancing this project is to showcase the growth in my software engineering and design skills because of my educational pursuits through a before and after snapshot. By implementing more complex software engineering concepts, I seek to illustrate my skills in software design and development. I seek to demonstrate a mastery of Python and Java knowledge and a clear understanding of source code and its components and integrating new features as well as adapt an existing code base to specific App Store audiences.

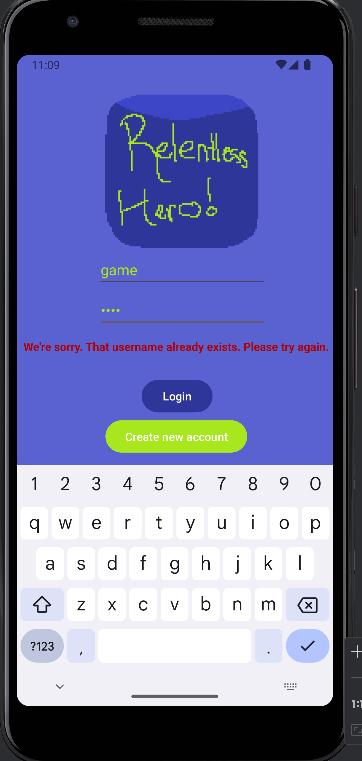
* Did you meet the course outcomes you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?

As for the planned enhancements, I have successfully recreated the core principles of the original artifact in Java and have successfully implemented several of the promised features within the first week of development. All activities have been created in the project’s new code base with proper inclusion in its manifest. Proper means of navigating the app have also been implemented with buttons that accurately describe destinations and next features that will be available to the user. With the addition of SQLite capabilities for score keeping, I also implemented an additional login feature that requires the user to create or use an existing account before engaging with the artifact. This feature was not previously planned as a part of the enhancements, but the included database functionality made this possible. In its current state, the artifact is capable of all CRUD features except. This was left out of the final version as the game has no current need to delete records for security purposes, and the database can be wiped by incrementing the database version. This will be included in the artifact’s final enhancement to allow for proper data upkeep but will not be accessible from the user unless they wish to delete all hero records. Usernames and password will be maintained and unable to be deleted for security purposes.

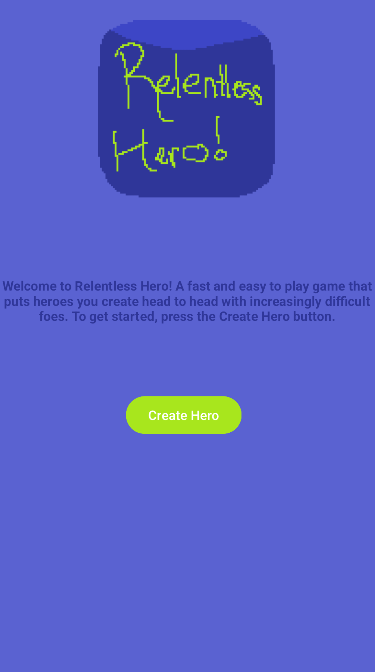
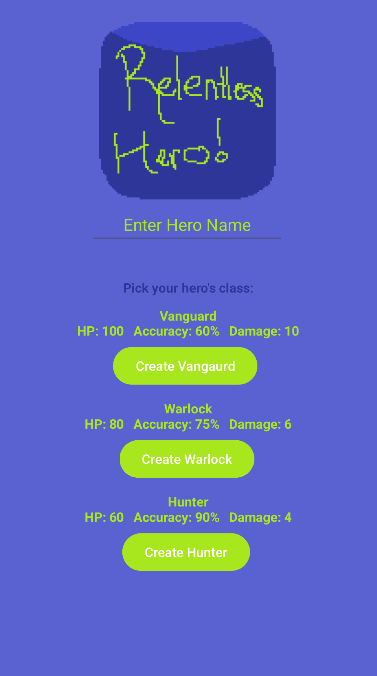
There are a total of nine java classes present spanning from activity classes to parent object classes, to event a recycler view adapter class, and the SQLite database handler. When opening the app, it defaults to open the MainActivty.java which prompts the user to sign in.

  (Origin: AndroidManifest.xml) (Origin: MainActivity.java)

Below is a screenshot of an existing user trying to create a duplicate account. Notice the error message in red stating that the account already exists.

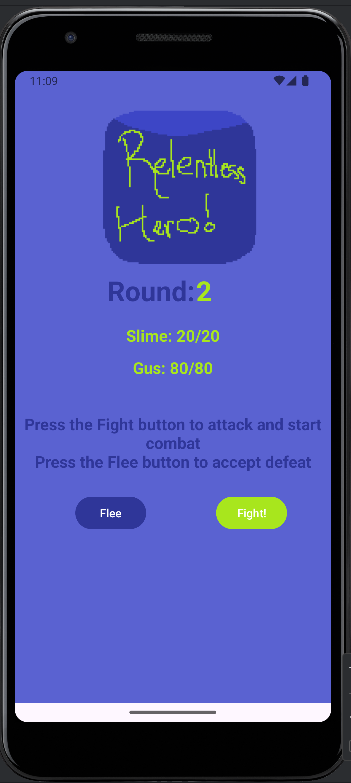
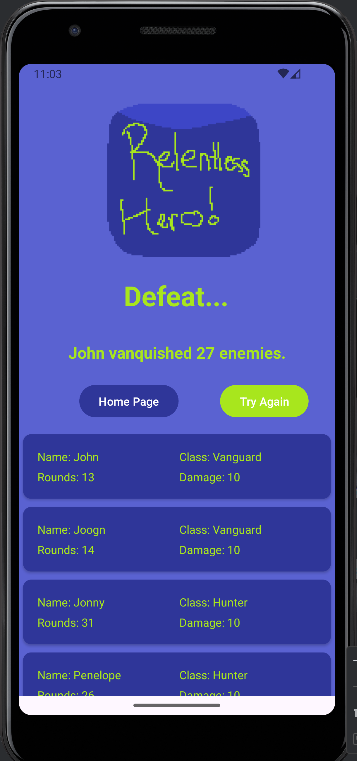
 (Origin: MainActivity.java)

The next screenshots showcase the original artifact’s recreation as an application where the user is able to input details concerning their hero. After logging in, the user is shown a welcome, home screen that describes the app’s purpose before launching them into character creation. Once the button “Create Hero” is selected, the user navigates to the character creation activity where they can give their hero a unique name and chose from one of the three classes available: Vanguard, Warlock, and Hunter. Each class advertises their unique traits in combat, and once a user enters a name, and selects a class button, the application proceeds to the combat.

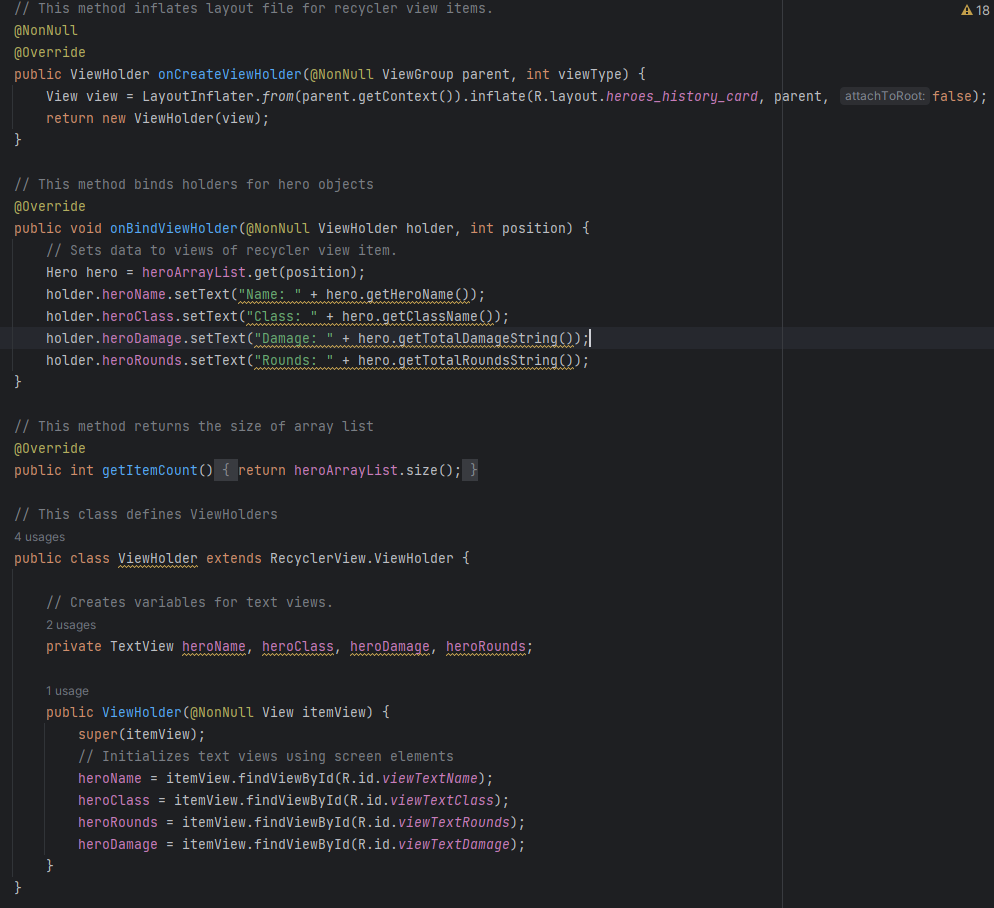
(Origin: ActivityHomePage.java) (Origin: ActivityCreateHero.java)

Once a “Create X” button has been pressed, the next activity is started where the hero is placed in combat against a randomly generated enemy with randomized states. These enemies pull from a list of defined names and utilize random number generation to create statistics like accuracy and damage. In the screenshots below, the combat simulation shows the current and maximum health for each opponent and once the “Fight” button is pressed, damage is dealt to the foe should the strike be accurate enough. The enemy then retorts with their own attack possibly harming the hero. This cycle continues until either the hero has been struck enough times to reduce their health past zero or the user has pressed the “Flee” button. Either way. The next activity is initiated where the user is shown they have been defeated, their character’s name, and how many enemies they were able to defeat. Following this are two buttons to navigate either to the home page or to try again where they can create a new hero and attempt another run. A record is stored of all characters created on the phone in an SQLite table that is used to display their hero history on the defeat screen. The hero’s name, class, how many rounds they succeeded, and their total damage are depicted here using a recycler view class.

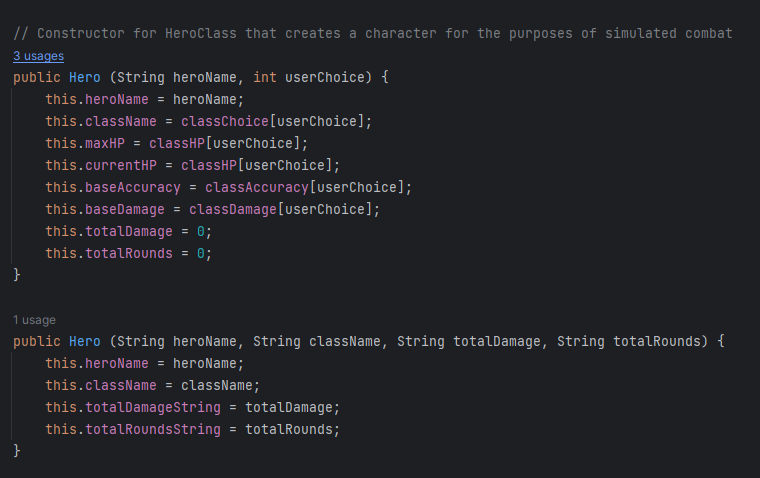
(Origin: ActivityCombat.java) (Origin: ActivityDefeat.java)

This HeroRVAdapter.java class functions by creating an ArrayList of the Hero.java’s objects. The ActivityDefeat.java class then populates each of them with the data stored in the database and selecting the specified string variables.

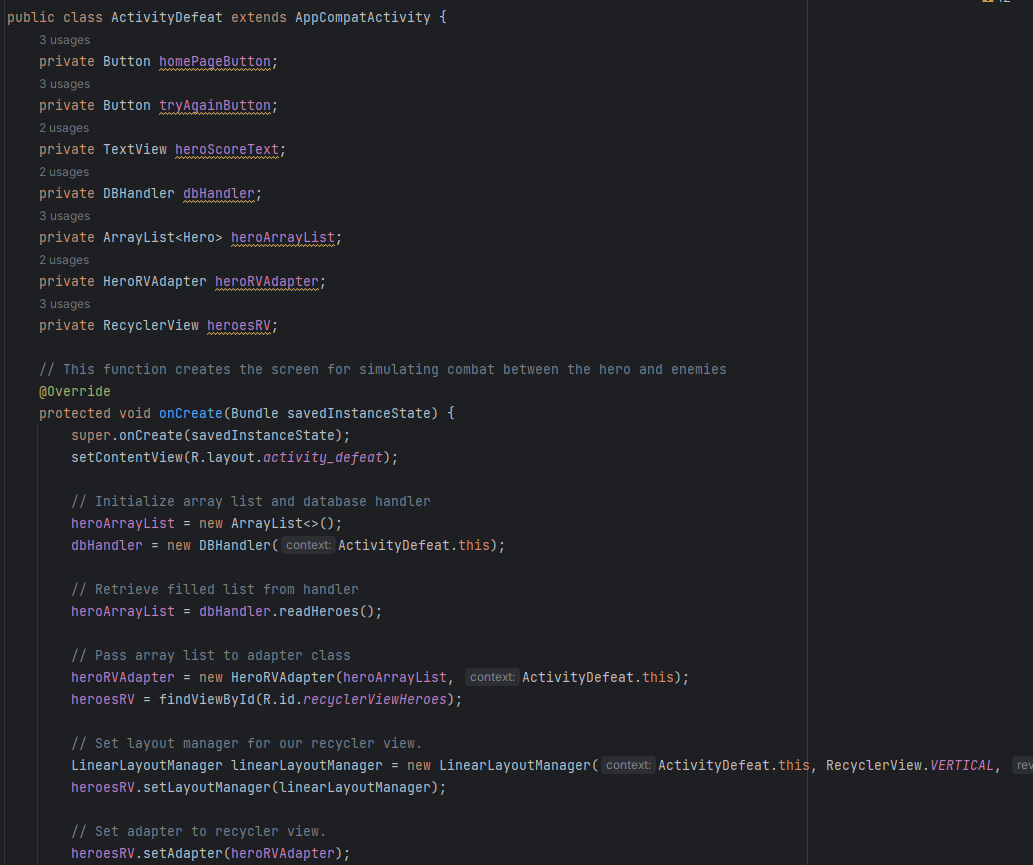


(Origin: HeroRVAdapter.java)

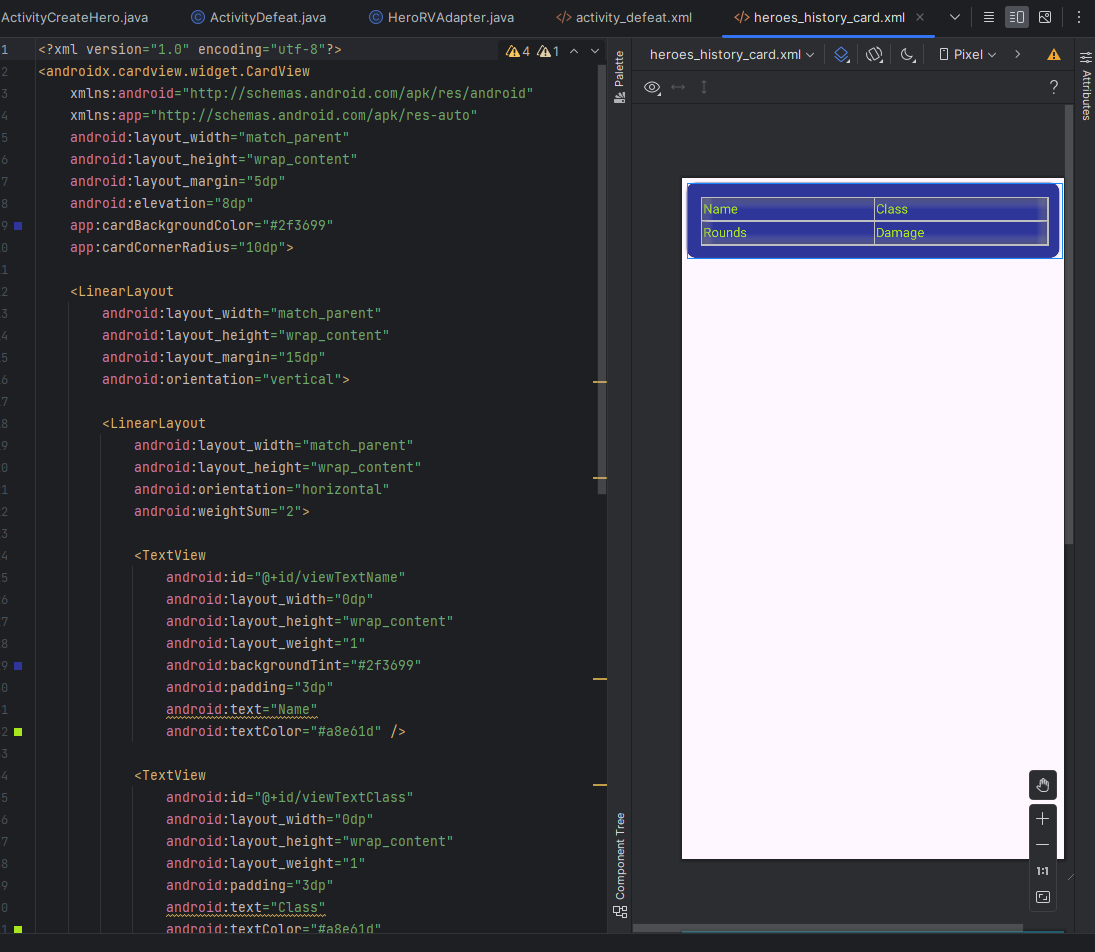
Normally, I would use only one constructor for the Hero.java class, but because SQLite in Java is limited in its data type storage, it can only store a single type. The heroes the user creates have many integers though that need to be stored in the database. To combat this, I converted the integers into a string before storing them in the database and created a separate constructor that only initialized a hero object with strings. This worked as intended because this constructor was only being used once a hero was already populated with data and did not need to start from scratch. In the screenshot below, it showcases the difference between the constructors.



(Origin: Hero.java)



(Origin: ActivityDefeat.java)



* 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?

As for challenges, I surely met my fair share creating the enhancements for this artifact. Because I am rewriting this code base from the ground up in Java, from Python in its original state, building an Android based application, and implementing new features, I had more than enough work lined out for me for this one-week period. The largest issues I faced working on this project were related to implementing a SQLite database and functionality into the app, so previous characters could be stored long term while the application is closed. This introduced a new set of problems that I am still trying to solve because the application needs to store all hero data as a String, due to Java’s data type limitations with SQLite. When attempting to run mathematical calculations related to health changes, damage, and accuracy, strings are not capable of this, so I attempted to import a library named java.lang.Integer.parseInt. However, this is not successful. I was able to fix this issue by reworking how the app handles combat and with what objects. Instead of trying to access the database every round of combat and updating and storing these values, I created local instance of the hero class, similar to the enemy class. Combat using these objects would continue until the hero falls where the live data would then be sent to the database where it would create a new record. Ultimately this improved the speed and clarity of the app allowing for the passing of hero variables between activities, keeping SQLite manipulation to a minimum.