Student Name	Date	Number of Hours	Description of Work Done
Everyone	Jan 17, 2025	1	Meeting. Distribution of work. Choosing team lead. Decide on the app. -> GitHub Repo -> Misc -> Applied Research_ Logo and Name Research.pdf
Russell Han Josef	Jan 18, 2025	1	Planning on the project scope and role distribution.
Russell Han Josef	Jan 20, 2025	1.5	Research on screens to have. Finding screens inspiration. App name and branding.
Simone Lue	Jan 20, 2025	1	Project Proposal Draft writing - Started the introduction section.
Simone Lue	Jan 21, 2025	2	Project Proposal writing - Finalizing draft writing -> Proposed Research Project, Project Planning and Timeline, Project Contract - Proposal in Github Repo -> ReportsAndDocuments -> SimoneL_Proposal.pdf
Simone Lue	Jan 22, 2025	2	Figma initialization, developing wireframes, general idea of screens
Russell Han Josef	Jan 22, 2025	2	Research on Heroku Dynos and Postgres workflow
Simone Lue	Jan 22, 2025	1	Wireframe for homepage
Russell Han Josef	Jan 22, 2025	1	Researched about conducting surveys and user testing.
Russell Han Josef	Jan 22, 2025	1	Created survey questionnaires and shared using Microsoft Forms. -> Github Repo -> Misc -> Form_Exploration App Survey_ Help Us Build Your Dream Travel Planner.pdf
Simone Lue	Jan 24, 2025	1	Wireframing for quiz screens
Simone Lue	Jan 25, 2025	0.5	Finalized proposal, ready for submission
Russell Han Josef	Jan 25, 2025	2	Commit Git for Proposal.pdf. Researched for APIs: OpenAI gpt model to use, Google Places, Google Maps, Eventbrite, OpenWeatherMap. Get all the API Keys needed.
Simone Lue	Jan 27, 2025	2	Wireframing screens: interactive maps, chatbot, my trips -> Github Repo -> Misc -> Figma Wireframes.png
Everyone	Jan 27, 2025	0.5	Regroup to discuss about the progress and knowledge sharing: Wireframe, APIs.
Russell Han Josef	Jan 28, 2025	0.5	Get Free credits from Heroku account using GitHub for Student Developer Pack
Russell Han Josef	Jan 28, 2025	2	Researched on Database Schema. Learned more about PostgreSQL vs Microsoft SQL vs MySQL. Then Firebase NoSQL. Learned about Hybrid Architecture Approach with both PostgreSQL and NoSQL. Exported 2 diagrams. -> GitHub Repo -> Misc -> WayPoint-SQL-Schema.png -> GitHub Repo -> Misc -> WayPoint-NoSQL-Schema.png
Russell Han Josef	Jan 28, 2025	1	Created Video to share surveys on Instagram. Link: https://www.instagram.com/reel/DFXOyODRLOl/?igsh=ZnF4OWhqbTk0NnVx
Russell Han Josef	Jan 29, 2025	1	Get more insights from Prof. Priya in how do we approach surveys collected and user testing if user doesn't reside in a same country.
Simone Lue	Jan 31, 2025	3	Research on Reactive Native Expo vs CLI, environment setup and ways to start a project + gluestack v2 ui. Decided on React Native CLI as it was more suitable for the project scope
Simone Lue	Feb 01, 2025	2	Started and added frontend project shell to github. Encountered issues with the folder structuring when adding to github: Empty folder was being pushed to github instead of with the code. Had to restructure multiple times before successful
Russell Han Josef	Feb 03, 2025	1.5	Created backend shell. Tested the shell and it's running. Pushed to GitHub repo.
Everyone	Feb 03, 2025	1	Regroup to discuss about Database Schema of PostgreSQL and NoSQL. Update on the frontend shell. Update on the backend shell. Troubleshoot how to run ios on the machine on the first time pulling. Ensure frontend shell can work on both machines. Tasks assignment for the upcoming week.
Russell Han Josef	Feb 04, 2025	1	Frontend shell wasn't working on Russell's machine. Troubleshooting. Cocoapods installed but pod wasn't installed successfully. Finding: XCode wasn't install properly. When running xcode-select -p on terminal. It showed other thing than "/Applications/Xcode.app/Contents/Developer" Details: The problem was that the path to the Xcode command-line tools was not correctly set, causing the xcrun command to be unable to locate the iOS SDK (iphoneos). This resulted in the error message: SDK "iphoneos" cannot be located. Code to run: sudo xcode-select -s /Applications/Xcode.app/Contents/Developer
Russell Han Josef	Feb 04, 2025	2.5	I encountered an issue while deploying my FastAPI app to Heroku. The deployment failed with a ModuleNotFoundError for the backend module. After investigating, I realized that the folder containing the app was named Implentation with a capital "I", but the Procfile was referencing it as implementation with a lowercase "i". Instead of renaming the folder, I updated the Procfile to correctly reference the folder name with the capital "I" as it appeared in the project. After this update, the deployment was successful, and the app was properly hosted on Heroku.

Russell Han Josef	Feb 04, 2025	1	Problem: Setting up PostgreSQL was challenging, especially connecting pgAdmin to both my local database and Heroku's remote database. I struggled with authentication issues, missing roles, and ensuring my tables were correctly created in both environments. Solution: I configured my local PostgreSQL by setting the correct roles and connected pgAdmin to Heroku using the provided DATABASE_URL. I ensured the database schema was consistent across both environments and created tables using SQLAlchemy. Explanation: This helped me understand how PostgreSQL differs locally and on Heroku, how to manage database credentials, and how to properly set up pgAdmin for database administration.
Russell Han Josef	Feb 04, 2025	1	Problem: Deploying to Heroku failed due to an incorrectly placed Procfile, missing dependencies in requirements.txt, and misconfigured environment variables like DATABASE_URL and SECRET_KEY. Solution: I moved Procfile to the root directory, updated requirements.txt, and set DATABASE_URL correctly in Heroku's environment variables. Restarting the Heroku dyno applied these fixes. Explanation: This taught me the importance of directory structure and configuration files in deployment and how to debug deployment failures using heroku logs --tail.
Russell Han Josef	Feb 04, 2025	2	Problem: After fixing deployment, my app still crashed on Heroku (H10 App Crashed) due to SQLAlchemy not recognizing Heroku's DATABASE_URL format and FastAPI failing to bind to the correct port. Solution: I modified db.py to convert postgres:// to postgresql://, ensuring SQLAlchemy could connect. I also updated Procfile to bind FastAPI to Heroku's $PORT. Explanation: This reinforced the differences between local and production environments, the need for dynamic configurations, and how Heroku manages deployments and environment variables.
Simone Lue	Feb 06, 2025	1.5	Started on the Quiz Screen component: Adding pressable buttons and sorting out the general layout of the text and button
Simone Lue	Feb 07, 2025	2.5	Completed QuizScreen and the styling file for it. Added: - Back/Next button for navigation between questions. - Back button not visible on the first question - Progress bar to visually represent questions done/left - Color change when button is selected - Have a couple of example questions set as placeholders
Simone Lue	Feb 08, 2025	2	Started on the layout of the HomeScreen as well as its corresponding styling file. Added: - Search bar (text input) - Horizontal scroll for 'My Trips' (with placeholder trips set + default card for when there are no current trips added) Problems encountered: - Had trouble with using Carousel (react-native-snap-carousel) - The dependencies had conflicts and version compatibility issues, specific issues with propTypes - Nothing was working Alternate solution: - Used FlatList with horizontal scrolling instead
Simone Lue	Feb 08, 2025	2	Started on the bottom navigation menu -> Problem: Tried using external library for icons but icons would not show up properly (Question mark in the middle of a box shows up) - Issues with linking fonts with Xcode Alternate plan: For now I am using emojis as a placeholder icon, will revisit adding external library of icons at a later time Bottom Navigation now has seamless navigation between Home and Quiz screens -> Quiz screen to be replaced at a later time, currently acting as a placeholder for other screens
Russell Han Josef	Feb 09, 2025	1	Created travel style quiz. Total 7 questions. Scoring system using point-based system.

Everyone	Feb 10, 2025	2	Regroup / Knowledge Sharing session. Russell: - Explained how the backend works for PSQL part. - Explained how to use pqAdmin tool as database management tool. - Explained the workflow with backend: when writing new code for backend, test it on local machine before pushing to Github. - Showed how deployment works with Github - Heroku setup. General: - Discussed the next workflow in tackling MVPs. - Discussed to re-order MVP priority based on survey. Task Assignments:
Simone Lue	Feb 11, 2025	2.5	Started on the backend for the quiz MVP -> quiz model, schema, api and routing Problem: Had trouble with the database setup/running it on my local machine and connecting to postgresql Solution: After consulting with Russell on the setup, server is able to run on my local device. Next step is to make sure that front end is connecting properly to the back end and able to pass information -> update to local database and make sure it is functioning as intended
Russell Han Josef	Feb 11, 2025	3.5	1. Foreign Key Dependency Issues Problem: Models had incorrect import order, causing foreign key errors. Solution: Adjusted import order in __init__.py to ensure dependencies load correctly. 2. Circular Import Issue Problem: Importing Base from db.py led to circular dependencies. Solution: Moved Base to base.py and updated model imports. 3. Missing email-validator Error Problem: FastAPI required email-validator, despite being in requirements.txt. Solution: Reinstalled dependencies manually on Heroku. 4. uvicorn: command not found on Heroku Problem: uvicorn was missing in the runtime environment. Solution: Updated Procfile to use python -m uvicorn and verified installation. 5. Heroku App Not Restarting Properly Problem: Deployment changes weren't reflecting. Solution: Restarted the app and purged Heroku build cache. 6. App Not Binding to $PORT Problem: FastAPI wasn't binding correctly to the environment port. Solution: Ensured uvicorn runs with --port=${PORT} in Procfile. 7. Database Connection Test Failed on Heroku Problem: Remote database connection wasn't verifying. Solution: Created /test-db endpoint and confirmed it works.
Russell Han Josef	Feb 11, 2025	0.5	Running db.py to Create Tables on Heroku: Problems & Solutions 1. Running db.py on Heroku caused ModuleNotFoundError: No module named 'app' Fix: Used PYTHONPATH=. python app/db/db.py to ensure the correct module path. 2. Tables were not appearing in Heroku Postgres after running db.py Fix: Explicitly set Base.metadata.schema = "public" in db.py to ensure tables are placed in the correct schema. 3. Needed a way to manually trigger db.py on Heroku Fix: Opened a Heroku shell with heroku run bash -a waypoint-travel, then executed: PYTHONPATH=. python app/db/db.py 4. Wanted to verify if tables were created in Heroku Postgres Fix: Used Heroku Postgres CLI to check tables: heroku pg:psql -a waypoint-travel SELECT tablename FROM pg_tables WHERE schemaname = 'public'; Final Outcome: Successfully ran db.py on Heroku, ensuring tables were created in the correct schema.

Russell Han Josef	Feb 12, 2025	1.5	Summary of Fixes & Progress 1. CRUD Implementation for Users Created POST /users → Create User (with password hashing). Created GET /users/{user_id} → Retrieve User by ID. Created PUT /users/{user_id} → Update User (name, email, password). Created DELETE /users/{user_id} → Delete User. 2. Fixed Errors InvalidRequestError → Added ForeignKey("users.id") in quiz_model.py. TypeError: 'password' is an invalid keyword argument for User → Ensured password_hash is used in user_model.py. NameError: name 'user_schema' is not defined → Fixed incorrect import in user_routes.py. zsh: no matches found: passlib[bcrypt] → Installed using pip install "passlib[bcrypt]". 3. Fixed Duplicate URL Path Issue Issue: "/users/users/{user_id}" in FastAPI /docs. Fix: Removed redundant /users prefix from routes in user_routes.py. 4. Tested Locally Verified all CRUD operations using FastAPI /docs. Confirmed correct URL paths after fixing duplication. Next Steps Test CRUD operations for Itineraries, Places, Badges, and Quiz Results. Once confirmed, deploy to Heroku and re-test on live API.
Russell Han Josef	Feb 12, 2025	1	Log Summary for Places CRUD Implementation 1. Implemented CRUD for Places Created POST /places → Add a new place. Created GET /places/{id} → Retrieve a place by ID. Created PUT /places/{id} → Update place details. Created DELETE /places/{id} → Remove a place. 2. Fixed Issues Fixed timezone inconsistency → Ensured last_updated is stored in UTC. Resolved datetime.utcnow() deprecation warning → Used datetime.now(timezone.utc).replace(tzinfo=None). Verified timestamps consistency → Matched last_updated with created_at format. 3. Successfully Tested POST /places → Verified place creation with manual data. GET /places/{id} → Retrieved created places correctly. PUT /places/{id} → Updated place details without timezone mismatch. DELETE /places/{id} → Successfully removed places from the database. Next Steps Implement User Favorites (user_favorite_routes.py). Ensure Users ↔ Places relationship works correctly. Prepare for Google Places API integration.
Russell Han Josef	Feb 12, 2025	0.5	Log Summary for User Favorites Implementation 1. Implemented CRUD for User Favorites Created POST /user_favorites → Add a place to favorites. Created GET /user_favorites/{user_id} → Retrieve a user's favorite places. Created DELETE /user_favorites/{favorite_id} → Remove a favorite place. 2. Fixed Issues Validated user and place existence before adding a favorite. Prevented duplicate favorites by checking existing records. Ensured added_at timestamp is stored in UTC for consistency. 3. Successfully Tested POST /user_favorites → Added places to favorites successfully. GET /user_favorites/{user_id} → Retrieved correct favorites for users. DELETE /user_favorites/{favorite_id} → Removed favorites as expected. 4. Updated main.py Included user_favorite_routes in FastAPI router. Next Steps Implement Badges (badge_routes.py). Ensure User ↔ Badges relationship works correctly.

Russell Han Josef	Feb 12, 2025	1.5	Set Up Google Places API Integration Chose Google Places API (Old Version) for simpler API key authentication. Tested API manually using Postman & cURL. Implemented FastAPI Route for Places Search Created /places/search endpoint to fetch nearby places. Integrated Google Places API (maps.googleapis.com). Cached results in PostgreSQL to reduce API calls. Restricted API to British Columbia (BC), Canada Implemented latitude/longitude boundary check to block requests outside BC. Verified restriction by testing New York (Successfully blocked). Error Handling & Optimizations Improved handling for invalid locations and API failures. Implemented database caching to avoid redundant API requests. Added X-Goog-FieldMask to optimize API responses. Tested & Debugged API Responses Verified working results for Vancouver, BC. Ensured API key security using environment variables (.env, Heroku Config Vars).
Russell Han Josef	Feb 13, 2025	1.5	Tasks Completed: - Configured React Navigation with StackNavigator & BottomTabNavigator. - Created Login & Signup screens with placeholder values. - Implemented Profile screen with user details, Edit Profile (future), and Log Out. - Built Settings screen with Travel Style, Notifications, Language, and Account Management. - Used SafeAreaView & ScrollView to fix UI layout issues. - Replaced deprecated Picker with @react-native-picker/picker. - Ensured dynamic spacing to prevent overlap with iPhone Dynamic Island. Next Steps: - Implement Edit Profile feature. - Add form validation for Login & Signup. - Prepare backend integration for authentication and profile updates. - Enhance UI with better styling.
Russell Han Josef	Feb 13, 2025	0.75	Tasks Completed: - Backend Integration for Authentication: - Reviewed backend schemas, models, and routes. - Confirmed API endpoint for user registration (POST /users/). - Identified and fixed login API endpoint (POST /users/auth/login). - Updated SignupScreen.js: - Connected to backend (POST /users/) for user registration. - Handled form submission, API request, and error handling. - Added navigation to Login screen upon successful signup. - Updated LoginScreen.js: - Integrated POST /users/auth/login using query parameters. - Ensured login request matches the correct FastAPI route. - Redirects users to Main app upon successful login. - Displays alerts for errors and invalid credentials. - Backend API Testing & Debugging: - Successfully tested user registration and login via FastAPI. - Ensured POST /users/auth/login worked with query parameters. - Verified API response handling in React Native app. Next Steps: - Implement persistent authentication (store session/token). eg. JWT - Add form validation for signup & login fields. - Enhance UI styling & error messages for better user experience. - Implement Edit Profile feature in ProfileScreen.js.

Russell Han Josef	Feb 13, 2025	2.25	Attempted to rename React Native app from "frontend" to "WayPoint" → Encountered issues, reverted to "frontend" Updated package.json and app.json to reflect the correct app name → Reverted due to build errors Checked and updated Xcode Signing & Capabilities → Used free Apple ID for provisioning Attempted to set correct Bundle Identifier for Firebase setup → Reverted due to build failures Installed Firebase dependencies (@react-native-firebase/app) → Successfully installed Fixed CocoaPods issues with modular headers → Modified Podfile and ran pod install --repo-update Configured Firebase in AppDelegate.swift → Updated to FirebaseApp.configure() Encountered xcodebuild error code 65 while running iOS build → Attempted multiple fixes Deleted and reinstalled CocoaPods, node_modules, and Xcode DerivedData → No success Manually deleted ios/build/ and cleaned Xcode project → Issue persisted Ran xcodebuild clean and pod install --repo-update → Did not resolve the issue Tried running the app via Metro Bundler (npx react-native run-ios) → Still failed Decided to fully reset the project by deleting and reinstalling all dependencies → Still encountering build issues
Simone Lue	Feb 13, 2025	3.5	- Built the look for the quiz results on Figma - Adding the code and styling for the results display to QuizScreen.js and QuizScreenStyles.js - Will dynamically display the type of traveler depending on the quiz scoring logic from the quiz - Added an “x” button that will take user back to the home page after seeing the results Problems Encountered: - Tried to do a share button functionality where user will be able to save the results as an image to their camera roll. This didn’t work out as there were Xcode dependencies that conflicted with my system. Had to remove this functionality. Can revisit once the MVP is complete - Dependencies that were added to do this share implementation were not completely erased and repeatedly threw errors as it was still somewhere in the files. Solutions and Alternatives: - Had to remove node modules and reinstall dependencies and pod files multiple times to get rid of the errors. Next Steps: - Send quiz results to the backend and save it to the database - Have the option to retake quiz and erase results previous from the backend - (Optional once MVP is complete) Retry the share functionality
Russell Han Josef	Feb 14, 2025	4	Summary Log: Firebase Realtime Database Setup & Next Steps Problems: Multiple React-Core dependencies causing conflicts React-RCTAppDelegate not linking correctly FirebaseAuth/FirebaseAuth-Swift.h file not found (even though not needed) Xcode build error: "unable to initiate PIF transfer session" ReactCommon module redefinition error CocoaPods installation issues Solutions Attempted: Refactored Podfile to use use_modular_headers! and fixed React-Core conflicts Updated AppDelegate.swift with FirebaseApp.configure() Removed and reinstalled dependencies (node_modules, Pods, Podfile.lock) Cleared Xcode cache (DerivedData, xcodebuild clean) Ensured only needed Firebase modules were installed What's Next: Start fresh to ensure a clean build Get React Native running first before adding Firebase Verify Podfile with default settings, then add Firebase Test a basic build (npx react-native run-ios) before integrating Firebase features Implement Firebase Realtime Database CRUD to confirm it works Next attempt: Clean setup from the beginning

Russell Han Josef	Feb 14, 2025	1.5	Work Log: Firebase Integration in React Native (iOS) Problems & Solutions Firebase Not Initializing (No Firebase App '[DEFAULT]' has been created) Firebase was not auto-detecting GoogleService-Info.plist. Manually initialized Firebase in firebase.js. Firebase connected successfully using manual config. Missing or Invalid FirebaseOptions property 'apiKey' Error Firebase could not find apiKey from GoogleService-Info.plist. Verified plist format and corrected key names. Ensured plist was inside Implementation/frontend/ios/. Linked plist in Xcode under Build Phases → Copy Bundle Resources. Still using manual config; plist auto-detection needs verification. Firebase Data Not Appearing in Realtime Database Firebase connection worked, but no data appeared. Updated Firebase database rules to allow reads/writes. Created a test function in LoginScreen.js to write data. Confirmed successful data write to Firebase Console. Next Steps Remove manual Firebase config and verify plist auto-detection. Fetch and display a list of data from Firebase. Secure Firebase Database rules based on authentication.
Russell Han Josef	Feb 14, 2025	0.5	Work Log: Firebase Auto-Detection Fix in React Native (iOS) Problems & Solutions **Firebase Not Initializing Automatically** Firebase was not detecting GoogleService-Info.plist. Manually initialized Firebase in AppDelegate.swift. Confirmed Firebase auto-detection now works. **Missing Firebase Setup in AppDelegate.swift** React Native Firebase requires Firebase to be initialized in AppDelegate.swift. Added FirebaseApp.configure() inside didFinishLaunchingWithOptions. Restarted the app and confirmed successful Firebase initialization. **Plist File Not Being Read by Xcode** GoogleService-Info.plist was not linked in Build Phases → Copy Bundle Resources. Manually added the plist file in Xcode. Verified correct plist location in Implementation/frontend/ios/. Next Steps Fetch and display data from Firebase in the app. Secure Firebase database rules based on authentication.

Simone Lue	Feb 15, 2025	2	Trying to send data to the backend and save the travel style pertaining to the user - Using AsyncStorage to save and retrieve the user_id upon successful logins Problem: - There were AsyncStorage runtime errors due to it not being properly linked - This was due to Cocoapods gem error -> broken or outdated Ruby gem Solution: - Reinstalled and updated ruby gem - Reinstalled cocoa pods - Reinstalled iOS pods - No runtime errors for AsyncStorage now Trying to retrieve userId from the user’s login session - Created a method in QuizScreen.js to retrieve userId - Successfully able to retrieve userid Trying to send quiz results to the backend - Modified api for posting quiz results - Created method in QuizScreen.js to send results to the backend Problem: - 404 Error: api endpoint does not exist or incorrect Next step: Fix the connection to API endpoint
Simone Lue	Feb 16, 2025	1.5	Fixing the connection to API endpoint - Need to send the quiz results to the back end Problem: - 404 Error: api endpoint does not exist or incorrect Solution: - main.py had “/quiz_results” as the router prefix - quiz_routes.py had “/quiz_results” as the route - To send the results to the quiz_results table, the quiz results endpoint should be `${API_BASE_URL}/quiz_results/quiz_results` instead of ‘${API_BASE_URL}/quiz_results` - Changed the endpoint to “/“ --> proper full endpoint is now `${API_BASE_URL}/quiz_results/` - The results are now stored in the quiz_results table with the correctly retrieved userId and travelStyle When the user redoes the quiz, the results save as a new row in the table rather than updating the existing record - Need to update the record instead of making a new row - Modified POST “/“ in quiz_routes.py to check if a result already exists for the user, update it if it exists - Now successfully modifies the record instead of creating a new one Next Steps: - Add frontend option for user to retake quiz. - Fix scoring logic for when users navigate backwards during the quiz. - Users should not be able to move on to the next question without selecting an answer
Simone Lue	Feb 17, 2025	1	Added frontend option for user to retake quiz: - Added retake quiz button code and styles - Added a method to trigger onPress to handle resetting all scores quiz UI - New quiz results are being sent to backend, updating the user’s travel style Next/Submit buttons are ‘disabled’ without selecting an answer - Added disabled button styling - Modified handleNextQuestion() to prevent moving forward when selectedAnswer is null Next Steps: - Fix scoring logic for when users navigate backwards during the quiz - When nagivating backwards through questions, retain previously selected answer
Simone Lue	Feb 17, 2025	0.5	Analyzed and summarized survey results to determing high/medium/low priority features for the application.

Simone Lue	Feb 17, 2025	1.25	Navigating backwards through questions, UI retains the previously selected answer - Used an array to track selected answers instead of a single selectedAnswer state - Updated handleAnswerSelection to store answers in the new array (which the index corresponds to the question index) - Set the selected answer when navigating to the previous question - Modify handlePreviousQuestion to retrieve and display the stored selection. Problem: - Navigating forwards AFTER navigating backwards retains the selected answer previously selected Solution: - When moving forward after going backwards, clear future selections to ensure past choices aren't remembered. Fix scoring logic for when users navigate backwards during the quiz. The current scoring logic does not automatically adjust the previous selection's score when changing an answer. Instead, it only adds points when selecting an option but does not remove points when an answer is changed - Modified handleAnswerSelection to first try to subtract points from the answers that are being changed Problem: - This method of scoring retained the scores from the questions that remained unchanged and added on an additional point for the new selection (whether from the same category or not) Solution: - Modify handleAnswerSelection to build scores from the remaining selections instead of subtracting points one by one - Modify handlePreviousQuestion and handleNextQuestion to reset scores of future selections when moving forward after going back.
Everyone	Feb 17, 2025	1.25	Meeting Notes Summary (February 17, 2025) 1. Upcoming 1-Week Tasks Focus on the next two MVPs: Personalized Recommendations (Google Places API integration for recommendations) Interactive Map (Google Maps API integration for visualization) 2. Past Week Progress Updates Team members shared knowledge and updates on completed tasks. 3. Heroku Backend Server Documentation Discussion on CRUD operations for backend endpoints 4. Connecting Simulator to Heroku Server Setting up the React Native simulator to interact with the backend hosted on Heroku. 5. Planning for Video Workflow Outlining the video workflow for the mid-term report. Deciding on tools and steps for video creation. 6. Firebase Realtime Database & App Distribution Revisiting Firebase Realtime Database setup. Setting up Firebase App Distribution for testing. 7. Google Places API on /search Path Integrating Google Places API for search functionality. Ensuring that the API can return filtered results based on user preferences.

Simone Lue	Feb 20, 2025	2	Implementing personalized recommendations feature - Built a FastAPI route (/places/recommendations) to fetch places based on user preferences. - Integrated Google Places API to get places based on user travel style (quiz results). - Stored cached places in PostgreSQL to reduce API calls. Problem 1: 422 Error (Unprocessable Entity) - FastAPI expected user_id as an integer, but it was sent as a string. Solution: - Ensured user_id is an integer Problem 2: 404 Error - FastAPI couldn't find user_id in quiz_results, even though it existed in PostgreSQL - The backend was using the local database instead of Heroku. - FastAPI was querying the wrong database (local waypoint_db instead of Heroku). Solution: - Connected FastAPI to Heroku Postgres by updating .env database_url Problem 3: API Route Conflict - FastAPI treated /recommendations as {place_id}, causing a 400 Bad Request. - The dynamic route @place_router.get("/{place_id}") was above /recommendations, so FastAPI assumed "recommendations" was a place_id. Solution: - Reordered routes in place_routes.py !!Confirmed API is working!! - FastAPI backend is using Heroku Postgres instead of local PostgreSQL. - Recommendations API (/places/recommendations) now works as expected.
Simone Lue	Feb 20, 2025	2	Built Recommended Places screen and styling Connecting frontend UI to make API requests to display the recommendations: - Developed methods to fetch recommendations - Developed filter options - Switched back to local to work on the app Problem: app UI was not displaying results - The frontend was now receiving an empty array upon making requests Solution: - Debugging logs indicated that google places was denying the requests due to invalid api key - Realized Heroku was configured with google places api but not locally - Added API key in .env and was able to retrieve the recommendations which reflected in the UI Next Steps: - Images for the recommended places are not found- need to fix - Refine filters - Make sure recommendations work for all travel styles
Simone Lue	Feb 21, 2025	2	Fixing the photo display on the recommendations screen - Logs show that the image can not be found - At first, thought it was the API key not being used in the frontend so it was restricting access to images, but upon correcting that, images were still not loading - Adjusted the method for image rendering a couple times and one finally worked (not too sure why) - Images now loading with the recommendations

Simone Lue	Feb 21, 2025	1	Making sure recommendations work for all travel styles: - Matching the different travel styles correctly to TRAVEL_STYLE_MAPPING - Had to rename the some of the keys to match the recognized travel styles Adjusting the displayed emoji for each travel style - Tried to do multiple emojis for combined travel styles but it offset the styling - Alternative: Selected one emoji best fit for different combined styles instead - Fixed the missing emoji for “No travel style” - Fixed the awkward wording for some of the travel styles such as "You are a You didn't align with any specific travel style Traveler" - Modify the return inside QuizScreen.js so that the formatting dynamically adjusts Next Steps: - Recommendations displaying for mixed travel styles - Recommendations fetch new results after travel style changes - Make sure the filters work
Russell Han Josef	Feb 22, 2025	2	Fixes & Improvements in SettingsScreen.js and Backend Initial Issues & Fixes Login Issues (422 Unprocessable Content) Issue: FastAPI rejected login requests due to incorrect request body formatting. Fix: Ensured email and password were correctly passed in the axios.post request in LoginScreen.js. User Data Not Persisting After Login Issue: Logged-in user details were not being saved for profile and settings. Fix: Stored user data in AsyncStorage after a successful login. Navigating to Home Screen After Login (REPLACE Error) Issue: navigation.replace('HomeScreen') failed due to missing screen. Fix: Updated App.js to correctly route users to Main after login. Backend Issues & Fixes Travel Style Not Saving (422 Unprocessable Content) Issue: FastAPI expected user_id in the request body for PUT requests. Fix: Created QuizResultUpdate schema to accept only travel_style. Modified PUT /quiz_results/user/{user_id} to update travel style correctly. Fetching Travel Style for User Settings Issue: Travel style was not being retrieved from PostgreSQL. Fix: Created GET /quiz_results/user/{user_id} to fetch travel style. Updated SettingsScreen.js to call this API and store the result in AsyncStorage. Frontend UI Issues & Fixes
Russell Han Josef	Feb 22, 2025	1	Drafted video recording flow.
Russell Han Josef	Feb 22, 2025	0.5	Fixed QuizScreen and LoginScreen to ensure first time user is able to take the quiz and save it correctly.

Russell Han Josef	Feb 23, 2025	3	Google Maps Integration on iOS Problem: Needed to integrate Google Maps SDK for iOS. Solution: Installed and configured Google Maps SDK (v7.0.0). Problem: API key was stored in Info.plist, causing security issues. Solution: Moved API key to .env and dynamically loaded it in AppDelegate.swift. Problem: "Tried to register two views with the same name AIRMap" error. Solution: Ensured only one instance of react-native-maps to fix duplicate registration. Problem: API key needed to be dynamically passed to Swift. Solution: Updated Podfile to load .env variables and inject API key into the build. Problem: Map was not displaying if the API key was missing. Solution: Added error handling to InteractiveMapScreen.js to show a message when the API key is missing. Problem: Needed to verify Google Maps displayed properly. Solution: Successfully displayed Google Maps centered on Vancouver, BC in the iOS simulator. Next Steps Implement user location tracking. Add custom markers for points of interest. Optimize map rendering and interactions. Test on a real iOS device for stability.
Russell Han Josef	Feb 23, 2025	2	Added markers with Google Places API.
Simone Lue	Feb 23, 2025	4	Midterm Report: Title page (complete), Introduction (complete), summary of initial proposed project (complete), changes to the proposal (complete), UPDATED project planning and timeline (IN PROGRESS: individual responsibilities; COMPLETED: Gantt chart, new proposed timelines, milestones, deiverables), Implemented Feature (IN PROGRESS: Login, Recommendations, Interactive Maps; COMPLETED: Quiz), Work Logs (IN PROGRESS), Closing and References (IN PROGRESS) Proposal changes: - Gantt Chart - Responbilities - Timelines - MVPs Priority
Russell Han Josef	Feb 24, 2025	1	Check on Firebase app Distribution. Upon checking, need to enrol in Apple Developer Program. Completed: Register for enrolment under educational institution. Request to waive the fee. What's next: Waiting for reply.
Everyone	Feb 24, 2025	1.75	Video Recording for Mid Term Report Checkpoint
Everyone	Feb 24, 2025	2.5	Finishing and finalizing midterm report. Preparing documents for submission. Push log to the github.
Simone Lue	Feb 26, 2025	2	Fixing filters Problem 1: Clicking a filter does not change the UI - Passing ‘places’ instead of ‘filteredPlaces’ in the FlatList filtering logic Solution: Passed the correct name Problem 2: Clicking a filter does not show anything (blank page) - Added debugging logs to check - Debugging showed that theres a mismatch between frontend filters and backend categories/frontend filtering is based on unmapped categories - Tried remapping google places categories -> didn’t work Solution: - Realized the retrieved google places for the current location were not related to any of the filters. - Changed one of the retrieved places to map from the raw API response category to one of the filter names and confirmed that it works. - For now, will have random google places categories map to the current applicable filters due to default location being set for recommendations retrieval and none matching the filters (ex. Mapping “museum” -> “spa”) - This filtering is to be integrated with interactive maps and will be changed.
Everyone	Feb 26, 2025	0.25	Midterm Feedback by Priya

Simone Lue	Mar 03, 2025	1.5	Combined the Interactive maps screen and the Recommendations screen - Both components working as intended - Both filters working separately Problem: Recommendations don't change with the travel style filter change for the maps Next Steps: - Make sure the recommendations pull from the travel style selected - Have both filters working together
Simone Lue	Mar 03, 2025	3	Added OpenAI dependencies - Added to the backend for security reasons - Since calling from the backend, added chatbot_routes, and updated main.py to include the new router Problem: openai version and syntax incompatibility - New syntax with the latest openai version Solution: - Fixed the syntax to match the latest version of openai - cURL response confirms api is working Added Chatbot Screen and simple styling - Able to send request to backend and generate AI response Next steps: - Style the Chatbot - Chat messaging needs to be shown in order - Loading visual to indicate the answer is being generated - Make refinements to provide tailored information
Russell Han Josef	Mar 04, 2025	3	Interactive Recommendations Screen Ensured the filter buttons update dynamically based on the selected travel style. Adjusted the zoom in, zoom out, and fullscreen buttons, positioning them correctly in the bottom-right corner of the map (35% of the screen). Refactored the category filter logic to dynamically pull unique categories based on available places. Places Router Updates Updated the API request to fetch places dynamically based on the selected travel style. Extracted unique categories from the API response to dynamically populate the category filter. Ensured error handling for cases where a travel style is missing or places cannot be retrieved. Next Steps Fix the extra white space issue in the scrollable category filter ➡ Investigate margin/padding issues in filterScrollContainer and listContainer Implement API caching for places data ➡ Cache the API response to prevent unnecessary requests when the same location and travel style are selected again

Russell Han Josef	Mar 04, 2025	2	Updated user_routes.py Removed quiz_results dependency. Now stores travel_style_id directly in the users table. Defaults travel_style_id = 4 (Undefined) on signup. Allows updating travel_style_id after quiz completion. Created Stack Navigator (AppNavigator.js) Wrapped BottomNavigation inside a Stack.Navigator. Added QuizScreen as a separate screen for navigation. Ensures navigation.navigate('QuizScreen') works from HomeScreen. Updated App.js Now uses AppNavigator instead of BottomNavigation. Ensures proper screen navigation for non-tab screens. Updated HomeScreen.js Fetches travel_style_id from AsyncStorage & Firebase. Shows "Take Quiz" banner if travel_style_id = 4. Logs user activity in Firebase when viewing the home screen. Updated LoginScreen.js & SignupScreen.js Integrated Firebase Realtime Database. Stores user details in Firebase on signup. Retrieves and syncs user data on login. 🛠 Next Steps: Update QuizScreen.js Ensure quiz tracking and progress-saving work correctly. Properly update travel_style_id in Firebase & Backend.
Russell Han Josef	Mar 04, 2025	2	Update QuizScreen Added Firebase to track Quiz Progression
Everyone	Mar 05, 2025	1	Updated Buttons for Zoom and Fullscreen Updated icon for list to load from Places API fixed the list spacing (scrollFilterContainer)
Simone Lue	Mar 05, 2025	5.5	Focus: UI Components Fixing the Bottom Navigation Icons: - Using Ion Icons - Previously tried to use depreciated version of IonIcons and had trouble displaying the icon properly - Installed latest version of react native vector icons and re-installed pods Problem 1: - Having the same issue displaying icons: square with '?' In the middle Alternative Solution: - Switching to FontAwesome6 for icons - Following readme for fontawesome6 to carefully install packages Problem 2: runtime error, vector icons module not found - Lots of errors with dependencies - Square icon with '?' - Tried to try with fontawesome5 - Nothing would work - Broke the node_modules package Solution: - Had to revert to original code Redoing the TabBar - Elevated look to the TabBar Problem: - Icons were getting cut off - Icons and label were shifted upwards and not centered Solution: - Played with padding a lot and finally got it centered

Simone Lue	Mar 05, 2025	4.5	Focus: UI Components Adding “More” menu - Small popup animated window when “More” tab is clicked - Used Animation for the window - Unable to navigate yet Blurring the background when the “more” menu is toggled on - Using react-native-community/blur Problem: - After careful and correct installation of dependencies and pods, it showed that the component <BlurView> was not working Solution: - Manually add blur library to Xcode and rebuilt app Adjusting the “More” menu - Able to toggle blur off and close popup when tapping outside the menu popup - Moved Settings and Profile to the popup navigation - Added InteractiveRecommendations and Chatbot to Bottom Tab Bar Problem 1: Tried to make the tab bar collapsable; screen background became empty Solution: - Using props to pass the hideTabBar and showTabBar methods to every screen Problem 2: Not every screen is scrollable Solution: - Using timeout option instead - Bottom tab times out after 5 seconds of inactivity to hide the screen Problem 3: InteractiveRecommendations flat list does not scroll behind the bottom tab bar Problem 4: Project build
Simone Lue	Mar 06, 2025	1	Retrieved lost progress: Toggling blur, adjusting popup menu, timeout option
Russell Han Josef	Mar 06, 2025	2	Fix device node modules issue. Unable to launch iOS device simulator. Checked the existing Bundler, Ruby, and CocoaPods versions. Verified the project's Gemfile.lock for required dependencies. Removed old Bundler 1.17.2 references from .bundle and vendor/bundle. Updated Bundler to the latest version using gem update bundler. Ran rbenv rehash to ensure the system recognized the new Bundler version. Installed missing gems bigdecimal and logger, which were removed from Ruby 3.4.2 defaults. Removed and reinstalled CocoaPods using gem uninstall cocoapods and gem install cocoapods -v 1.15.2. Updated the project dependencies using bundle install and bundle update --bundler. Verified the installed CocoaPods version with pod --version. Ran bundle exec pod install in the ios directory to set up iOS dependencies. Started Metro Bundler without issues using npx react-native start --reset-cache. Successfully ran the app using npx react-native run-ios.
Russell Han Josef	Mar 06, 2025	3.5	◇ Backend Changes (FastAPI & PostgreSQL) Migrated user_id to UUID Format Replaced Integer user IDs with UUID for security, scalability, and consistency across PostgreSQL, Firebase, and AsyncStorage. Added Email Availability Check Before Signup Implemented /users/check_email/ API to prevent duplicate signups. Integrated email validation in SignupScreen.js. Created itinerary_models.py for Collaborative Planning Consolidated Itinerary, ItineraryDay, ItineraryMember, and Activity models. Ensured UUID support across all itinerary models. Added API Endpoint to Fetch Travel Style Details Created /travel_styles/{travel_style_id} API to return travel style name & description. Used in ProfileScreen.js to enhance UX.

Russell Han Josef	Mar 06, 2025		◇ Frontend Changes (React Native) Ensured user_id is Always Stored as UUID String Updated SignupScreen.js, LoginScreen.js, and HomeScreen.js to store user_id as a UUID string in AsyncStorage. Implemented Travel Style Update Across All Data Sources SettingsScreen.js now updates travel_style_id in AsyncStorage, PostgreSQL, and Firebase for real-time sync. Fixed Dropdown Issue in SettingsScreen.js Replaced onChangeValue with onSelectItem to prevent alerts from triggering when expanding the dropdown. Updated HomeScreen.js to Use Firebase Instead of PostgreSQL Fetches travel_style_id from Firebase + AsyncStorage instead of making API calls, improving performance. Enhanced ProfileScreen.js to Display Travel Style Name & Description Uses travel_style_id from AsyncStorage to fetch travel style details from PostgreSQL. Shows meaningful travel style descriptions instead of just an ID.
Russell Han Josef	Mar 06, 2025	1.5	Fix Homescreen and added some details. More descriptions to be added....
Simone Lue	Mar 07, 2025	1.5	Focus: Chatbot Making the chatbot and user messages send in a normal texting format - Most recent messages on the bottom instead of on top of older messages - Order of the messages are preserved inside setMessages() without overwriting previous updates - Latest messages always appearing at the bottom by using ‘inverted’ on flatlist Next Steps: - Style the Chatbot - Loading visual to indicate the answer is being generated - Make refinements to provide tailored information
Russell Han Josef	Mar 07, 2025	1.5	Itinerary Updated Router Added Itinerary List Screen.
Russell Han Josef	Mar 07, 2025	2	Itinerary: Added Itinerary Details Screen. Added Swipeable feature to delete.
Russell Han Josef	Mar 07, 2025	1.5	Itinerary: Added new feature to re order the day. Using npm install react-native-draggable-flatlist Due to this feature, I have to update the model, schema, and router. Add "Add Day" button at the bottom of Swipable Flat List
Russell Han Josef	Mar 07, 2025	1.5	Itinerary: Added ItineraryDayScreen to show all the activities within a day. Added Function for time sorting. Using Model to add itinerary. Implement useFocusEffect after navigating back from ItineraryDay to ItineraryList to refresh the activities

Simone Lue	Mar 07, 2025	2.5	Configuring the AI to provide tailored information - Added content body to the role of the system (AI) - Ai is defined as a travel assistant with a default location in Vancouver BC - Chatbot screen retrieves user's travel style and tailors recommendations to the user's style Problem: Had trouble sending the travel style to the backend Solution: - Retrieved user travel style from from async storage and stored in state variable - Passed state variable to the backend Next Steps: - Style the Chatbot - Loading visual to indicate the answer is being generated - (If time permits) Tailor responses with weather information
Simone Lue	Mar 09, 2025	2.5	Added Chatbot and user avatar - Indicate Chatbot/user message - Styled and positioned avatars in desired position - Adjusted styling for message responses Styled the text input box - Changed the styling to a more stylish look - Send icon instead of text button Added a customer Header for the Chatbot Added animated "…" to indicate to users when chatbot is fetching a response - useState isTyping to track when the bot is processing a response - Before sending in a request, insert "…" as a placeholder message from the bot - "Animate" the "…" by cycling through "." ".." "…" every 500ms while isTyping is true - Remove "…" when the response arrives and replaces with the real message Tried to retain bottom navigation on profile, settings, and chatbot screens Problem: - Navigation is separate from Tab Navigator - Tried to use separate AppNavigator - Tried to use hidden Tab Screens - Could not resolve; will revisit Next Steps: - (If time permits) Tailor responses with weather information
Simone Lue	Mar 09, 2025	1.5	Problems: - No consistency in navigating between screens (some using Tab Navigation, some using Stack Navigation) - Back swipe is not working on Tab Screens - Bottom Navigation Bar is not showing up on screens in the "More" menu Solution: - Discarded Tab Navigator method - Strictly using Navigation and routes as custom bottom navigation - Changed main stack and app navigator stacks Results: - Consistent navigation - Back swipe working on all screens - Bottom navigation shows up on all screens unless specified - Flat list in InteractiveRecommendations is scrolling due to these changes

Simone Lue	Mar 09, 2025	1.5	Focus: Navigation No active state on the Bottom Navigation - Adding detection for active screen Problem 1: Incorrect Active Screen Detection - activeRouteName.name === item.name; was incorrect because activeRouteName was already a string, so .name was invalid. - This caused isActive to always be false, meaning no tab was ever marked as active. Solution 1: - Changed const isActive = activeRouteName === item.name; to directly compare the string values Problem 2: Navigation Structure Issue - By using navigation.navigate("Main", { screen: item.name }), the active screen was inside a nested stack. - route.name only gave the top-level screen, so it wasn't detecting the focused tab correctly. Solution 2: - Used navigation.getState() to extract the focused screen inside the "Main" stack Results: - Active state added - Activate state shows label and changes background colour
Simone Lue	Mar 09, 2025	1	Focus: Navigation Ensure HomeScreen is active on initial load - Default activeRouteName to "Home" on first load - Ensure "Home" is highlighted on first load before navigation updates. - After navigation changes, activeRouteName updates normally. Problem 1: - The first useEffect runs before navigation state is fully loaded, so "Home" is not recognized as active. - Setting active state inside useEffect([navigation, route]) does not immediately reflect the correct screen. Solution: - Use useFocusEffect to detect the correct active screen on initial load. Problem 2: - Home is recognized as active on initial load but navigating to other screens do not update the active state Solution: - use setTimeout() to delay updating active state until navigation completes because navigation and state updates are happening out of sync
Simone Lue	Mar 10, 2025	2	The filterScrollContainer disappeared once the bottom nav got added back to the screen Problem: - Through debugging, found that the Flatlist in InteractiveRecommendations got pushed up and covered the filters Solution: - Reduced Flatlist sizing - reduced minimum height of the scroll container so that it stops scrolling vertically Changed the splash screen title to WayPoint Tried adding splash image logo - Adding and editing in xcode Storyboards Problem: - Image will not load - Image was in the wrong place Solution: - Moved image to images.xcassets INSIDE xcode.workspace

Simone Lue	Mar 10, 2025	0.5	Limiting Chatbot response - OpenAI generates too many recommendations and suggestions: took too much time to process and text is too long - Limiting chatbot recommendations to 3 suggestions - Added additional prompt to the full system prompt that is sent to openai Added an initial load in message from chatbot - Use setTimeout() inside useEffect() - Introduction message displays after 1 second
Russell Han Josef	Mar 10, 2025	1	
Russell Han Josef	Mar 10, 2025	0.6	Simone shared about the Navigation progress (done) and Chatbot. Russell shared about 50% progression of the Itinerary.
Russell Han Josef	Mar 10, 2025	1	Done
Simone Lue	Mar 10, 2025	3.5	Researching - OpenWeatherMap API calls: Need geographical coordinates (lat, lon) - Direct Geocoding API to get geographical coordinates (lat, lon)- Requires city name, state code, and country code for accurate results of coordinates - react-native-geolocation-service library - Need to request phone's geolocation feature Location Permissions - Creating a screen to allow user to enable location services - Created styling for the LocationPermissions screen - Modify Podfile to set up script for permissions - Add permissions to Podfile Problem: - Podfile was not recognizing RNPermissions/LocationWhenInUse - Throw an error when trying to access permissions Solution: - Modified podfiles and removed RNPermissions/LocationWhenInUse and RNPermissions/LocationAlways to only use RNPermissions - The core RNPermissions pod dynamically manages location permissions Next Steps: - Use retrieved latitude and longitude coordinates to call OpenWeatherMap API
Simone Lue	Mar 12, 2025	1	Using geographical coordinates retrieved to access OpenWeatherMap API - Changed simulator's location to New West. - Want: - weather.icon - weather.main - main.temp : Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit - Verified API call works - Created weather_routes.py and linked to main.py Next Steps: - Complete weather router - Fetch coordinates for every session and pass to router - Retrieve openweathermaps api data - Show details on screen to confirm
Simone Lue	Mar 13, 2025	1	OpenWeatherMap API - Completed weather_routes to retrieve temperature, icon, and weather condition - Set up method in the frontend to pass location coordinates to the backend - Showing the retrieved information on LocationPermissionsScreen (temporarily) to confirm working API call and to confirm the passing for information from frontend to backend Restructuring LocationPermissions screen - Passing location permissions to homescreen - if location permissions are granted, fetch location to pass to homescreen

Simone Lue	Mar 13, 2025	1.25	Restructuring HomeScreen - Accept getLocation() as a prop to fetch location - Displaying weather information on homscreen instead of in LocationPermissions Problem 1: - Rendering LocationPermissions screen behind the Homescreen - Both screens rendering simultaneously Solution: - Instead of rendering as a child inside HomeScreen's main view, use conditional full-screen rendering Problem 2: Weather information not displaying - axios and API_BASE_URL imports missing from HomeScreen Solution: - Added necessary imports Problem 3: LocationPermissions screen loads up on subsequent app launches even when permissions have been granted - permissions start off as false and only sets to true when user presses "Allow" on the LocationPermissions screen. - The component doesn't know permissions were granted in a previous session until button has been pressed Solution: - Added useEffect to check permission status of the device on component mount Next Steps: - LocationPermissions screen won't go away if permissions not granted
Russell Han Josef	Mar 13, 2025	3	Backend: Fixed POST /itineraries/ by ensuring created_by matches an existing user ID. Implemented GET /itineraries/{itinerary_id} to return full itinerary details. Added POST /itineraries/{itinerary_id}/days/ to allow adding days to an itinerary. Added POST /itineraries/{itinerary_id}/days/{day_id}/activities/ to support adding activities to a day. Implemented DELETE /itineraries/{itinerary_id} to allow itinerary deletion. Implemented DELETE /itineraries/{itinerary_id}/days/{day_id} to allow itinerary day deletion. Implemented PUT /itineraries/{itinerary_id} to allow editing itinerary details. Frontend: Implemented Itinerary List Screen to fetch itineraries from PostgreSQL and display them. Improved Itinerary List Item UI for better readability. Implemented Itinerary Form Screen to allow creating and editing itineraries. Integrated react-native-calendars for selecting start and end dates in the itinerary form. Ensured that upon itinerary creation, the user is navigated to ItineraryDetailScreen with the new itinerary ID. Implemented Itinerary Detail Screen to display itinerary details and handle deletion. Added Itinerary Day Screen to list all activities for a given itinerary day. Implemented react-native-date-picker for activity time selection. Converted activity list in ItineraryDayScreen to be swipeable, with a placeholder delete action. Fixed swipeable delete button height issue to ensure alignment with activity cards. Improved UI consistency across itinerary-related screens. Next steps: Implement actual deletion for activities in ItineraryDayScreen. Allow users to edit itinerary days and activities instead of just deleting them. Integrate Firebase Realtime Database for tracking changes in itineraries.

Simone Lue	Mar 13, 2025	2.5	Adjusting the weather display in home screen - Rounding temperature to display no decimals - Adjusted the layout of the weather box Adjusting the MoreMenu - Converted MoreMenu from a separate screen into a Modal Overlay - Modal overlay is toggled when the more button is pressed which displays over the current view Styling Modal menu - Fixing up the UI - Implementing BlurView to blur background when modal is toggled Problem: - BlurView is pushing my modal menu out of position Solution: - Tried to put modal menu inside BlurView -> did not work; blurred everything - Use absolute positioning to place modal menu on top of blurview Next Steps: - LocationPermissions screen won't go away if permissions not granted - Handle denied permissions - Format chatbot responses - start Events screen
Russell Han Josef	Mar 13, 2025	2	Added an Invite Collaborators button in the Itinerary Detail Screen Placed the button in the bottom fixed button container Designed the InviteCollaboratorsScreen with inline styling Created a search bar for filtering users Displayed a list of users with an Invite button Added a pending invites section to track invited users Used dummy data for now, preparing for Firebase integration Fixed Text strings must be inside a Text component warning Wrapped the trash icon inside a Text component Checked and ensured all text elements are properly inside Text components
Russell Han Josef	Mar 14, 2025	1.5	Updated InviteCollaboratorsScreen.js to store invitations under /invitations/invitee/{userId}. Tracked pending invites under /live_itineraries/{itineraryId}/pendingInvites/. Ensured invited users are displayed in InviteCollaboratorsScreen.js. Updated ItineraryListScreen.js to show pending invitations before shared itineraries. Fetched invitations from /invitations/invitee/{userId}. Added "Accept" and "Decline" buttons with placeholder alerts.
Simone Lue	Mar 14, 2025	0.5	Research EventBrite API Problem: Event search API is no longer available - Unable to search general events without specific event ID Alternative Solution: - Research for different events API - Found PredictHQ which allowed searching of local events - Set up token and tested api call on PredictHQ's website Next Steps: - Set up Events component (Screens, styles, routes)
Russell Han Josef	Mar 15, 2025	3	Fixed fetchItineraryDetails to update days properly Ensured the invite button only shows if the user is the itinerary owner Updated fetchSharedItineraries to fetch owner details from FastAPI Modified renderItineraryItem to display owner name and email Prevented users from inviting themselves in InviteCollaboratorsScreen Ensured adding a new day updates the UI correctly

Simone Lue	Mar 15, 2025	3.5	Chatbot Text Input not clearing - Added useRef to manually clear text input after sending message Set up Events Screen - Set up placeholders for how the events screen should look - Added temporary styling Set up Events router - Set events router and connected to main.py - Set GET method for the API - Set params for the API call Set up methods in Events Screen - Using Geolocation to retrieve device's coordinates - Passing coordinates to the router to use the PredictHQ API - Fetching response data Problem: API 401 Unauthorized error - Token refuses to work - Also discovered PredictHQ is a paid subscription after a free trial -> will not be able to use Decision to remove Events MVP following failure to search for a Events search API
Russell Han Josef	Mar 15, 2025	1.5	Improved UI/UX for Day: Implemented PUT endpoint for editing itinerary days Added left swipe action to reveal the Edit button on day cards Updated the modal to pre-fill day details for editing Improved date display format to "Sat, Mar 15" using locale options Handled timezone issues by parsing date strings into local Date objects Fixed button rendering logic to show "Remove" for collaborators and "Delete" for the owner Ensured isCollaborator updates correctly before rendering buttons Updated fetchItineraryDetails to properly check if the user is a collaborator Confirmed navigation back to ItineraryListScreen after removal Verified Firebase updates when a user removes themselves from an itinerary
Russell Han Josef	Mar 15, 2025	1	ItineraryDayScreen: Fix the delete activity height to match the card. ItineraryListScreen: Show Last Updated for better UX. Itinerary Models, Routes, Schema: Add last_updated_by column to the itineraries table
Russell Han Josef	Mar 15, 2025	2	Fix Activity Update API - Convert UUIDs to Strings in Response Encountered an issue where updating an activity was failing Checked the frontend code for issues in the request payload Verified that the correct X-User-Id header was being sent Ensured the itineraryId, dayId, and activityId were correctly passed Reviewed the FastAPI update activity route for possible issues Debugged the request and response to find inconsistencies Identified that UUID fields in the response were causing validation errors Prepared to modify the FastAPI response to return UUIDs as strings
Russell Han Josef	Mar 16, 2025	3	InviteCollaboratorsScreen: Added function for remove collaborators as the owner. Added a dedicated edit icon on the day card to decouple editing from the swipe gesture Integrated a collaborators list in the Overview tab using a Firebase listener Improved the Overview UI with a styled card and clear section for collaborators Ensured add day, edit day, delete day, and drag-and-drop reorder functionalities remain intact Refactored and organized code for clarity and maintainability