COMP6080
Web Front-End Programming

Week 7.1
Assignment 3 & JS-Stack Projects

Starting a ReactJS application

Now we're going to start a ReactJS application inside this repository. We'll use **create-react-app** for this, which will by default use **yarn** as the package manager.

Building Components, Talking to the API

A common design pattern for ReactJS apps is to use **react-router-dom** to manage your page routes, and then have two key folders in the `src` directory:

- "/components/": For all components that don't map directly to routes
- "/pages/": For all components that map directly to routes

Let's explore this together.

Using Libraries in ReactJS

One of the best aspects about working on these ReactJS projects is that you can easily leverage countless node packages. It is VERY rare that you will have to reinvest the wheel.

A simple example for assignment 3:

- Trying to setup a countdown timer

Vanilla V React - Actions

While we learned about "addEventListener" for vanilla JS webpages, it's important we use React's actions on components (with limited exceptions).

This ensures that React can manage it's life cycle most appropriately.

Vanilla V React - Actions

```
const Goals = () => {
  const [showModal, setShowModal] = React.useState(false);
  const [showProgress, setShowProgress] = React.useState<boolean>(false);
  const [showDate, setShowDate] = React.useState<boolean>(false);
  const [showAge, setShowAge] = React.useState<boolean>(false);

  const messages = useMessages();
  const { width } = useWindowSize(initialScreenWidth);
  const isMobileOrSmaller = width < mobileBreakpoint;

  const homePageQuery = useGetHomePageInfoQuery({
    variables: {
      type: FiCalculatorScenarioType.PrimaryGoal,
    },
  });
  return checkQuery(homePageQuery, ({ fireCalculator, getSelf }) => {
    const { passiveIncome, investmentBalance } = getSelf.mergedPortfolio;
    const passiveIncomePa = fireCalculator ? fireCalculator.passiveIncomePa : 0;
    const FIDate = fireCalculator ? fireCalculator.FIDate : undefined;
    const FIAge = fireCalculator ? fireCalculator.retirementAge : undefined;

    const overallProgress =
      passiveIncome !== 0 && passiveIncomePa !== 0 ? passiveIncome / passiveIncomePa : 0;
    // const timeUntilFI = new Date(data.FIDate).getTime() - new Date().getTime();
    // const daysUntilFI = timeUntilFI / 1000 / 60 / 60 / 24;
    // const monthsUntilFI = Math.floor(daysUntilFI / 30.41666);
    // const yearsUntilFI = Math.floor(monthsUntilFI / 12);

    return (
      <>
        <Head>
          <title>{pearlerTitle('Goals')}</title>
        </Head>
        <Flex
          p={7}
          flexDirection="column"
          onClick={() => {
            if (showProgress) {
              setShowProgress(false);
            }
            if (showDate) {
              setShowDate(false);
            }
            if (showAge) {
              setShowAge(false);
            }
          }}
        >
          <Flex pb={4} justifyContent="space-between" flexWrap="wrap">
            <PText pr={4} pb={[4, 4, 4, 0]} font="h3" color={colors.textDark}>
              Your Investments Goal
            </PText>
            <Flex>
              <PButton isInverted="default" mr={4} onClick={() => setShowModal(true)}>
                Edit
              </PButton>
            </Flex>
```

```
21  const Goals = () => {
22    const [showModal, setShowModal] = React.useState(false);
23    const [showProgress, setShowProgress] = React.useState<boolean>(false);
24    const [showDate, setShowDate] = React.useState<boolean>(false);
25    const [showAge, setShowAge] = React.useState<boolean>(false);
26
```

```
67            }}
68          >
69            <Flex pb={4} justifyContent="space-between" flexWrap="wrap">
70              <PText pr={4} pb={[4, 4, 4, 0]} font="h3" color={colors.textDark}>
71                Your Investments Goal
72              </PText>
73              <Flex>
74                <PButton isInverted="default" mr={4} onClick={() => setShowModal(true)}>
75                  Edit
76                </PButton>
77              </Flex>
78            </Flex>
79            <Flex sx={{ borderBottom: `1px solid ${colors.neutralAsh}` }} />
80            <Image mt={7} size={48} src={goalsImg} alt="Goals Image" />
81            <Flex width={1} justifyContent="space-between" mb={3}>
```

Vanilla V React - Actions

```
14  export interface OnboardIdCheckProps extends OnboardComponentSharedProps {}
15
16  const OnboardAMLSubmit: React.FC<OnboardIdCheckProps> = props => {
17    const [mutate, mutationResult] = useOnboardingAmlDigitalCheckMutation();
18    const loading = checkMutation(mutationResult, () => {
19      props.setCompleted();
20    });
21    React.useEffect(() => {
22      (window as any).digitalIdAsyncInit = (t => {
23        return () => {
24          (digitalId as any).init({
25            clientId: config.digitalid.id,
26            onComplete,
27          });
28        };
29      })();
30    }); // TODO: Removed dependency array
31
```

Assignment 3

Let's explore this together.

Express Servers

The most common way to build an HTTP server with NodeJS (Javascript) is to build an express server. Let's do this with NPM.

Using a limited knowledge of HTTP, combined with your knowledge of Javascript so far, this can be a very simple library to make sense of.