Requirement ID 1.1: Initial Agile Reference Stories

Ticket Details
Description: Develop a spreadsheet with reference stories grouped into 6 pools (1, 2, 3, 5, 8, 13 story points). This should reflect past assignments or projects and be used as benchmarks when pointing out stories for the project.
AC: The acceptance criteria is that each story should be assigned a reference story number, pool value, story source, and a brief description of the story summary.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin, Kusuma Murthy, Nimra Syed, Nikka Vuong
Estimated Points: 1 point
Actual Points: 1 point
Branch: N/A
Extra Notes: This is Requirement 1.1.

Requirement ID 1.2: Initial Requirements Stack

Ticket Details
Description: As a team, brainstorm requirements for all the Sprints. Provide a brief description of each requirement, estimate each story using planning poker, and prioritize each one.
AC: The acceptance criteria is to create a spreadsheet with the above information.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin, Kusuma Murthy, Nimra Syed, Nikka Vuong
Estimated Points: 3 points
Actual Points: 3 points
Branch: N/A
Extra Notes: This is Requirement 1.2.

Requirement ID 1.3: Initial Architecture Documentation

Ticket Details
Description: Create the Architecture documentation. This includes identifying what the product is, a detailed description of each component, and UML diagrams illustrating how the systems work.
AC: The acceptance criteria is to complete the Architecture documentation.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin, Kusuma Murthy, Nimra Syed, Nikka Vuong
Estimated Points: 3 points
Actual Points: 3 points
Branch: N/A
Extra Notes: This is Requirement 1.3.

Requirement ID 2.1: Sprint 1 Requirements Stack

Ticket Details
Description: As a team, finalize requirements for Sprint 1. Provide a brief description of each requirement, estimate each story using planning poker, and prioritize each one.
AC: The acceptance criteria is to create a spreadsheet with the above information.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin, Kusuma Murthy, Nimra Syed, Nikka Vuong
Estimated Points: 3 points
Actual Points: 3 points
Branch: N/A
Extra Notes: This is Requirement 2.1.

Requirement ID 2.2: Sprint 1 Requirements Artifacts

Ticket Details
Description: As a team, finalize requirements artifacts for Sprint 1. Provide a brief description of each requirement, estimate each story using planning poker, and prioritize each one.
AC: The acceptance criteria is to create a document containing all Sprint 1 tickets.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin, Kusuma Murthy, Nimra Syed, Nikka Vuong
Estimated Points: 3 points
Actual Points: 3 points
Branch: N/A
Extra Notes: This is Requirement 2.2.

Requirement ID 3: Add Tabs for Each Page to Navigation Bar

Ticket Details

Description: Add recommendation and favorites page to navigation bar.
AC: The acceptance criteria is two new tabs added to the navigation bar.

Ticket Logistics
Assignee: Nikka Vuong
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian
Extra Notes: This is Requirement 3.

Requirement ID 4: Remove Unnecessary Question from Profile Questions

Ticket Details
Description: Remove an unnecessary question "How far do you want to travel?". This question's answer is no longer used in our recommendation model, and therefore it is no longer needed.
AC: The question is successfully removed from the frontend and backend, and the question is no longer visible to the user. The backend functionality is preserved and the app as a whole functions properly. The code is also tested.

Ticket Logistics
Assignee: Kusuma Murthy
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian/tree/kusuma-fixQuestionsAndUserAuth
Extra Notes: This is Requirement 4.

Requirement ID 5: Add Drop Down Menu for Countries in Profile Setup

Ticket Details
Description: To ensure that the user selects the correct country from the team's list, implement a drop down selection so the user can type in their country and find the option in the list. This is for the profile setup page.
AC: The user can scroll through the list of countries in the drop down and/or start typing to narrow down the search of their country.

Ticket Logistics
Assignee: Sophia Jacob.
Estimated Points: 1 point
Actual Points: 1 point

Branch: https://github.com/SAJacob7/Elysian/tree/main
Extra Notes: This is Requirement 5.

Requirement ID 6: Add Swipe Left/Right Feature for Recommendations

Ticket Details
Description: The user should be able to swipe left and swipe right on the cities recommended on the recommendations page. Swipe right should store the city to the favorites pages and userFavorites table in the database (this is implemented in ticket 9.1). Swipe left should be stored in the userDislikes table in the database (this is implemented in ticket 9.2). This ticket should also change the single click to a double click since a single click will now signify the start of the swiping gesture.
AC: The users should now be able to swipe right and left on the city. The user should be able to also double click on the city tile in order to access the city information. The swipe left and right feature is integrated correctly with the backend recommendation model. The app is thoroughly tested and there are no bugs.

Ticket Logistics
Assignee: Kusuma Murthy
Estimated Points: 3 points
Actual Points: 3 points
Branch: https://github.com/SAJacob7/Elysian/tree/KusumaSwipeLeftAndRightFeature
Extra Notes: This is Requirement 6.

Requirement ID 7: Create a New Favorites Page

Ticket Details
Description: Load the user's favorite locations from Firebase userFavorites database onto the favorites page.
AC: The acceptance criteria is the user's favorite locations now show up on the favorites page.

Ticket Logistics
Assignee: Nikka Vuong
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian
Extra Notes: This is Requirement 7.

Requirement ID 8: Update Code from Home Page to Favorites Page

Ticket Details
Description: Refactor the home page code and move all of the code to the Favorites page. However, when the user logs into the app, they should still default to the Home page. The Home page should be empty.
AC: All of the code from the Home page is refactored and moved to the Favorites page. The user is still defaulted to the Home page. The Home page should be empty.

Ticket Logistics
Assignee: Kusuma Murthy
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian/tree/KusumaSwipeLeftAndRightFeature
Extra Notes: This is Requirement 8.

Requirement ID 9.1: Create Function to Add/Remove Favorites Cities to/from Firebase

Ticket Details
Description: Create a function to add favorite cities to the Firebase userFavorites collection when users swipe right on the Recommendation page. Create another function to remove favorited cities from the Firebase userFavorites collection when users dislike cities from the Favorites page.
AC: Ensure that cities are updated in the usersCollection based on user actions.

Ticket Logistics
Assignee: Anna Lin
Estimated Points: 3 points
Actual Points: 3 points
Branch: https://github.com/SAJacob7/Elysian
Extra Notes: This is Requirement 9.1.

Requirement ID 9.2: Create Function to Add Disliked Cities to Firebase

Ticket Details
Description: Create a function to add disliked cities to the Firebase userDislikes collection when users swipe left on the Recommendation page.
AC: Ensure that cities are updated in the userDislikes on left swipes.

Ticket Logistics
Assignee: Anna Lin
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian
Extra Notes: This is Requirement 9.2.

Requirement ID 10: Add Descriptions of Each City

Ticket Details
Description: Add short, informative descriptions for each city to help users better understand what makes each destination unique. These descriptions should appear alongside the city recommendations and this enhances the overall user experience without overwhelming the interface.
AC: Each city displays a clear and accurate description, the text loads properly with the recommendations, and the layout remains clean and readable across the app.

Ticket Logistics
Assignee: Nimra Syed
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian/tree/nimra-favorite-updates
Extra Notes: This is Requirement 10.

Requirement ID 11: Research about Dynamic Model Implementation

Ticket Details
Description: Research on how to create a dynamic recommendation model so that when users dislike or favorite cities, their decisions will be taken into consideration by the model.
AC: Do enough research to begin implementing/updating the model.

Ticket Logistics
Assignee: Sophia Jacob
Estimated Points: 1 point
Actual Points: 1 point
Branch: N/A
Extra Notes: This is Requirement 11.

Requirement ID 12: Update Datasets to Account for More Cities/Countries

Ticket Details
Description: Generate more cities to add to the cities.csv dataset to allow the model to generate recommendations for a larger scope of cities.
AC: cities.csv dataset contains more city entries.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian
Extra Notes: This is Requirement 12.

Requirement ID 13.1: Modify Model to Use Two-Tower Method

Ticket Details
Description: To modify the model and make it more accurate for recommendations, the team will create a recommendation-style model using the two-tower architecture approach. This is done to independently process the user queries and the city features, generating embeddings for each, which are then compared to determine their scores for personalized recommendations.
AC: This model should be implemented with the two-tower approach and successfully loaded into the backend with personalized recommendations given for different queries.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin
Estimated Points: 3 points
Actual Points: 3 points
Branch: https://github.com/SAJacob7/Elysian/tree/sophia-dynamic-model
Extra Notes: This is Requirement 13.1.

Requirement ID 13.2: Train Model and Generate City Vectors

Ticket Details
Description: Train the model and store it as a tflite into the backend, while also generating the city vectors to compare the user queries to see which cities match their preferences.
AC: The model should correctly give recommendations and output the city_vectors.npy and the model tflite.

Ticket Logistics

Assignee: Sophia Jacob, Anna Lin
Estimated Points: 3 points
Actual Points: 3 points
Branch: https://github.com/SAJacob7/Elysian/tree/sophia-dynamic-model
Extra Notes: This is Requirement 13.2.

Requirement ID 13.3: Load Trained Model to Backend

Ticket Details
Description: To ensure that the model can be run during app setup, the model needs to be put into the backend and stored in Render.
AC: The tflite model should be light-weight and able to run efficiently in the backend and have endpoints to call to it.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian/tree/sophia-dynamic-model
Extra Notes: This is Requirement 13.3.

Requirement ID 13.4: Load Encoders to Backend

Ticket Details
Description: To encode the user's answers into integers that the model can understand, the team needs to build an encoder list. This list will get pre-built and will be saved as pickle files for the backend to load.
AC: With the completion of the ticket, there should no longer be "Encoder not loaded" issues and the model can still use the inputs encoded to give recommendations.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian/tree/sophia-dynamic-model
Extra Notes: This is Requirement 13.4.

Requirement ID 13.5: Implemented Recommendation Layer on Top of Model

Ticket Details
Description: Compute the cosine similarity scores between each city and the user vector (composed of the user profile, user favorites, and user dislikes). Use the most similar city as the next recommendation. This computation should be done every time the user performs a swipe. This function will be an endpoint used by the frontend.
AC: Ensure that the cosine similarity score is computed.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin
Estimated Points: 3 points
Actual Points: 3 points
Branch: https://github.com/SAJacob7/Elysian/tree/sophia-dynamic-model
Extra Notes: This is Requirement 13.5.

Requirement ID 13.6: Change Logic for Recommendation Page

Ticket Details
Description: Create a function in the Recommendation page called "fetchNextCity" to call the backend API "next_city" to generate recommendations based on the user actions (i.e., left or right swipe). This will make the recommendations more dynamic instead of having pre-generated lists of cities for users to swipe through.
AC: Ensure that the "next_city" endpoint is called properly and that the UI is updated per swipe with the new city.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian/tree/sophia-dynamic-model
Extra Notes: This is Requirement 13.6.

Requirement ID 13.7: Test Recommendation Model

Ticket Details
Description: Test the recommendation model by creating new user accounts with different profiles and checking the recommended cities.
AC: Ensure that the recommendation model is generating a variety of cities that fit the user profile.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian/tree/sophia-dynamic-model
Extra Notes: This is Requirement 13.7.

Requirement ID 14.1: Update UI and Database Logic for the Favorites Page

Ticket Details
Description: Update the Favorites Page to improve both the user interface and how data is stored and retrieved from the database. This includes making the page easier to navigate, ensuring 'liked' destinations load quickly, and keeping the favorites list synced with the user's profile so selections are saved accurately.
AC: The Favorites Page should display all 'liked' destinations correctly, allow users to add or remove favorites without errors, and ensure changes are immediately reflected in the database and user profile.

Ticket Logistics
Assignee: Nimra Syed
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian/tree/nimra-favorite-updates
Extra Notes: This is Requirement 14.1.

Requirement ID 14.2: Add Function for User to Update Name and Username

Ticket Details
Description: User clicks on edit button to edit their name and username on profile page, which will update in the database, as well.
AC: The acceptance criteria is the user can edit their name and username.

Ticket Logistics
Assignee: Nikka Vuong
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian/tree/nikka-updateprofile

Extra Notes: This is Requirement 14.2.

Requirement ID 14.3: Update UI for the Profile Page

Ticket Details
Description: Move Q/A from their profile setup and logout button to a menu pop up.
AC: The acceptance criteria is the Q/A and logout button are on the menu button.

Ticket Logistics
Assignee: Nikka Vuong
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian/tree/nikka-updateprofile
Extra Notes: This is Requirement 14.3.

Requirement ID 15: Research About Cloud Storage

Ticket Details
Description: Research about viable cloud storage options that will be compatible with our app architecture and backend. Current options are: AWS, Firebase Storage, Cloudinary.
AC: A cloud storage platform is chosen and it is compatible with our app architecture. If the software needs to be bought then it should be approved by Karla Cordes.

Ticket Logistics
Assignee: Kusuma Murthy
Estimated Points: 1 point
Actual Points: 1 point
Branch: N/A
Extra Notes: This is Requirement 15.

Requirement ID 16.1: AWS Account Setup

Ticket Details
Description: The AWS account should be purchased using the card provided by Karla. Next, the AWS account should be set up with an S3 bucket, an IAM role, Lambda, and an API gateway. The S3 bucket is where all of the images the user uploads should be stored. A custom API needs to be created so that when users upload a picture to the app, they call the upload API endpoint, which then triggers the Lambda function. Next, a customized Lambda function along with an

IAM role needs to be created. The lambda function generates a temporary pre-signed URL that allows the user to securely upload images directly to an Amazon S3 bucket without revealing the AWS credentials in the app.
AC: All four AWS services are set up in the AWS account.

Ticket Logistics
Assignee: Kusuma Murthy
Estimated Points: 3 points
Actual Points: 3 points
Branch: N/A
Extra Notes: This is Requirement 16.1.

Requirement ID 16.2: AWS Image Storage Setup

Ticket Details
Description: Integrate AWS into the ExpoGo app and with Firebase. In this ticket, the developing team should be able to send test requests to AWS using the AWS API, and this should trigger the Lambda function. The Lambda function, which uses IAM roles, should create a temporary signed URL that allows the user to securely upload images to the S3 bucket storage. The overall backend flow of AWS should be integrated into the backend of the mobile app.
AC: The backend flow of AWS is integrated into the backend of the app.

Ticket Logistics
Assignee: Kusuma Murthy
Estimated Points: 1 point
Actual Points: 1 point
Branch: N/A
Extra Notes: This is Requirement 16.2.

Requirement ID 17: Change API for City Images

Ticket Details
Description: Update the API used to fetch city images to improve image quality. Have to ensure the new API integrates smoothly with the app and displays relevant images for each destination.
AC: City images load successfully from the new API, match the correct destinations, maintain good quality, and do not negatively impact app performance.

Ticket Logistics
Assignee: N/A

Estimated Points: 1 point
Actual Points: 1 point
Branch: N/A
Extra Notes: This is Requirement 17.

Requirement ID 18: Limit Length of City Descriptions

Ticket Details
Description: Set a character or word limit for city descriptions to keep the interface clean and easy to read. This ensures users can quickly scan destination details without large blocks of text disrupting the layout.
AC: City descriptions do not exceed the defined length, longer text is properly truncated (with ellipses if needed), and the UI remains consistent across all devices and screens.

Ticket Logistics
Assignee: Nimra Syed
Estimated Points: 1 point
Actual Points: 1 point
Branch: https://github.com/SAJacob7/Elysian
Extra Notes: This is Requirement 18.

Requirement ID 19: Implement a Search Bar to Search for Cities

Ticket Details
Description: If the user would like to specifically look for a city they know they want, then implement a search bar to find these cities and save them.
AC: The feature should allow searching for cities, liking them, and saving them into favorites.

Ticket Logistics
Assignee: Nikka Vuong
Estimated Points: 1 point
Actual Points: N/A
Branch: N/A
Extra Notes: This is Requirement 19.

Requirement ID 20: Update UI for Recommendation Page

Ticket Details

Description: Update the UI for the Recommendation page so that the city card fills up the entire page. Also, make the UI look smoother and more aesthetic.
AC: Ensure UI on the Recommendation page is user-friendly and aesthetic.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin
Estimated Points: 1 point
Actual Points: N/A
Branch: N/A
Extra Notes: This is Requirement 20.

Requirement ID 21: Refactor Code for Firebase Calls

Ticket Details
Description: To make the team's code more efficient and reusable, follow a layered architecture approach. This will allow all database calls to be handled by the backend.
AC: The database calls should be efficient and fully handled by the backend, not the frontend.

Ticket Logistics
Assignee: Sophia Jacob, Anna Lin
Estimated Points: 1
Actual Points: N/A
Branch: N/A
Extra Notes: This is Requirement 21.