

# Capstone Project Summary

My Capstone project builds on the previous course 4 Serverless project of to-do list, adding APIs, DynamoDB tables and UI elements to it so the to-do app now has the ability to collect and manage user profiles.

## Purpose:

The to-do app uses third party authentication from Auth0, one of the drawbacks from using third party authentication is our app does not have the profile of our users, such as name and email. Such information is valuable to any organization, thus I decide to expand our app to allow users to voluntarily create such profile, and to manage profile through update and delete operation.

I have been a back end developer for 20 years; I have not done a lot on the front end. To deepen my knowledge in front end, I also decided to enhance the UI arrangement of to-do list, adding the following capabilities:

- Ability to update due date of any to-do item
- Automatic sorting of all to-do items: unfinished items are listed before finished ones; among unfinished items, the one with earlier due date is listed first

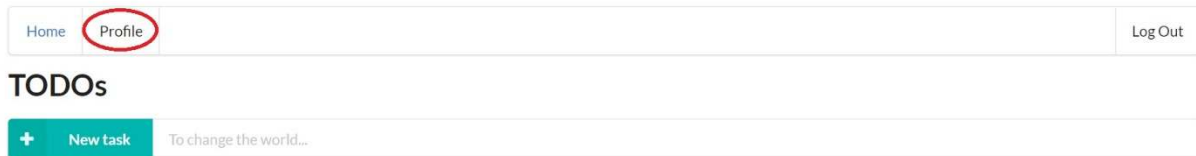
## System Modification:

- A new DynamoDB table \$PROFILE\_TABLE, to store user profile
  - For each user, there can be 0 or 1 record in this table
  - Partition key is userId, since that is all we need to query table, there is no need for a sort key
  - Table takes two other fields: userName and userEmail
- APIs to create and manipulate user profile
  - GetProfile, get method
  - CreateProfile, post method
  - UpdateProfile, patch method
  - DeleteProfile, delete method
  - APIs have individually-defined IAM permissions
- All resources, APIs and permissions are successfully defined and deployed via serverless template.

## UI Enhancements:

Based on existing to-do app UI, with the following enhancements:

Upon user login, a new menu item 'Profile' will appear:



The image shows a horizontal navigation bar with three items: 'Home', 'Profile', and 'Log Out'. The 'Profile' item is circled in red. Below the navigation bar is a section titled 'TODOs'. It contains a teal button with a white plus sign and the text 'New task', followed by a text input field with the placeholder text 'To change the world...'.

User can click on “Profile”, UI will go to the Profile screen. At this point UI calls GetProfile API to query user profile based on userId, if none is found, UI will display empty name and email field for input:



The image shows the 'Profile' screen. At the top is a navigation bar with 'Home', 'Profile', and 'Log Out'. Below the navigation bar is the title 'Profile'. Under the title are two text input fields, one labeled 'Name' and one labeled 'Email'. At the bottom of the form is a grey button labeled 'Create Profile'.

Once user fills the necessary information, clicks “Create Profile” button, UI calls CreateProfile API to insert an item into the DynamoDB table:

localhost:3000 says  
Profile is successfully created!

OK

Home

Profile

Log Out

## Profile

Name

Test Account

Email

test.account@gmail.com

Create Profile

Once profile is created, or if at the time of login user already has an existing profile, UI will display “Update Profile” screen:

Home

Profile

Log Out

## Profile

Name

Test Account

Email

test.account@gmail.com

Update Profile

Delete Profile

User can update name or email, in this same screen, user can also delete profile. UI will invoke corresponding APIs.

In the to-do list section, user can now update due date of any to-do item:

## TODOs

+

New task

To change the world...

- ☐ Test3

02/23/2021
- ☐ sort test 2



Once updated, all to-do items will be sorted based on if they are completed, and based on their due date:

## TODOs

+

New task

To change the world...

- ☐ sort test 2

03/02/2021
- ☐ Test3

03/05/2021

Checking the “done” checkbox of any to-do item will also trigger a re-sort:

## TODOs

+

New task

To change the world...

- ☐ Test3

03/05/2021
- ☒ sort test 2

03/02/2021

So does the “New task” function, any new task added will be inserted into the appropriate position in the displayed to-do list, based on its due date.

For testing, download and build the client/ portion from the GitHub repository, all above function should work.

The work added for capstone should meet all course 4 rubric requirements, except the one for DynamoDB composite index, as that is not necessary for the new table. Therefore, it satisfies at least three rubrics as required by the capstone project.