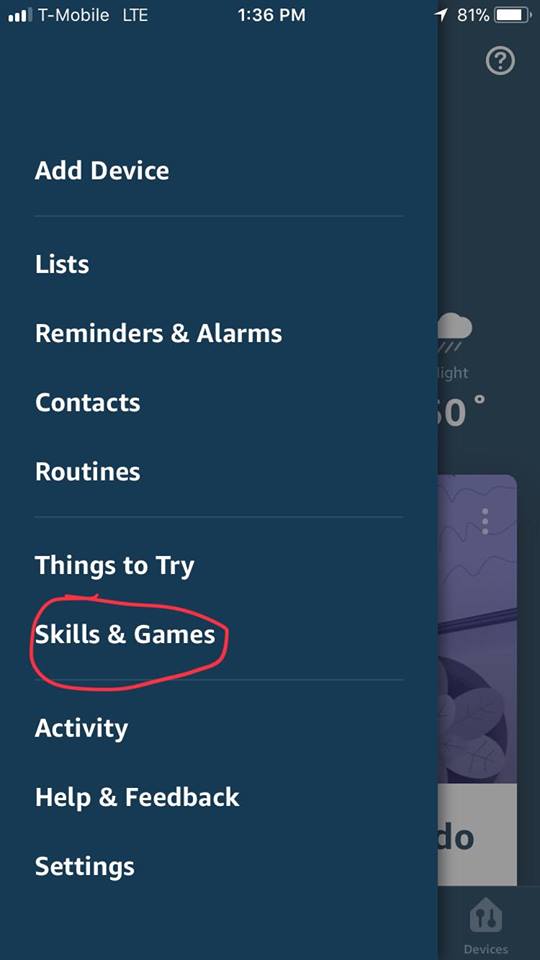
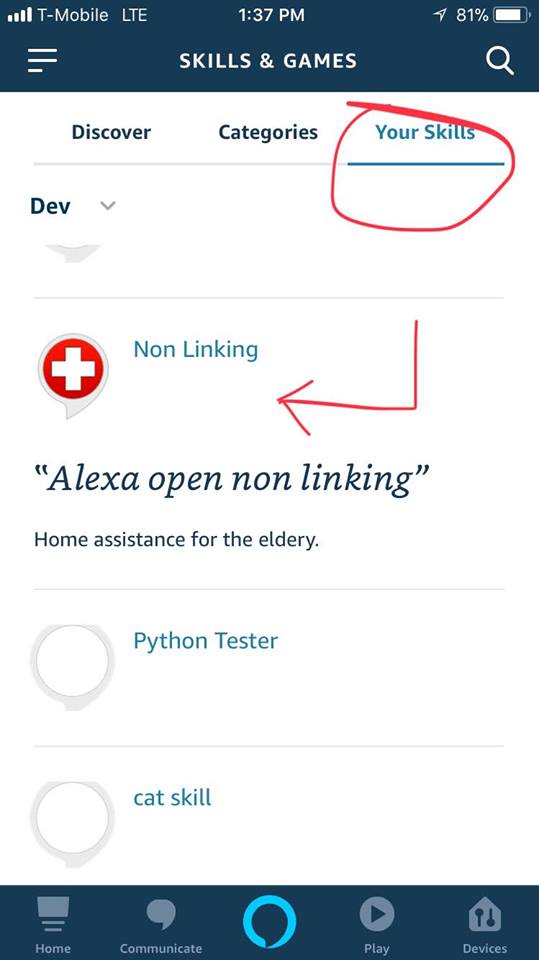
Auth0:

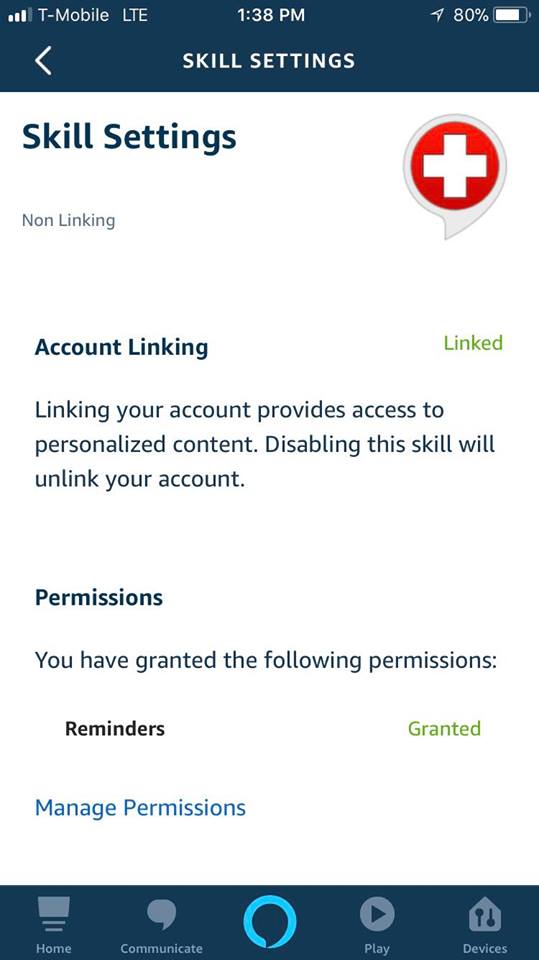
**Purpose:** To allow profile linking with Alexa. With profile linking, we’ll be able to access data, with permission, of linked users. With access to user’s data, the skill can use it for certain parameters such as inputting the data into a database.

**How to:** The first step to ensure the process of linking accounts work is connecting the skill to [Auth0](https://auth0.com/). Auth0 is a third party website that allows multiple log in authorization from many companies such as google or amazon. Follow the following tutorial from [Jovo](file:///C:\Users\Angel\Downloads\•%09https:\www.jovo.tech\tutorials\alexa-account-linking-auth0) to set up Auth0 into the Alexa skill. Once finished, users of the skill will need the alexa app. Follow the next following steps below (this is for the client account linking now)









From this page, an option to link an account will be available. Once done, the user will still logged on until they disable the skill from the app.

**Resources:**

* <https://www.jovo.tech/tutorials/alexa-account-linking-auth0>
* <https://auth0.com/>

SDKv1 Adaptor:

**Purpose:** Alexa development environment has evolved recently as the Software Development Kit has been upgraded to version 2. Version 2 contains new and improved implementations, but majority of the skill’s code was using the previous version.

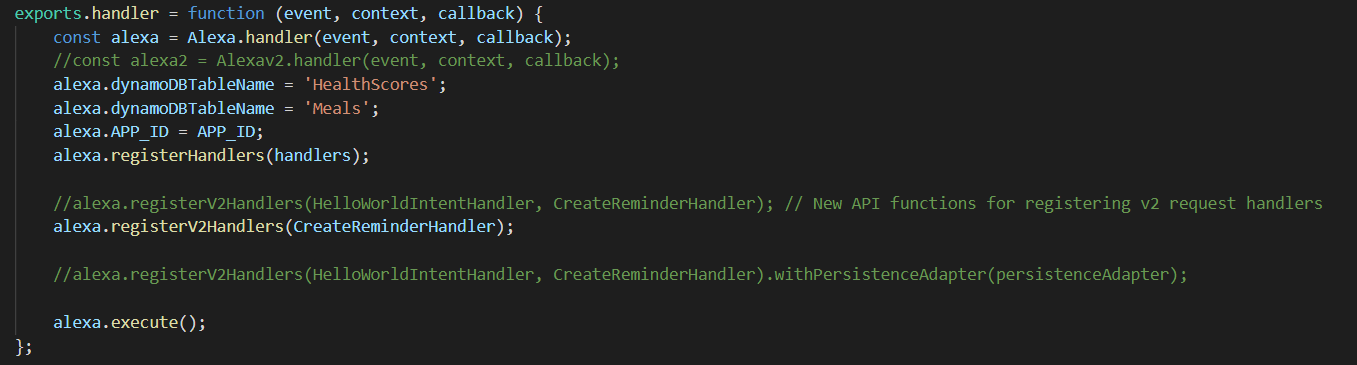
**How to:** Within the package of the code, there is a need to install this adaptor in order for both version 1 and 2 to work simultaneously. Use the following line of code to install the SDKv1 adaptor through npm:

npm install --save ask-sdk-v1adapter

Once installed the package will now contain the folder <insert name>. In order to use it add the following lines to the code:

const Alexa = require('ask-sdk-v1adapter');

Once implemented into the code, version 1 and 2 can work together. Here is how to execute the code as it shows how to separate intent functions.



The intent functions are responsible for all of the skills intents. The screenshot above demonstrates how to run version 1 and 2 intents by separating them based on their versions. Using alexa.regsterHandlers(handlers) will execute all version 1 functions as all of them are under handlers. To run version 2 code is alexa.registerV2Handlers(‘v2\_function\_name’, ‘other\_v2\_function’), which you place all the version 2 function names into the parenthesis.

**Resources:**

* <https://ask-sdk-for-nodejs.readthedocs.io/en/latest/ASK-SDK-Migration-Guide.html>

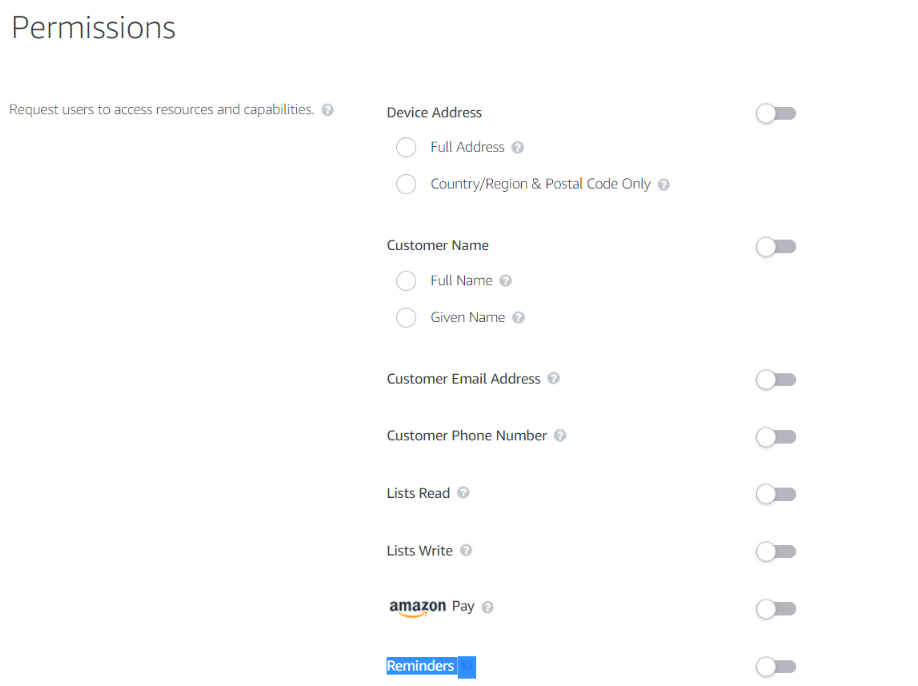
Reminders API:

**Purpose:** Alexa Reminders API allow patients to create reminders for their appointment.

**How to:** Creating the reminder has been done in the code. The function is running on SDKv2 and at the moment there seems to be no way to implement it on SDKv1. Although the code of creating reminder is partially done, it’s important to turn on the permissions to allow reminders.



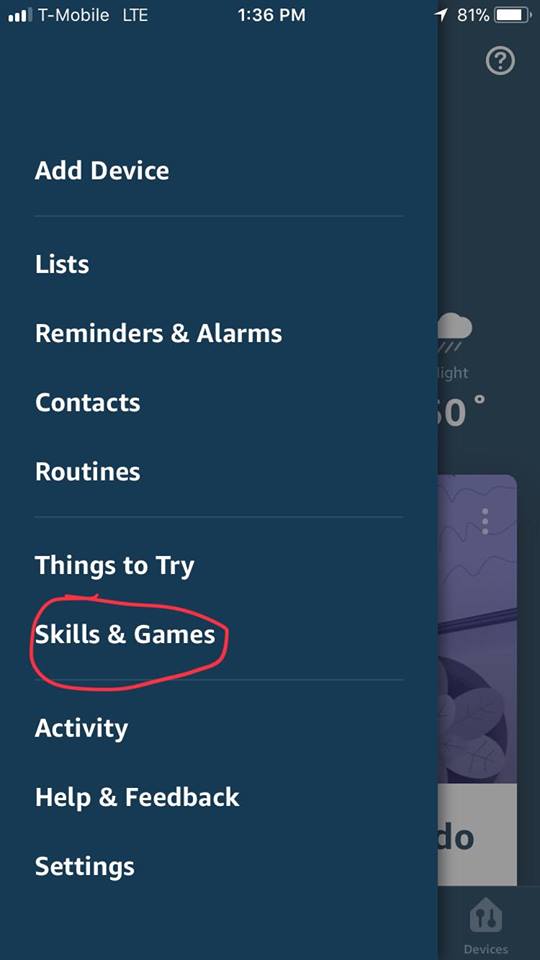
From the developer console of the skill, scroll all the way down to access the permissions page.

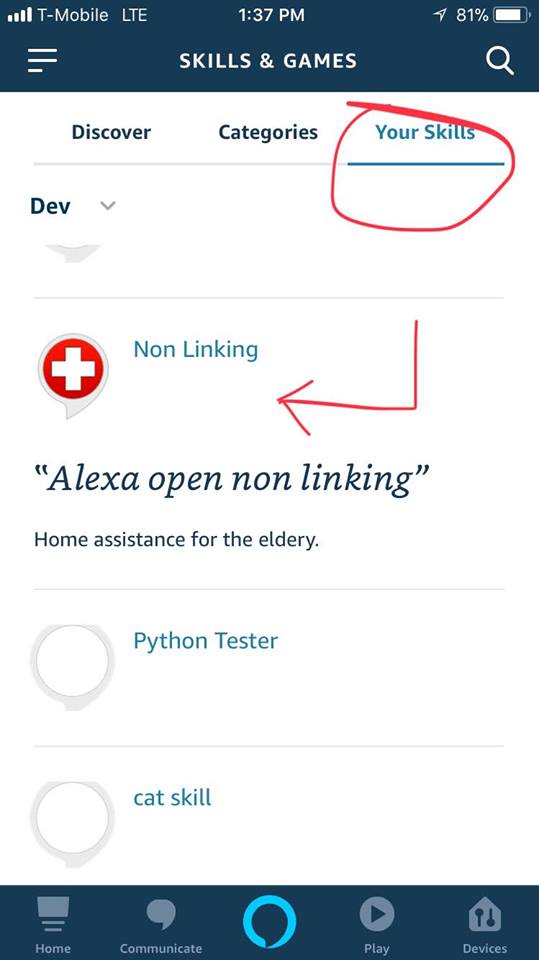


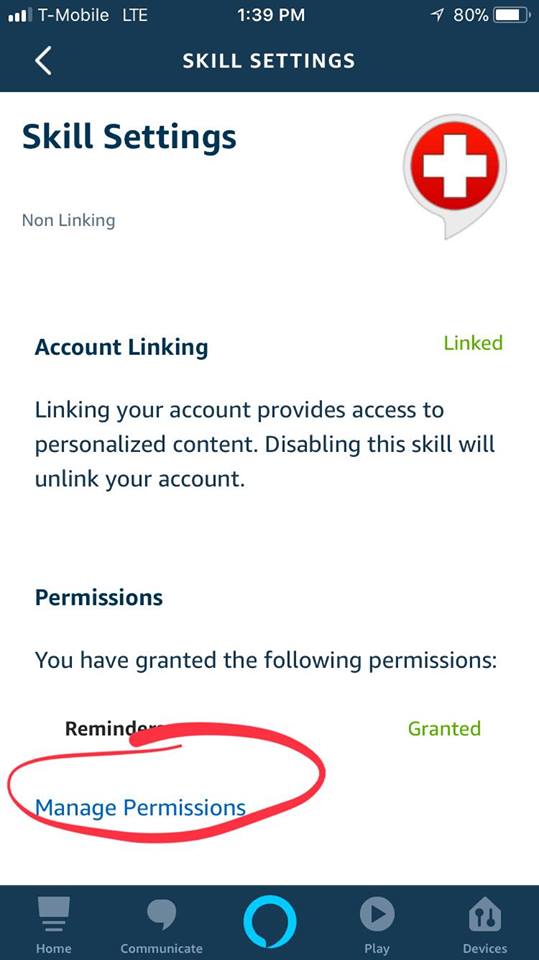
Turn on the reminders on this page and the development side is now complete.

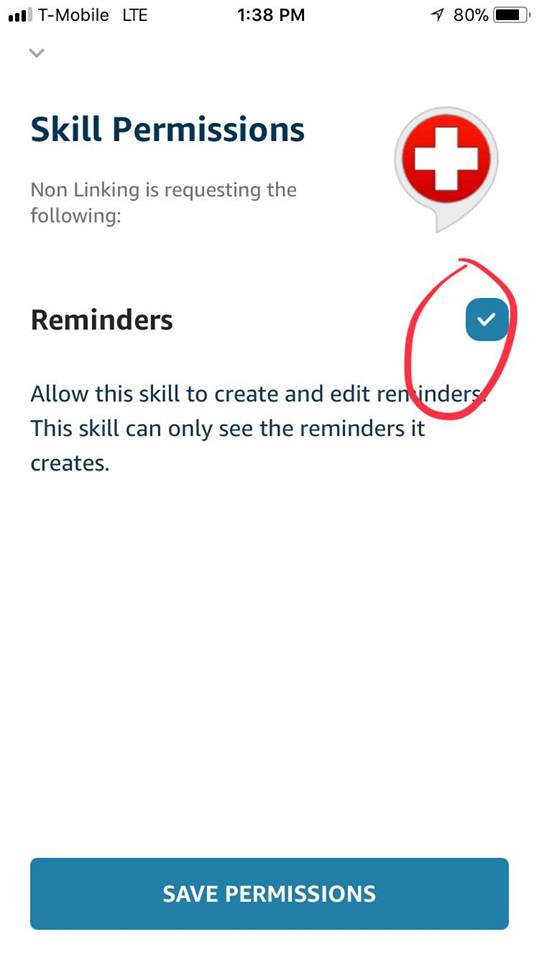
Now for the client side.











Follow the images steps above in order to turn on permission in the client side.

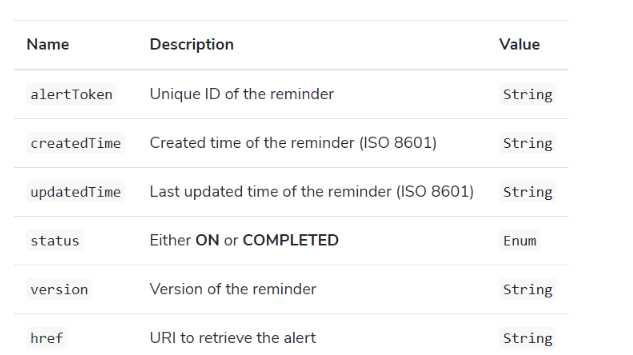
Other:

Alexa SDK can be done in either python or node.js and the given code in this package is done in node.js. Most of the lessons for how to code alexa skills in node.js was mostly from CodeCademy. Tutorials can also be found all over the web as Alexa becoming a poplar gadget these days. If you wish to get a good review for using node.js for alexa skills, then the link provided below will help.

(\*) <https://www.codecademy.com/learn/learn-alexa>

\*Lesson 4(Add Persistence to Your Skill (SDK v1)) is beneficial to the code and AWS side so it is recommended to know this. This is crucial in the AWS side as the dynamodb permissions need to be turned on.

For the reminders API, the purpose of that was to create reminders into the user’s app. The main goal for this function is to actually give patients the ability to make an actual appointment. One idea is when creating the reminder, grab the reminder token and place it onto a WDACS database. To grab a reminder token is to access it through the json response.



To grab the alertToken use the following line of code

reminderToken = JSON.stringify(reminderResponse.alertToken);

Once the token is obtain, place it onto a database as well as the time the user wanted the appointment on. The time and date can be passed onto the database by collecting it from the slot data it’s tied to in the lambda code. It’s important to pair the date and time with the token as we can use the date and time as a slot in another intent. This will allow us the ability to give users to update or delete their reminder. Since the token is within the database of WDAC, this will give them an idea what day and time the patient wants there appointment.

Use this page to get an idea how to delete and update a reminder

<https://www.jovo.tech/docs/amazon-alexa/reminders#delete-reminder>