1. Create a new project folder
2. Create a “data.json” in the root directory
3. Create a ‘public’
4. Create a ‘views’ folder in the root directory
5. Create three EJS templates in the “views” folder:
   1. “form.ejs”: Has an HTML form with a userName text input, a pogAmount number input, and a submit input. This form has the ‘/submit’ action and the ‘POST’ method
   2. “balance.ejs”: Displays <%- username %>’s <%- pogAmount %>. Has a link to “/”.
      1. If the value of <%- username %> is “none”, write “No user selected” to the page using embedded javascript, and do not write the <%- pogAmount %>.
   3. “list.ejs”: Has a link to “/edit”. Has a embedded JS code block that uses a for loop.
      1. Create an unordered list
      2. For every user in <% data %>, add a list item that shows a link to “/balance” plus a “user” query parameter whose value is the user’s name. The link’s text is also the user’s name.
6. Create an app.js in the root directory.
7. In the app.js file, import all required modules for the project
8. In the app.js file, create an expressJS application
9. In the app.js file, tell the ExpressJS application to use the ‘public’ folder for its static files.
10. In the app.js file, tell the ExpressJS application to use the bodr-parser module to read url encoded body form data.
11. In the app.js file, read the data.json file and save it to a object called “data”.
    1. Then parse the JSON data
12. In the app.js file, create 4 expressJS endpoints:
    1. A GET endpoint at “/” that renders the “list.ejs” template with the <% data %> insert point as the data variable.
    2. A GET endpoint at “/edit” that renders the “form.ejs” template with no insert points.
    3. A GET endpoint at “/balance” that renders the “balance.ejs” template.
       1. If the query parameter “name” has a value that is a key in the data object, send the “name” as the <%- username %> and the key’s value as <%- pogAmount %>
       2. If no query parameters are provided, or the query parameter “name” is not a key in the data object, send “none” as the <%- username %> and 0 as the <%- pogAmount %>.
    4. A POST endpoint at “/submit” that
       1. Adds the pogAmount number to the value belong to the key in the data object that equals the userName text input.
       2. Redirects the user to “/balance?name=X”, where X is the username text input.
13. In the app.js file, start a HTTP listen server with the ExpressJS application.
14. Test all of your work for functionality
15. When complete, delete you ‘node\_modules’ folder
16. Rename your project folder to “FirstnameLastname” and place it in the ‘NEEPogBank” folder of this DocPac directory. If this does not exist, create it. ***Make sure the capitalization and spaces match exactly***
17. Create a Pull Request to submit your work.