functionality :
-find skills based on default associated emotion
-find skills based on user impact records

track for each use :
which emotion was identified
which skill was chosen
the impact

TABLES

users : collected

- user_id : PRIMARY KEY SERIAL,
-name : VARCHAR,
-email : VARCHAR,

skills : base info

-skill_id : PRIMARY KEY SERIAL,
-skill_title: VARCHAR,
-skill_details:VARCHAR,
-icon: VARCHAR (image)

records : collected

-record_id : PRIMARY KEY SERIAL,
-user_id : users.user_id,
-emotion_id : emotions.emotion_id,
-date : date,
-skill : skills.skill_id,
-before_lvl : INTEGER,
-after_lvl: INTEGER,
-impact: INTEGER (before_lvl - after_lvl)

emotions : base info

-emotion_id : PRIMARY KEY SERIAL,
-emotion_text : VARCHAR

emotion_skills : base info

-emotion_skill_id : PRIMARY KEY SERIAL,
-skill_id: skills.skill_id,
-emotion_id: emotions.emotion_id

PAGES

Landing Page :
*Login component?
-- button : loads most recent user record into state

* ! Secondary feature : Continue without logging in . -- button

Update/Finish last Record Page :
*On mount display info on the most recent record -- UpdateRecord component
*set the records after_lvl -- LevelSlider component
* skip btn - Links the i'm feeling page

I'm Feeling Page :
*Typeahead component
*populate with the emotions table
*typing brings up the list
*Clicking on an Emotion component
*Calls the API with the Emotion
*LevelSlider component
*Sets state of before_lvl
*if 6 or 7 trigger SI/SH box
*if SI box click
*set state records.si
*Link to SIResources component
*continue to skills btn
*if SH box click
*set state records.sh
* Display SHSkillsGrid component

9X3 Skills Display :
*SkillsGrid Component
*9 cards with matching skills (Skill component)
*icon and title
*on click Skill component : displays skilldetails

*Pinata component - middle card that generates new cards

2X2 Skills Display :
*SHSkillsGrid Component
*4 cards with matching skills (Skill component)
*icon and title
*on click Skill component : displays skilldetails
*Pinata component - card that generates new cards

User Dashboard page :

*RecordsList component
*map of Record component
*on click: Record component : Load into UpdateRecord component
* ! secondary feature display the info in a chart
*MakeRecord component
*set state - Form - post request

Components:
Record
RecordsList
UpdateRecord ,
MakeRecord,
Skill
SkillsGrid
SHSkillsGrid
EmotionTypeahead
LevelSlider
SIResources
Pinata,
Navbar

state{
record.record_id
record.before_lvl,
record.after_lvl,
record.impact,
record.user_id,
record.emotion,
record.skill,
record.si,
record.sh,
recordsArray,
user.skillsArray, (filter by emotion - pick 2)
emotion.skillsArray, (remove the 2 user skills from the list - pick 3)
skillsArray ,(remove all 5 from the list)
skill_1,
skill_2,
skill_3,
skill_4,
skill_5,
skill_6,
skill_7,
skill_8
}

on login load the most recent record into the state
on submit after_lvl or on click skip btn : clear record form state.

on mount feelings page - post to records with all null values
then get that new record_id and set state record.record_id

on record.emotion change state (select from typeahead) :
Call the API and set sate for user.skillsArray &emotion.skillsArray.
then :
*pick 2 from the user.skillsArray set state skill_4 & skill_5
*filter skill_4 & skill_5 from emotion.skills Array
*pick 3 from emotion.skillsArray set state skill_1, skill_2 ,skill_3
*filter skill_1, skill_2, skill_3, skill_4, skill_5 from the skillsArray
*pick 3 from skillsArray and set sate skill_6, skill_7, skill_8

on record.before_lvl changeState - record it :
this will be a slider or images decide later

on click view records btn : get from records table and set state recordsArray
display : map RecordsList component to Record component

on click : Record component :
load record into record update component

on submit : RecordUpdate component
put to database