Definition of Done

Project Name: WaveStyled
Team Name: Wave INC: Matthew Daxner, Sanjay Shrikanth, Griffen Shu, Arka Pal, Dhatchi Govindarajan
Release Date: 6/1/2022

Sprint 1:

A. ***(Story Points: 8)*** As a user, I want to be able to add items from my actual wardrobe to my virtual wardrobe so that I can easily visualize everything I own. ***(Priority: 1)***

- ☐ **Acceptance Criteria (User Story):** As a user of the app, I can see my wardrobe on the app once I add a clothing item.
- ☐ **Task 1:** Create a script that generates a random wardrobe that conforms to the database constraints for testing purposes
  - i. **Acceptance Criteria (Task):**
    - A random wardrobe is generated as the Python script is run
- ☐ **Task 2:** Draft a skeleton Python object class framework that encapsulates the clothing item and wardrobe constructs. The class constructors should be able to take in the database data for each item and convert them to use Python objects
  - i. **Acceptance Criteria (Task):**
    - A database is created and the clothing items should be able to be entered into the database and also be converted into Python objects.
- ☐ **Task 3:** Create a Python REST API web service that receives information from the Node.js server and converts the database information into Python objects
  - i. **Acceptance Criteria (Task):**
    - Users should be able to send information from the front end to the Node server. The Node server should be able to receive info and correctly add it to the database.
- ☐ **Task 4:** Add endpoints to the Python framework so that on request, the wardrobe or any specific item can be returned to the user
  - i. **Acceptance Criteria (Task):**
    - Backend is able to send wardrobe items to be returned to the front end upon requests from the front end.
- ☐ **Task 5:** Explore ReactNative and get a rough idea of how to display the planned UI structure on the application(**Spike)**
  - i. **Acceptance Criteria (Task):**

Definition of Done

- Develop a good enough understanding of the front end to make the actual front end for WaveStyled

B. ***(Story Points: 5)*** As a user, I want my wardrobe to be saved when I close the app so that I do not need to re-enter it whenever I reboot the app. ***(Priority: 2)***

- ☐ **Task 1:** Plan out the database table structure and create a SQL database & table that stores all necessary information about items in the wardrobe
  - **Acceptance Criteria (Task):**
    - Users are able to enter information and it is properly and correctly stored in the database.
- ☐ **Task 2:** Create the Node.js server that servers as the middleware of the webservice & facilitates communication between user database
  - **Acceptance Criteria (Task):**
    - Requests from the front end are properly processed by the backend Node server and also is integrated correctly with the database.
- ☐ **Task 3**. Add initial HTTP endpoints to the Node server to begin the communication framework (add, put, delete, etc.)
  - **Acceptance Criteria (Task):**
    - Add all of the GET, POST, UPDATE, DELETE API endpoints required for the frontend

Sprint 2:

A. ***(Story Points: 8)*** As a user I want to see my recommended fits so that I can easily choose(or not) my outfit for the specified weather occasion. ***(Priority: 3)***

- ☐ **Task 1:** Create a separate database on the SQL server that stores outfits and their information, including their pieceid references, color, and clean/dirty tags
  - **i. Acceptance Criteria (Task):**
    - A working database is created and is running to store the attributes of the clothing from a wardrobe.
- ☐ **Task 2:** Add endpoints to the Node and Python server that can send/receive outfit tuples from the database and prepare them for processing
  - **i. Acceptance Criteria (Task):**
    - The backend is able to receive requests for outfits from either the frontend or the database, and is able to be processed for machine learning.
- ☐ **Task 3:** Create an Outfits class in Python that encapsulates outfit data

Definition of Done

i. **Acceptance Criteria (Task):**
   - The python recommender has a dedicated Outfit class with all of the attributes necessary for machine learning.

☐ **Task 4:** Create a random-outfit generator that takes in the outfit data from the database and outputs a sequence of piece-ids for a given weather and occasion (Naive Outfit Recommender on the Python end)

i. **Acceptance Criteria (Task):**
   - Users are able to receive random combinations of clothing items using a naive recommendation algorithm which randomly selects clothes.

☐ **Task 5:** Use Image processing libraries and code a function that takes a sequence of item piece ids (outfits) and displays the corresponding image on the local screen (Backend outfit testing)

i. **Acceptance Criteria (Task):**
   - Users can simply see their outfits on the frontend rather than just reading a description of the outfit.

B. **(Story Points: 13)** As a user I want a well-designed user interface so that I can easily navigate through my wardrobe, recommendations, and previously-chosen outfits **(Priority: 2)**

☐ **Task 1:** Navigate through different screens in the app

○ **Acceptance Criteria (Task):**
   - Users are able to navigate through Home screen, Recommend screen, Calibrate screens.

☐ **Task 2:** Be able to save clothing items details entered by the user

○ **Acceptance Criteria (Task):**
   - Users can upload and store their clothing items into the database using the backend API calls.

☐ **Task 3**: View the entire wardrobe loaded from backend on the app

○ **Acceptance Criteria (Task):**
   - The whole wardrobe is visible on the frontend through card based scrollable UI.

☐ **Task 4:** Store photos when a clothing item is added

○ **Acceptance Criteria (Task):**
   - Users are able to store the description as well as the photo of the clothing item once they upload it to the application.

☐ **Task 5:** Present randomly generated outfits to view on the app

Definition of Done

- **Acceptance Criteria (Task):**
  - Given a user's set of clothing items added on the frontend, the backend is able to use the Python recommender to recommend randomly generated outfits.

Sprint 3:

A. **(Story Points: 21)** As a user I want to be able to calibrate the recommendation model so that it gives me outfits that fit my preference and learn what I like to wear for future recommendations. **(Priority: 1)**

- ☐ **Task 1:** Be able to present random outfits and allow the user to like, dislike, or skip outfits.
  - Acceptance Criteria
    1. When I navigate to the calibrate and recommend screen, I see outfits generated by my wardrobe
    2. In the recommended screen, swiping an outfit to the right should like, and swiping an outfit to the left should dislike. Tapping the checkmark icon should like, tapping the x button should dislike, and tapping the hanger button should make that the outfit of the day
- ☐ **Task 2:** Feed the calibrated data to the recommender model, and update the model based on user preferences.
  - Acceptance Criteria
    1. After liking/disliking outfits in the calibrate screen, I am able to see more outfits in the recommend screen that fit my preferences

B. ***(Story Points: 8)*** As a user, I want to be able to submit a picture of my clothing item so that when I receive a recommendation, I can easily identify which piece of clothing it is referred to and clearly visualize my outfit right on my phone screen. **(Priority: 2)**

Definition of Done

Sprint 4:

(Story Points: 13) As a user, I want a well-designed user interface so that I can easily navigate through my wardrobe, recommendations, and previously-chosen outfits **(Priority: 2)**

- ☐ **Task 1:** Use better styling to make the UI look more appealing and less clutter
  - Acceptance criteria
    1. All screens should have consistent colors and fonts
    2. The outfits in the cards should be centered/well spaced out
    3. All text should be well centered/visible

C. **(Story Points: 8)** As a user, I want to be able to log into my account to access my wardrobe and outfits of the day **(Priority: 1)**

- ☐ **Task 1:** Be able to add user authentication that prompts the user for their email and password to gain access
  - Acceptance Criteria:
    1. When booting the app for the first time, I am able to create an account, provide a valid password, and renter my password
    2. When booting the app with an existing account, I am able to login to the app without creating an account
    3. I am able to navigate to the home screen of the app after logging in successfully
- ☐ **Task 2:** Generate a unique user-id on login so the HTTP requests know which user is sending it
  - Acceptance Criteria
    1. On account creation, a new user entry should be added to the Firebase console under the Authentication tab, which includes the user's email and userID
    2. The userID seen on the Firebase console should be identical to the HTTP request userID parameters seen on the Python Link.py
    3. If the IDs are consistent with the user sending the request login and ID propagation is successful

D. ***(Story Points: 13)*** As a fashion designer, I want to be able to have multiple tuned wardrobes so that I can have different recommendations for different sets of clothes ***(Priority: 3)***

Definition of Done

- ☐ **Task 1:** Improve the recommendation model such that it is able to recommend better outfits tailored to the user preference
    - Acceptance Criteria
        1. When I get recommendations, I expect to see outfits better tailored to my preferences the more I use the app
- ☐ **Task 2:** Come up with more factors that affect outfit preference, and having our product take those into account(requires modifying both recommendation model and UI)
    - Acceptance Criteria
        1. I should be able to add details that affect my preferences of choosing a clothing item and should be recommended fits that take those into account