SOP Review Portal

Status as of 10/04/2023

Developers
Karen Liu (Frontend)
Jesus Cova (Backend)

Table of Contents

Current Features

- Add/Edit SOPs, Users, Job Roles, User to Job Role relations, SOP to Job Role relations
- Permissions
    - Admin User
    - General User
- Pages/Routes:
    - All Users:
        - Sign In
        - Home (Read SOPs)
    - Admin Only:
        - Users
        - SOPs
        - Job Roles
        - User to Job Role
        - SOP to Job Role

Unfinished Features

- Generate Completion Report - 2 types
    - Shows status of required SOPs for a specific user
    - Shows status of all SOPs (ex. "22/30 users have read *CP-01 Organization*")
    - Show status of every user
    - Examples
- Account Page - General User

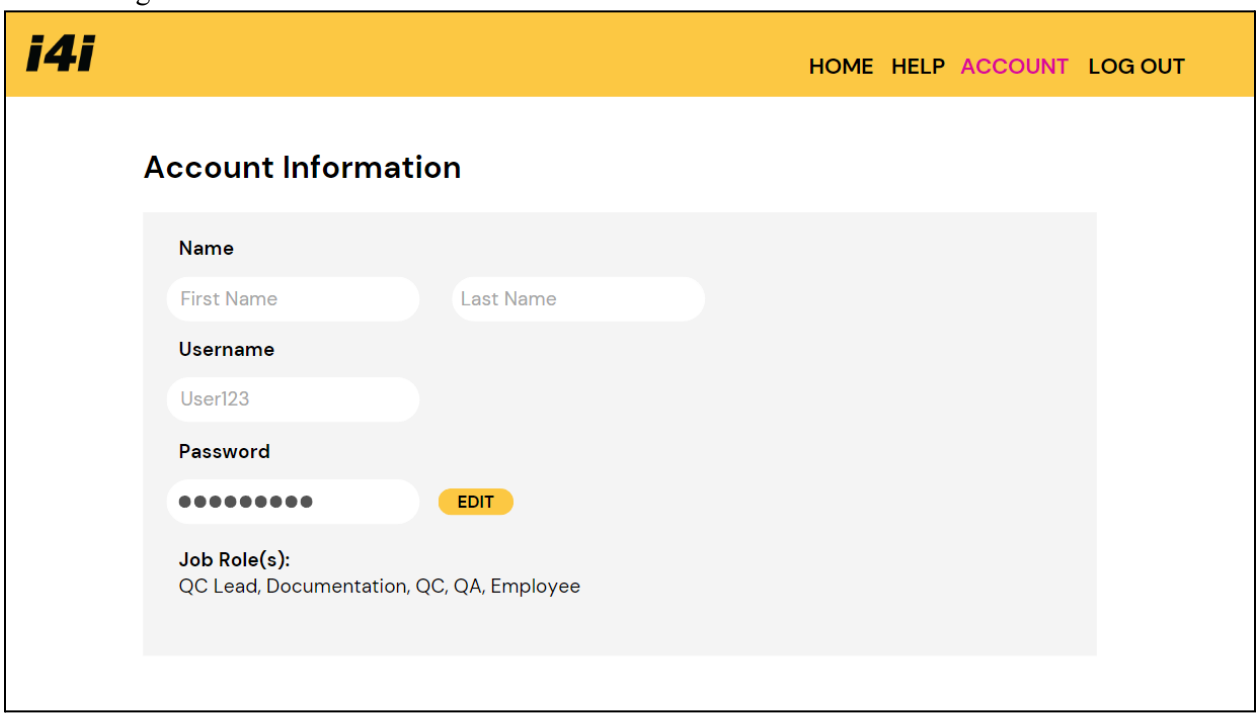


Figure 1 - General User Account Page Design Idea

- An Audit Trail Page
- Form Validation
- Back-End Security Features (JWT)
- Upload File for SOPs

- Toggling Completed Status for read SOPs
- Consider using “sequelize” for the backend/database

Features for the Future and Next Steps

- Table Pagination
- Help Page
- Make code more reusable (a lot of repeated functions and similar components)

Web Framework

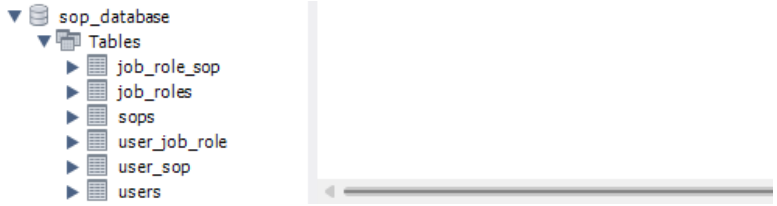
Table 1 - Web Frameworks

	Technology	Version
Front-End Development	React.js	18.2.0
Back-End Development	Express and Node.js	4.18.2 and 18.15.0
Database	MySQL	8.0+
Styling	CSS	CSS3

How to Set up Machine - Local Development

Prerequisites

- Visual Studio Code 2022 Installed
- Node.js Installed (Choose the recommended version)
- MySQL Installed (Web Community installer)
    a. https://www.youtube.com/watch?v=u96rVINbAUI (Video explaining downloading MySQL Workbench)
    b. https://www.youtube.com/watch?v=re3OIOr9dJI (Timestamp: 24:39 explains how to create a Schema and tables)
    c. https://www.youtube.com/watch?v=q5wFWfsS-4I (Video explains how to set up relationships with foreign keys)
- Set up database
    a. After installing MySQL and setting up the MySQL Workbench create a Schema named sop_database (Technically, you can name it anything, just make sure you change it in **db.js** in the project)
    b. Create 6 tables with the following names, columns, and data types



- job_role_sop

Table Name: job_role_sop
Schema: sop_d
Charset/Collation: utf8mb4 utf8mb4_0900_ai_ci
Engine: InnoDB
Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	✓	✓					✓		
job_roleid	INT		✓							
sopid	INT		✓							

- job_roles

Table Name: job_roles
Schema: sop
Charset/Collation: utf8mb4 utf8mb4_0900_ai_ci
Engine: Inn
Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	✓	✓					✓		
title	VARCHAR(45)		✓							

- sops

Table Name: sops
Schema: sop_data
Charset/Collation: utf8mb4 utf8mb4_0900_ai_ci
Engine: InnoDB
Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	✓	✓					✓		
title	VARCHAR(255)		✓							
version	VARCHAR(45)		✓							'0'
effectiveDate	DATE		✓							
link	VARCHAR(1000)		✓							

- user_job_role

Table Name: user_job_role
Schema: sop
Charset/Collation: utf8mb4 utf8mb4_0900_ai_ci
Engine: Inno
Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	✓	✓					✓		
userid	INT		✓							
job_roleid	INT		✓							

- user_sop

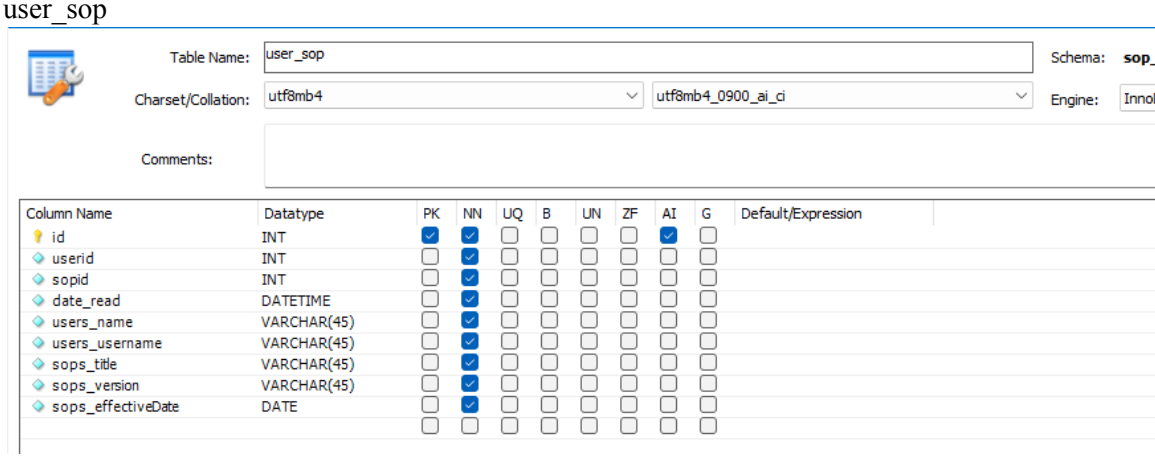


Table Name: user_sop
Schema: sop_
Charset/Collation: utf8mb4 utf8mb4_0900_ai_ci
Engine: Innol
Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	☑	☑	☐	☐	☐	☐	☑	☐	
userid	INT	☐	☑	☐	☐	☐	☐	☐	☐	
sopid	INT	☐	☑	☐	☐	☐	☐	☐	☐	
date_read	DATETIME	☐	☑	☐	☐	☐	☐	☐	☐	
users_name	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
users_username	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
sops_title	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
sops_version	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
sops_effectiveDate	DATE	☐	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	

- users

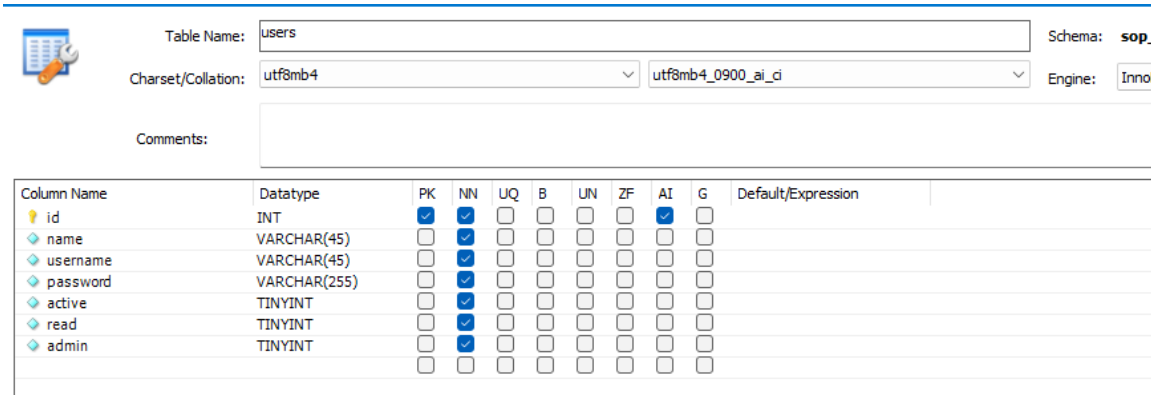
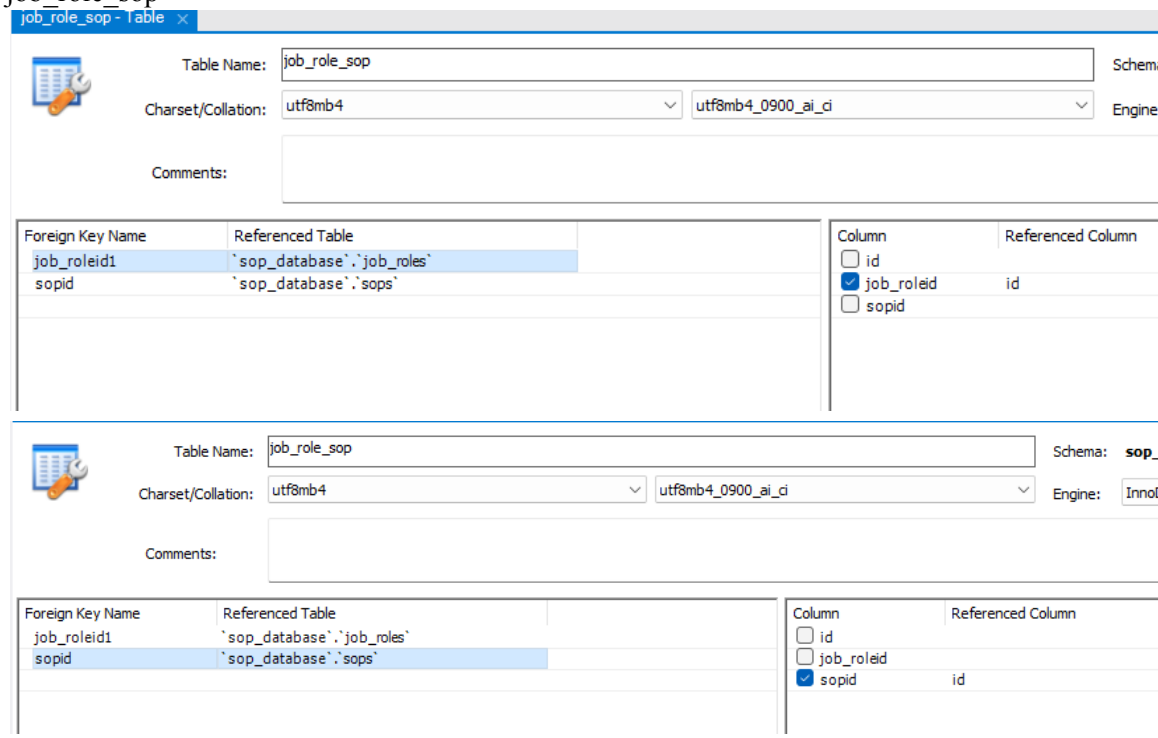


Table Name: users
Schema: sop_
Charset/Collation: utf8mb4 utf8mb4_0900_ai_ci
Engine: Inno
Comments:

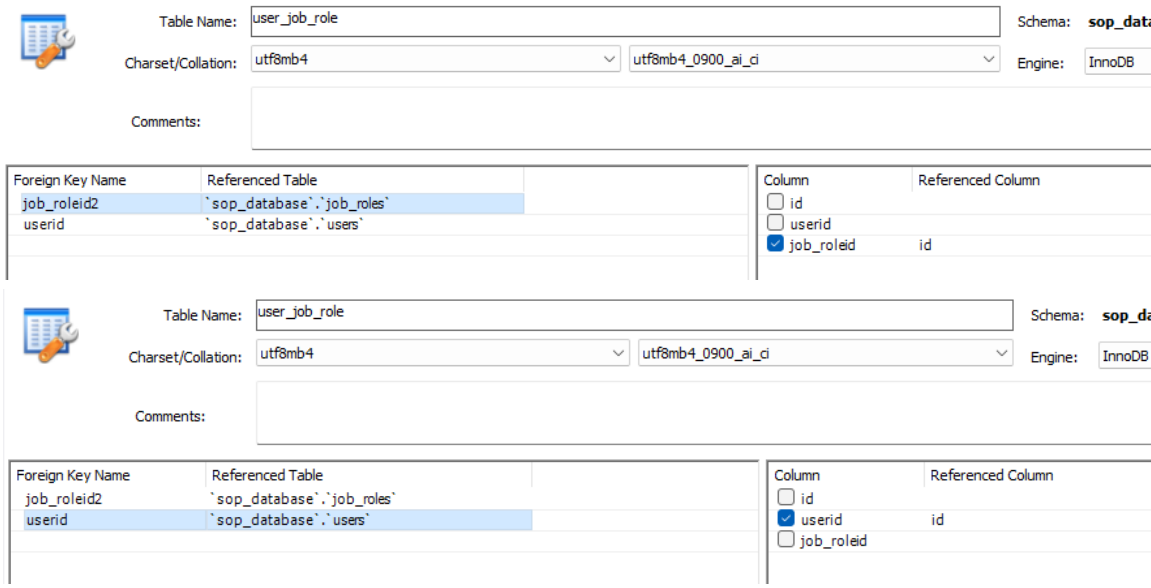
Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	☑	☑	☐	☐	☐	☐	☑	☐	
name	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
username	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
password	VARCHAR(255)	☐	☑	☐	☐	☐	☐	☐	☐	
active	TINYINT	☐	☑	☐	☐	☐	☐	☐	☐	
read	TINYINT	☐	☑	☐	☐	☐	☐	☐	☐	
admin	TINYINT	☐	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	

c. Define relationships using Foreign Keys
   - Foreign Keys are used to reference values from other tables and define relationships. Since these tables have a many-to-many relationship, intermediate tables need to be made to "link" the 2 tables.
   - https://fmhelp.filemaker.com/help/18/fmp/en/index.html#page/FMP_Help/many-to-many-relationships.html (Helps explain the reason for using intermediate tables and foreign keys)
   - (Note: The user_sop table does not use foreign keys despite having information from other tables. Since it's being used as an "audit trail" that may have information from non-existing sops or users, relationships must not be defined for that table)

- job_role_sop



- user_job_role



<u>Create Your Own Project</u>

1. <u>Create React App</u>
   - Run this commands in the VS Code Terminal to create a basic react app and follow through prompts to create a basic react app template:
     - npx create-react-app my-app
2. <u>Delete Unnecessary files</u>
   - Any files that will be replaced by the files uploaded in TFS

3. Install Front-End Dependencies/Libraries
    - Material UI
4. Create Express App
    - In another folder run these commands in the VS Code Terminal to create a basic express app:
        - npm init
        - npm install express
5. Install Back-End Dependencies
    - These are the dependencies used in the project. For each, type this command in the VS Code terminal: npm install <dependency name>
        - mysql2
        - bycryptjs
        - body-parser
        - cookie-parser
        - cors
        - nodemon
        - jsonwebtoken

How to Run the Project

1. Ensure the prerequisites are met
2. Download the project from Visual Studio
3. Open the project folder through VS Code
4. In the VS Code Terminal run these lines
    a. cd server
    b. npm run server
        i. If "Server is running on port 5001" appears in the terminal, the server is running as expected
5. Open another terminal through VS Code and run these lines in the new terminal
    a. cd client
    b. npm run start
        i. You will be redirected to the project page in your default browser

Learning Resources

React.js: https://scrimba.com/learn/learnreact
Express.js and MySQL:

- https://www.youtube.com/playlist?list=PLpPqplz6dKxUaZ630TY1BFIo5nP-_x-nL
- https://www.youtube.com/watch?v=re3OIOr9dJI (Recommended)
- https://www.youtube.com/watch?v=0aPLk2e2Z3g (Recommended)
- https://www.youtube.com/watch?v=8ly39na3LLM
- https://www.w3schools.com/MySQL/default.asp
- https://www.tutorialspoint.com/expressjs/index.htm
- https://www.youtube.com/watch?v=fPuLnzSjPLE

Design Prototypes

- Canva - Round 1
- Figma - Round 2

Project File Structure

Table 2 - Project File Levels

Level 1	Level 2	Level 3	Level 4	Level 5	Level 6
SOP Review Portal	client	public	I4i.png	N/A	
			Index.html		
			manifest.json		
			SOP.png		
		src	components	AdminHome	AddEditSOPDialog.js
					AddSOPForm.js
					EditSOP.js
					SOP.js
					Table.js
				context	authContext.js
				css	AddSOPForm.css
					App.css
					GenerateReport.css
					Navbar.css
					SignIn.css
					Table.css
				GeneralHome	GenerateReport.js
					SOP.js
					Table.js
				JobRoles	AddJBDialog.js
					AddJBForm.js

					EditJB.js
					JB.js
					JobRoleTable.js
				Relate	SOP-JobRole.js
					SOPJB.js
					SOPJBForm.js
					SOPJBRelationDialog.js
					SOPJBRelationForm.js
					UJB.js
					UJBRelationDialog.js
					UJBRelationForm.js
					User-JobRole.js
				SignIn	SignIn.js
				Users	AddUserDialog.js
					AddUserForm.js
					EditUser.js
					User.js
					UserTable.js
				BigButton.js	N/A
				Navbar.js	
				NavButton.js	
				SmallButton.js	
			App.js	N/A	
			Index.js		

			Index.css	
			reportWebVitals.js	
		.gitignore	N/A	
		package-lock.json		
		package.json		
	server	controllers	auth.js	N/A
			jobRole.js	
			role_sop.js	
			sop.js	
			user_role.js	
			user_sop.js	
			user.js	
		routes	auth.js	N/A
			jobRoles.js	
			user_sop.js	
			sops.js	
			user_role.js	
			user_sop.js	
			users.js	
		db.js	N/A	
		index.js		
		package-lock.json		
		package.json		

Table 3 - File/Folder Descriptions

File/Folder Name	Description
.gitignore	Specifies intentionally untracked files that Git should ignore
AddEditSOPDialog.js	Defines dialog used to add SOPs
AddJBDialog.js	Defines dialog used to add Job Roles
AddJBForm.js	Defines the form used to add Job Roles
AddSOPForm.css	Contains the styles in AddSOPForm.js and other similar forms
AddSOPForm.js	Defines the form used to add SOPs
AddUserDialog.js	Defines the dialog used to add users
AddUserForm.js	Defines the form used to add users
AdminHome	Folder for all of the components in the “/SOPs” route
App.css	Contains the styles used in App.js
App.js	Defines the components that display for each route
auth.js	Contains the login and logout functionality
auth.js	Defines the routes for logging in and out and its respective functions
authContext.js	Controls authentication and defines the logged in user
BigButton.js	Defines a large button component
client	Folder for the front-end components of the project
components	Folder for all of the components and pages
context	Folder for all React features using Context
controllers	Contains all the queries/requests with the database
css	Folder for all of the css files
db.js	Defines the connection parameters to the database
EditJB.js	Defines the editable row of Job Roles
EditSOP.js	Defines the editable row of SOPs
EditUser.js	Defines the editable row of users
GeneralHome	Folder for all of the components in the “/Home” route
GenerateReport.css	Defines the style
GenerateReport.js	Defines the generate report button for the
i4i.png	i4i logo used as the website icon
Index.css	Contains the style used in Index.js

Index.html	File that provides context for React to render to. Also defines the website icon, title, and other data
Index.js	File where the App component is rendered
index.js	Defines the dependencies and routes used for the application
JB.js	Defines a read-only row of Job Roles
jobRole.js*	CRUD functionality for the job_roles table
JobRoles	Folder for all of the components in the "/JobRoles" route
jobRoles.js	Defines the routes for the job_roles table with its respective functions
JobRoleTable.js	Defines the table of Job Roles for an Admin user
manifest.json	Contains basic metadata about the website such as name and version
Navbar.css	Contains the style used in Navbar.js
Navbar.js	Defines the navigation bar at the top of the page
NavButton.js	Defines the buttons used in Navbar.js
package-lock.json	lockfile that holds information on the dependencies or packages installed
package.json	contains metadata about the project (name, version, and dependencies)
public	Folder that contains anything that is not used by the app when it compiles
Relate	Folder that contains the components for the SOP-Job Role and User-Job Role relation pages
reportWebVitals.js	Library used to capture the user experience of a web page
role_sop.js	Defines the routes for the job_role_sop table with their respective functions
role_sop.js*	CRD functionality for the job_role_sop table
routes	Folder which contains the files that define the routes for every backend functionality
server	Folder for the back-end components of the project
SignIn	Folder that contains the components for the sign in page
SignIn.css	Contains the style used in SignIn.js
SignIn.js	Defines the sign in page
SmallButton.js	Defines a small button used in the tables (ex. cancel, delete, edit)
SOP Review Portal	Project folder
SOP-JobRole.js	Defines the table of SOP to Job Role relations
SOP.js (AdminHome)	Defines a read-only row of SOPs in the "/SOPs" route
SOP.js (GeneralHome)	Defines a read-only row of SOPs in the "/Home" route

sop.js*	CRUD functionality for the sops table for both admin and general users
SOP.png	Image used in SignIn.js
SOPJB.js	Defines a read-only row of SOP to Job Role relations
SOPJBRelationDialog.js	Defines the dialog used to add SOP to Job Role relations
SOPJBRelationForm.js	Defines a form used to add SOP to Job Role relations
sops.js	Defines the routes for the sops table and their respective functions
src	Contains files that are used when the app is compiled
Table.css	Contains the style used for all of the tables
Table.js (AdminHome)	Defines the table of SOPs for an Admin user
Table.js (GeneralHome)	Defines the home screen containing a table of SOPs to be completed/read
UJB.js	Defines a read-only row of User to Job Role relations
UJBRelationDialog.js	Defines the dialog used to add User to Job Role relations
UJBRelationForm.js	Defines the form used to add User to Job Role relations
user_role.js	Defines the routes for the user_job_role table and their respective functions
user_role.js*	CRD functionality for the user_job_role table
user_sop.js	Defines the audit trail/user_sop table routes and their respective functions
user_sop.js*	CRD functionality for the user_sop table/audit trail
User-JobRole.js	Defines the table of User to Job Role relations
User.js	Defines a read-only row of users
Users	Folder that contains all of the components in the “/Users” route
users.js	Defines the users table routes and their respective functions
users.js*	CRUD functionality for the users table for both admin and general users
UserTable.js	Defines the table of users for an Admin user

Database Tables

users:

id	name	username	password	active	read	AddedBy*	AddedDate*	LastModifiedBy*	LastModifiedDate*

sops:

id	title	version	effectiveDate	link

job_roles:

id	title

user_job_role

id	userid	job_roleid

job_role_sop

id	job_roleid	sopid

user_sop

id	userid	sopid	date_read	users_name	users_username	sops_title	sops_version	sops_effectiveDate

Definitions:

Term	Definition
Audit Trail	References the user_sop table in the database that will store a record of all the sops that users have read
CRUD	Create Read Update Delete
CRD	Create Read Delete

Notes From the Past Developers

Jesus:

- Neither of us had experience using this technology going into the project
- If you don't have any experience, the first week or two will be spent almost entirely on learning and very little progress will be made on the project (Don't worry, that's normal)
- Would recommend creating projects of your own to gain some experience using the technologies during the learning stage
- The reason we chose this technology stack is because:
    a. React and Express both use javascript
    b. They are very popular technologies that have huge communities for help
- The final week will mainly be spent putting your knowledge together to build something functional and documenting everything
- We split the project where one person focused on the front end and the other focused on the backend. If you're focusing on the backend, I would suggest at least learning the React basics and gain some front end experience. Otherwise, you will get lost when integrating both sides.
- Planning how you will complete deliverables will prove to be very important.
- Make sure you've discussed with Gilles and Mike to understand what is preferred/expected when adding a functionality. Ultimately, it'll be Mike and other admin users using the application the most so discuss any features and design ideas with him.
- Create design and functionality mock ups to discuss with Gilles and Mike so that they can provide suggestions
- If you have any design ideas, it'll also be a good idea to bring it up with the doc team as they have very good suggestions
- Gilles is very knowledgeable and technical (especially in SQL) so don't be afraid to ask him for any help
- Make sure to maintain communication with your teammate and keep them updated with your progress and what you'll be working on for the day.

Karen:

- If you are new to web development, the resources that we have provided in the Learning Resources section are a good start, since we built the website off of those tutorials.
- Make sure that the deliverables/features that you wish to complete are reasonable and within your capabilities (3 weeks goes by very quickly and you may not get everything done!)
- Make sure to plan out what you want to go over in meetings (ex. Progress, questions, next steps) prior, as you may not get too many meetings
- If you are stuck, there will probably be an answer to your question on stack overflow, W3schools and other websites. Look things up!
- Good luck!! 😎 This project is a great way to build web dev experience 😀