Dishcovery.

README File

Defne Genç | Janet Zhong | Amrita Palaparthi | Kyla Guru

Access our Hi-Fi prototype on Expo Go:

What is Dishcovery and When Would You Use It?

Dishcovery is a mobile app that helps users **identify**, **learn about**, and **cook with** foreign ingredients.

Using an image recognition API within the app's central **scan** function, it can identify ingredients that users cannot yet recognise. A user can leverage the

“scan” button in the navigation bar to scan an ingredient they would like to learn more about.

Upon scanning an ingredient, the user will be able to view information about the ingredient’s origins, its flavour profile, its cultural context, and its health benefits. They can also access recipes using that ingredient and request recipes should they find that they cannot see a recipe they would like to use.

Some additional use cases for the app are: searching for dishes or recipes, searching by cuisine, manually searching by the name of an ingredient, and saving recipes they like for later use.

We anticipate that individuals may leverage this app in the grocery store or farmer’s market when they encounter an ingredient that they are unfamiliar with in order to learn more about it. We also support the use case of exploring new foreign foods to cook, both through keyword and filter search as well as based on an ingredient that they have scanned.

Installation Instructions for Hi-Fi prototype

Access on the web:
https://expo.dev/accounts/amritapv/projects/cs47-dishcovery/updates/9104a345-05e2-496a-a329-aa5d5e1d4724

Access on iPhone or Android

Download the Expo Go app and scan the following QR code at the top of the page. You may need to use the following login credentials for Expo Go:

- Username: amritapv
- Password: correcthorsebatterystaple

Opening the source code

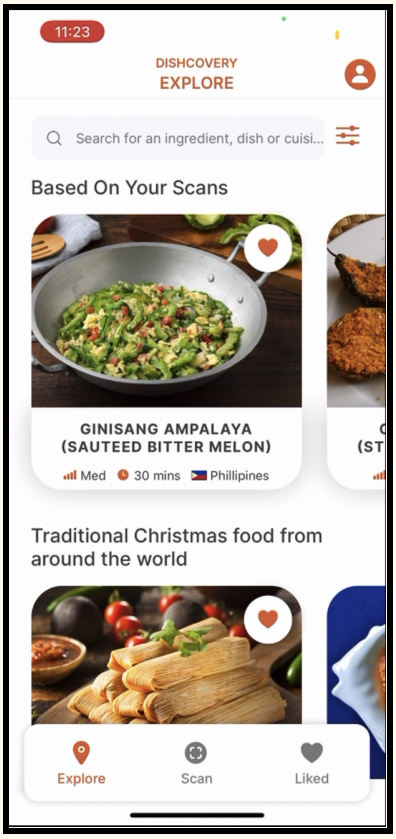
1. Download the code from Github
2. Download Expo and node and yarn
3. Open in VS Studio or your favourite text editor
4. Run yarn install or npm install
5. Run npx expo start and open either on Expo Go or iOS or android simulator

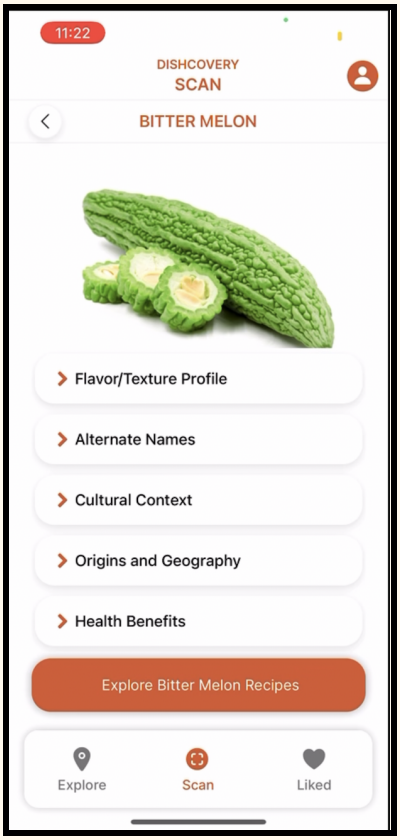
Operating Instructions / How to Use the App

The landing page for a user opening the app is the **Explore** page. Using the navigation bar, users can navigate to the **Scan** page or the **Liked** page containing their saved recipes.

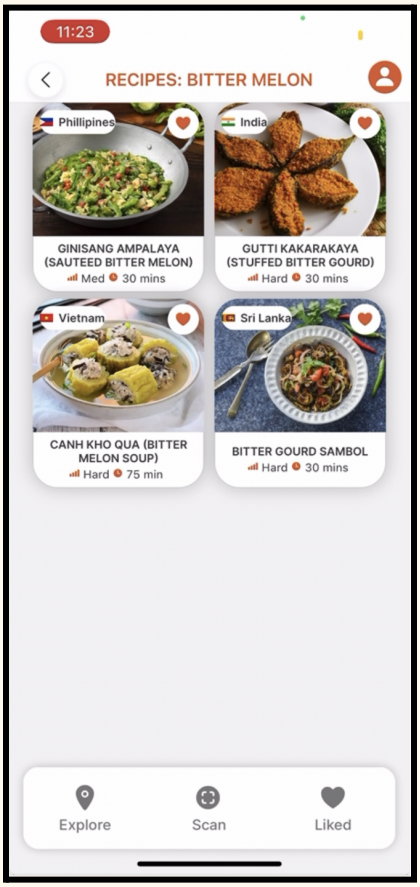
We proceed task-by-task to explain how a user might navigate through the app. For a user with an ingredient on hand that they would like to learn about, they would first want to navigate to the **Scan** tab.

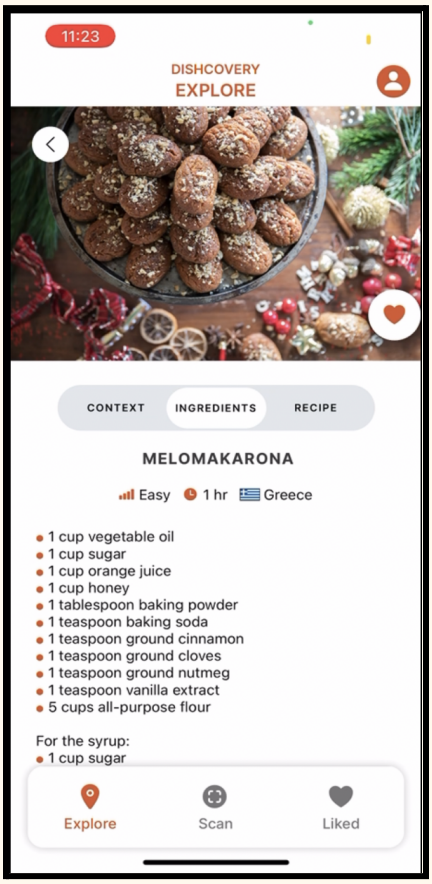
After that, they place their item within the frame and click to scan. The app will tell the user if the scan failed by turning its borders red, and if it succeeded by turning the borders green. Following a successful scan, the user can see more recipes using the ingredient via the **“Explore Bitter Melon Recipes”** button.

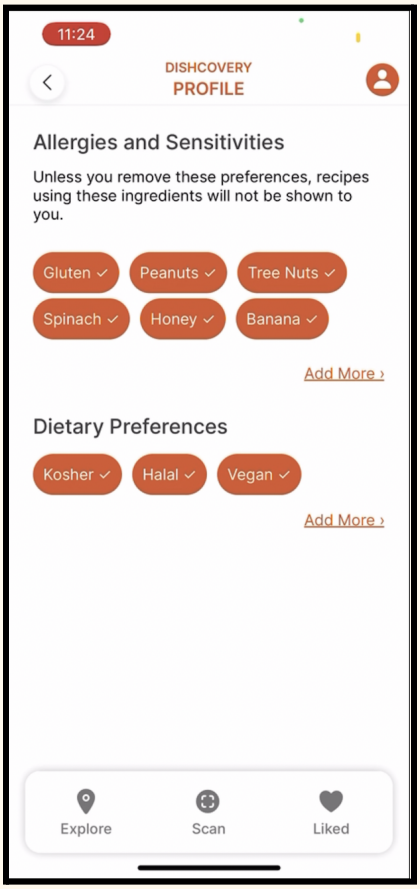




The leftmost screen above also shows where a user would be searching for an ingredient. The search bar can flexibly allow the user to find an ingredient, cuisine, or dish they are interested in, as well as save those recipes.







Above are some additional screens showing how a user can further explore an ingredient through its recipes, view an ingredient, and change their profile settings.

Limitations

Coding Limitations

The current limitations of our code are the hard-coded aspects as well as some incomplete segments. For example, our “Liked” page is not yet functional, which is something we need to implement further on. The same is true for the search pages, as well as the profile settings and filter settings. All of our context and

recipe data is currently hard-coded and we do not yet have a database. More of the coding limitations are covered in the Wizard of Oz and hard-code section.

Content Limitations

The authenticity of the recipes is really important, but this requires manual sourcing. When forming our database of recipes and ingredient contexts, we need to vet inauthentic or inaccurate information and can't just use any database. This is key to Dishcovery's values, so we would need extensive research to be able to source the right recipes.

Additionally, there is difficulty in allocating cuisines to recipes. Currently, we have one cuisine per recipe, which can be very incorrect. Greek dolmades can just as easily be Turkish dolma with very similar if not the same recipes, or one dish can have a multitude of variants across different countries (such as stuffed peppers in the Balkans).

We need to take into account colonial history, occupied territories, and reasons behind one dish being local to two different nations/regions. Cyprus, Palestine/Israel, the Kashmir region, Kurdistan, and the Balkans are only a few of the regions which require additional sensitivity when speaking of their recipes. Because of this, we may also need to allocate multiple countries to a dish or even just associate it with a region, listing its variants across that region.

Wizard of Oz/ Hard-coded features

The database of recipes is only 8-12 recipes and is hard-coded.

We have manually sourced this using google and chatGPT. Ideally there would be a database of recipes perhaps pulled from a Recipe API.

The cultural context of our selected items is currently hard-coded and stored locally on the app.

In practice, this could potentially be sourced by searching the internet and interviewing individuals who prepare traditional cultural recipes. Care must be taken to ensure this information is authentic, as it is a core value of Dishcovery.

Accuracy of image recognition is Wizard of Oz-ed

Our image recognition is fully functional via Clarifai API food image recognition. However, Clarifai gives many options and so for the purposes of the demo we have Wizard of Oz-ed the accuracy of the search results. Our image recognition only uses guesses that are in both the Clarifai API food-item library (around 500 generic ingredients) as well as our custom library of cultural food items. Currently we only have three items, bitter melon, cardamom and lemongrass. So if one scanned a cucumber, it would probably come up as bitter melon. This is useful for our demo but we would need extra screens to pick the right item from a range of Clarifai results if this were real.

The "liked" recipes are hardcoded.

The "liked" page is not functional at the moment due to time constraints but could be coded using Async Storage, Supabase or Firebase. For now, there are a few recipe cards hard-coded onto the "liked" page to display what this might look like if the user were to save recipes to view them later.

The search results, filter page and profile page are hard coded to search only for "bitter melon."

The search results, filter page and profile dietary requirements are not yet functional. A way to implement this is the Algolia search API, or a more rudimentary search using filtered arrays.

Tools Used

Software/ Design tools:

- We redesigned our entire app on **Figma** (link). See our revamped med-fi design.
- We wrote our code in **React Native** using **Expo.**
- We used **Github** (link to our Dishcovery repository) and **VS Studio Liveshare** to code collaboratively.

Coding tools or tutorials used:

- We used the **Clarifai food recognition API** (link) for our image recognition. We followed similar steps to this **Clarifai AI tutorial** (link), but modified it to do food recognition, and to fit Dishcovery's UI.

- For the recipe coding, we used some aspects of this **tutorial** (link) and this **github repository** (link). But most of it has been heavily modified to fit our design.
- We **manually sourced** our context for ingredients, context for recipes and recipes **using blogs or websites from Google** as well as generating them on **ChatGPT**. We then tried to manually verify for authenticity or accuracy, in case ChatGPT had any biases.