**what wrong will be there if I will not use unwrap() function here?**

 const submitHandler = async (e) => {

    e.preventDefault();

    try {

      const res = await login({ email, password }).unwrap();

      // console.log(res);

      dispatch(setCredentials({ ...res }));

      navigate(redirect);

    } catch (err) {

      toast.error(err?.data?.message || err.error);

    }

  };

In the context of Redux Toolkit's createAsyncThunk or RTK Query's createApi, the unwrap() function is used to return the fulfilled value of a promise, or throw the rejected value. This can be particularly useful for handling the result of the promise directly in a try/catch block.

If you do not use unwrap(), here are the potential issues you might face:

1. **Error Handling**: Without unwrap(), if the login promise is rejected, it will not throw an error that can be caught by the catch block. Instead, the res variable will simply contain the rejected promise, which may not have the desired structure or data you expect in an error scenario. This means the toast.error line in the catch block may not work correctly because err may not contain the expected error information.
2. **Result Handling**: The unwrap() function extracts the actual data from the fulfilled promise. Without it, res will be the entire action object, not just the data returned by the API call. This means your dispatch(setCredentials({ ...res })) line will be trying to spread an action object, which could lead to unexpected behavior or errors.