**App Component (App.jsx):**

The App component serves as the root component of your application and is responsible for rendering the navigation bar and handling routes using the react-router-dom library.

**Create Area(CreateArea.jsx):**

CreateArea Component

useState Hooks:

isExpanded: A state to manage whether the note creation area should be expanded or not.

status: A state to manage the status of the note (e.g., "Learned", "Want to Learn", "Noted").

note: A state to manage the note's data, including title and content.

useEffect Hook:

This effect runs when props.editNote changes.

If props.editNote is provided, the component populates the note state with the title and content from props.editNote, and sets isExpanded to true. This is likely used for editing an existing note.

handleChange Function:

Handles changes to the form inputs (title and content).

It updates the note state whenever the user types in either the title or content fields.

submitNote Function:

Handles the submission of the note.

Prevents the default form submission behavior.

Checks if both title and content are provided, shows an alert if either is missing.

Calls props.onAdd, passing the note and status (likely a function passed from the parent component to add or update the note).

Resets the note state and sets isExpanded to false.

expand Function:

Expands the note creation area by setting isExpanded to true.

JSX Return:

Returns a form for creating or editing a note.

The first textarea is for the title. It expands the area on click.

The second textarea for the content is displayed only if isExpanded is true.

A dropdown select for the status of the note is also shown if isExpanded is true.

A submit button, visible when isExpanded is true, to submit the note.

Navigation Bar (<nav>): It provides links to navigate between the Home, Flashcards, and Contact pages.

Routes (<Routes>): It defines routes for the Home, Flashcards, and Contact pages using the <Route> component.

**Home Component (Home.jsx):**

The Home component is responsible for rendering the home page content.

Project Component

Purpose: The Project component is designed to display individual project information.

Props: It receives several properties (props) from its parent component:

id: A unique identifier for the project.

link: A URL to the project's webpage.

img: A source link for the project's image.

name: The name of the project.

Rendering:

The component returns a <div> with a class name project-info and an id attribute set to the passed id prop.

Inside this div, there's an anchor (<a>) tag that wraps an image and a heading. The anchor tag's href attribute is set to the link prop, allowing users to navigate to the project's webpage.

The image (<img>) displays the project's picture using the img prop as its source.

The project's name is displayed in a <h3> tag, using the name prop.

Home Component

Purpose: The Home component serves as a container for the overall layout of a webpage, specifically focusing on displaying a list of projects.

Import: It imports projects from "./projects", which is likely an array of project objects.

Rendering:

The component returns a <div> with a class name home, encapsulating all its content.

Within this div, there are two main sections:

Header Section: Contains a brief personal introduction, with a heading (<h1>) that links to a GitHub profile and a paragraph (<p>) describing the person as a "Senior Computer Science Student."

Main Content Section: Contains a list of projects.

It starts with a heading <h1> stating "List of Projects."

The projects are displayed in a <div> with the class proj-container.

The projects array is mapped over, and for each project object (proj), a Project component is rendered. The properties of each project (like id, name, img, link) are passed down to each Project component.

Project Component: The Project component is a functional component that receives project information (id, name, img, link) as props and displays it.

Projects List: The Home component maps through the projects array and renders a list of projects using the Project component.

**Card Component (Card.jsx):**

The Card component is responsible for rendering individual flashcards and handling card-specific actions.

handleSearchChange:

Purpose: Updates the searchTerm state based on the user's input in the search field.

Functionality: Takes an event object as a parameter, updates searchTerm state with the current value in the search input field.

filteredNotes:

Purpose: Filters the flashcards based on the search term and selected filter.

Functionality: Uses filter and searchTerm states to filter notes. It includes logic to check if the note's question or answer includes the search term and if the note's status matches the selected filter.

fetchNotes:

Purpose: Fetches flashcard data from a server.

Functionality: Makes an HTTP GET request to a specified URL, updates the notes state with the fetched data, and sorts them by their order property.

onDragEnd:

Purpose: Handles the logic when a drag-and-drop action ends.

Functionality: Updates the order of the notes based on the drag result and calls updateOrderOnServer to reflect these changes on the server.

updateOrderOnServer:

Purpose: Updates the order of the flashcards on the server.

Functionality: Iterates over the notes and makes an HTTP PUT request for each note to update its order on the server.

handleFilterChange:

Purpose: Updates the filter state when a different filter option is selected.

Functionality: Takes an event object as a parameter, updates the filter state with the selected filter value.

addOrUpdateNote:

Purpose: Adds a new flashcard or updates an existing one.

Functionality: Depending on whether editNote is set, it either updates an existing flashcard or adds a new one to the server and updates the notes state accordingly.

deleteNote:

Purpose: Deletes a flashcard.

Functionality: Makes an HTTP DELETE request to remove a flashcard from the server and updates the notes state to remove it from the UI.

editNoteHandler:

Purpose: Sets the flashcard to be edited.

Functionality: Finds the flashcard to be edited from notes and sets it to editNote.

handleSortChange:

Purpose: Updates the sortOption state when a different sorting option is selected.

Functionality: Takes an event object as a parameter, updates the sortOption state with the selected sorting value.

sortedAndFilteredNotes:

Purpose: Sorts and filters the flashcards based on selected sorting and filtering options.

Functionality: Applies sorting and filtering logic on filteredNotes based on the sortOption state.

handleNoteSelection:

Purpose: Manages the selection of flashcards for sharing.

Functionality: Updates the selectedNotes state to include or exclude the ID of a selected note.

shareNotes:

Purpose: Prepares and triggers the sharing of selected notes.

Functionality: Constructs a mailto link with the selected notes in JSON format for sharing via email.

**Contact Component (Contact.jsx):**

The Contact component renders a form for users to submit contact information.

useState for formData:

Purpose: Manages the state of the form data (subject, email, content).

Functionality: Initializes the form data with empty strings and provides a method (setFormData) to update this state.

useState for messageSent:

Purpose: Tracks whether the message has been successfully sent or not.

Functionality: Initializes the messageSent state as false and provides a method (setMessageSent) to update this state.

handleChange:

Purpose: Handles changes in the form inputs.

Functionality: Updates the formData state whenever a user types in the form fields. It dynamically sets the value for each field (subject, email, content) based on the name attribute of the input and the current value.

handleSubmit:

Purpose: Handles the form submission.

Functionality: Prevents the default form submission behavior, prepares the message data from the formData state, and sends a POST request to a specified URL (presumably a server endpoint) with the message data. The function also handles the response:

Sets messageSent to true on successful submission (HTTP status 201).

Logs an error message for a 404 error (Not Found) or other errors.

Catches and logs network errors.