

Bootcamp 2020 Class 6: React Expense Tracker

Bootcamp 2020 Class 6: React Expense Tracker	1
Part I - Create React Project using Create React App	2
Part II - Open the Project with VS Code	4
Part III - View the Project in the Browser	6
Part IV - Add Repository to GitHub with the CLI	9
Part V - Deploy React App to Surge with a GitHub Workflow	17
Part VI - Create Components	22
Part VII - Create Hook for Updating State with Add Transaction	28
Part VIII - Create Context and Reducer	31
Part IX - Show Transactions from State the Transaction History	37
Part X - Differentiating between Income and Expense	39
Part X - Updating the Account Summary	40
Part XI - Updating the Balance	41
Part XII - Add an Action to Delete Existing Transactions	42
Part XIII - Add an Action to Add New Transactions	45

Part I - Create React Project using Create React App

Step 1: Create a new local repository

First, I navigate to my local GitHub folder where all my local repositories are kept.

```
Adil:~ adil$ cd /Users/adil/Documents/GitHub_
```

Now I create my local repository by creating my basic react application with:

```
npx create-react-app <app-name>
```

```
[Adil:GitHub adil$ npx create-react-app react-expense-tracker]
npx: installed 98 in 10.594s

Creating a new React app in /Users/adil/Documents/GitHub/react-expense-tracker.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...

yarn add v1.19.1
[1/4] 🔎 Resolving packages...
[2/4] 🚀 Fetching packages...
[3/4] ⚡ Linking dependencies...
warning "react-scripts > @typescript-eslint/eslint-plugin > tsutils@3.17.1" has unmet peer dependency "typescript@>=2.8.0 || >= 3.2.0-dev || >= 3.3.0-dev || >= 3.4.0-dev || >= 3.5.0-dev || >= 3.6.0-dev || >= 3.6.0-beta || >= 3.7.0-dev || >= 3.7.0-beta".
[4/4] ↗ Building fresh packages...
success Saved lockfile.
warning Your current version of Yarn is out of date. The latest version is "1.22.4", while you're
on "1.19.1".
info To upgrade, run the following command:
$ brew upgrade yarn
success Saved 14 new dependencies.
info Direct dependencies
├── cra-template@1.0.3
│   └── react-dom@16.13.1
└── react-scripts@3.4.1
    └── react@16.13.1
info All dependencies
└── @babel/plugin-syntax-typescript@7.10.1
└── @babel/plugin-transform-flow-strip-types@7.9.0
└── @babel/plugin-transform-runtime@7.9.0
```

```
Success! Created react-expense-tracker at /Users/adil/Documents/GitHub/react-expense-tracker
Inside that directory, you can run several commands:

yarn start
  Starts the development server.

yarn build
  Bundles the app into static files for production.

yarn test
  Starts the test runner.

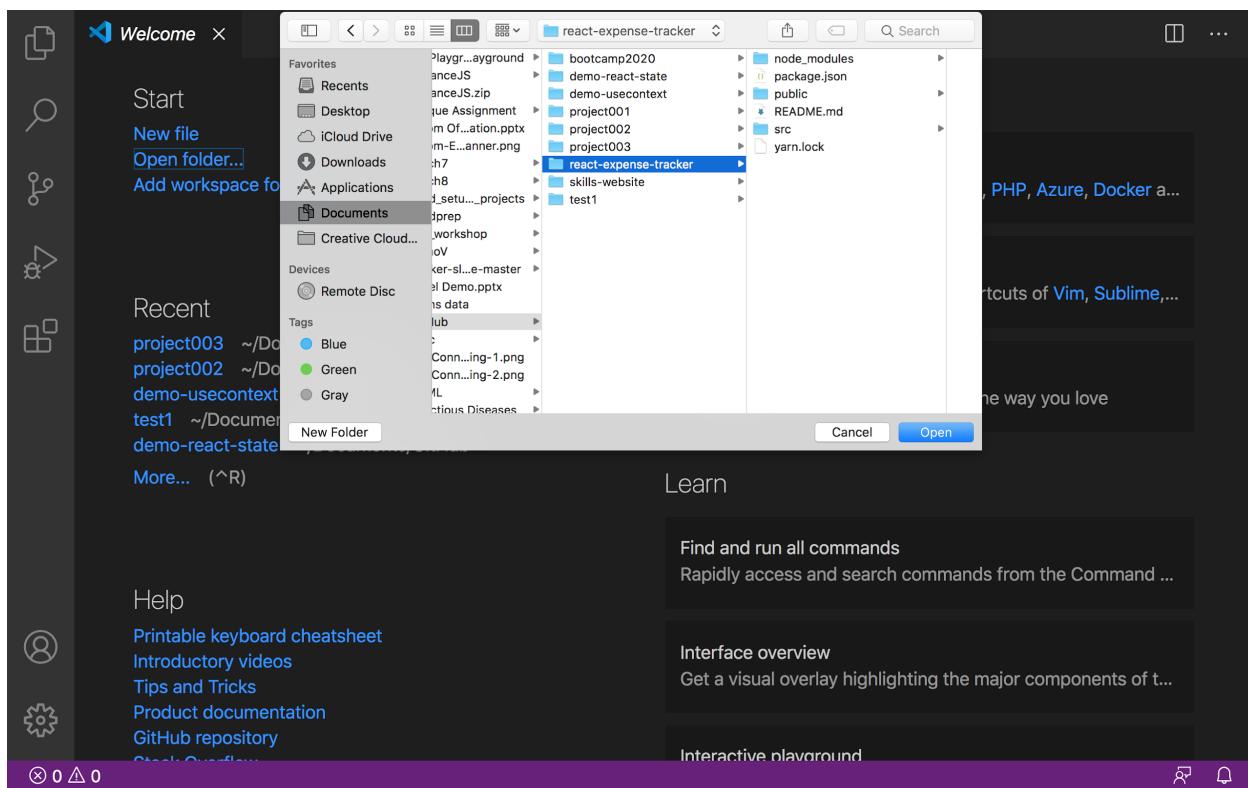
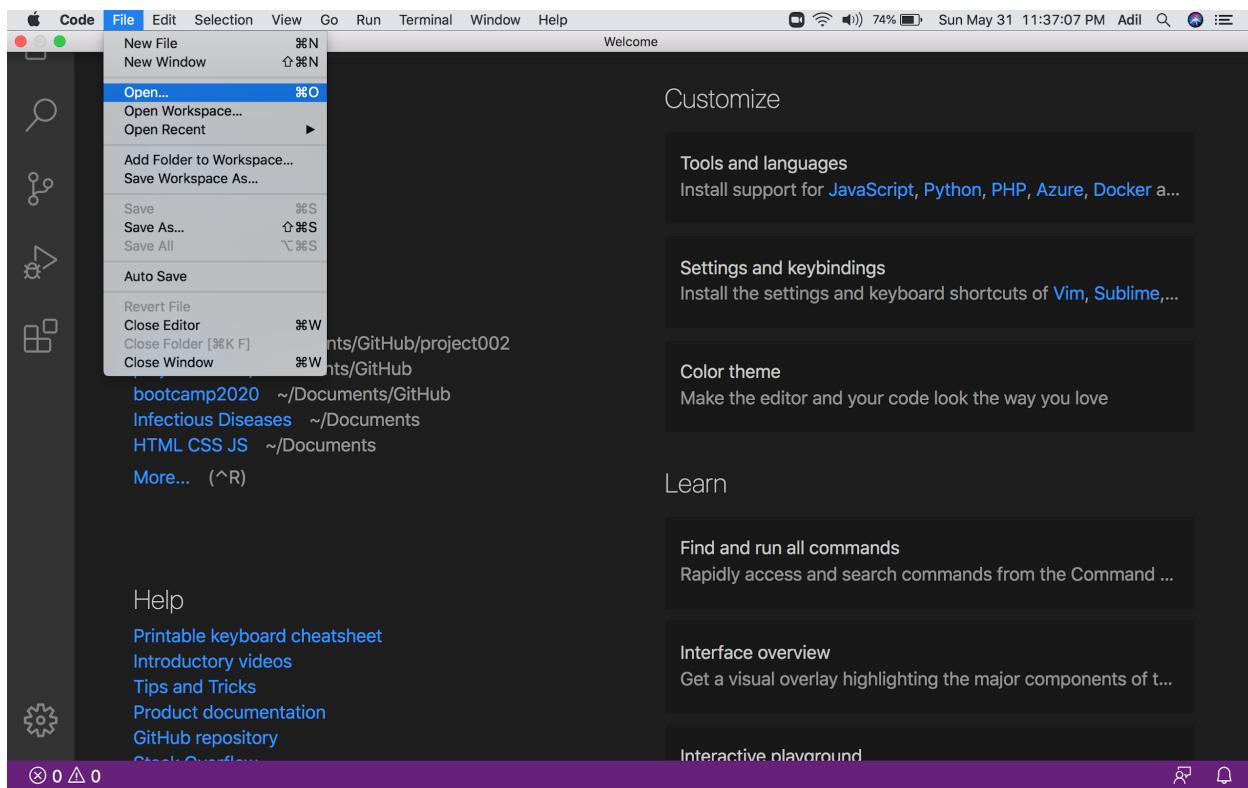
yarn eject
  Removes this tool and copies build dependencies, configuration files
  and scripts into the app directory. If you do this, you can't go back!

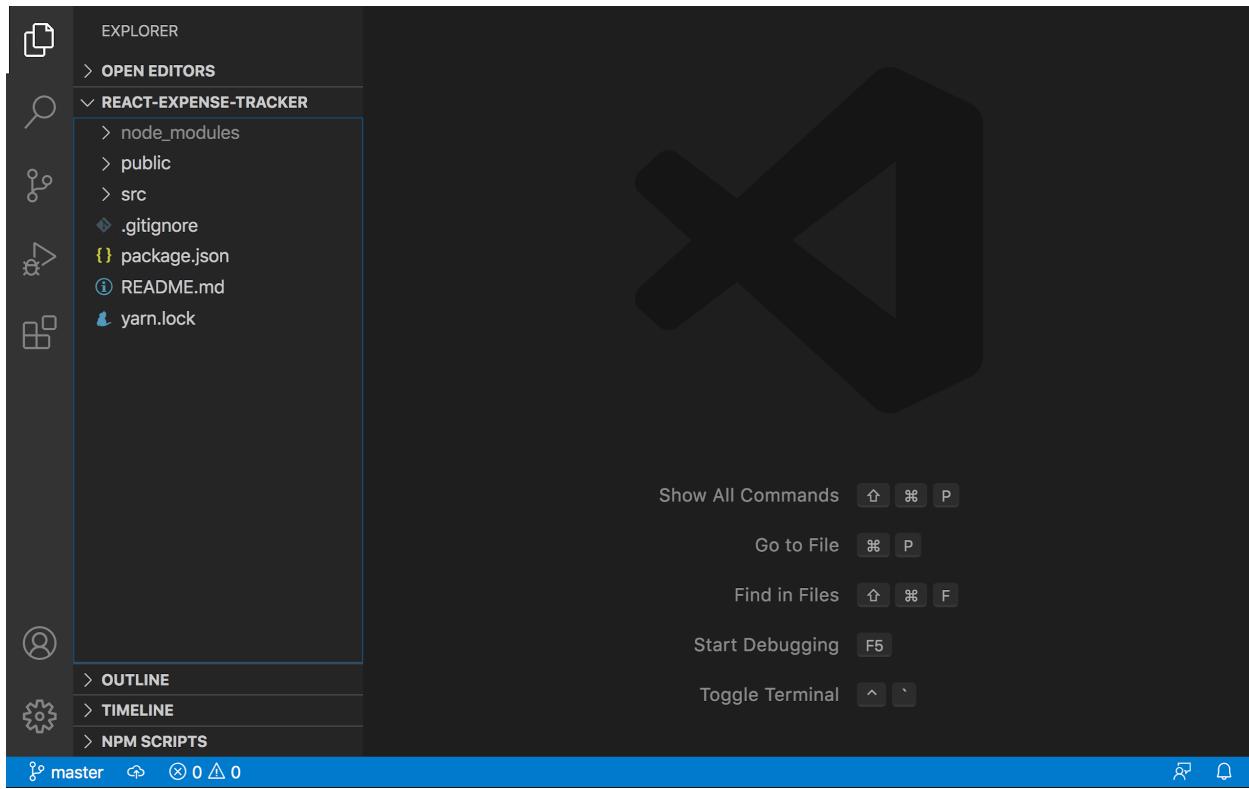
We suggest that you begin by typing:

  cd react-expense-tracker
  yarn start

Happy hacking!
Adil:GitHub adil$ _
```

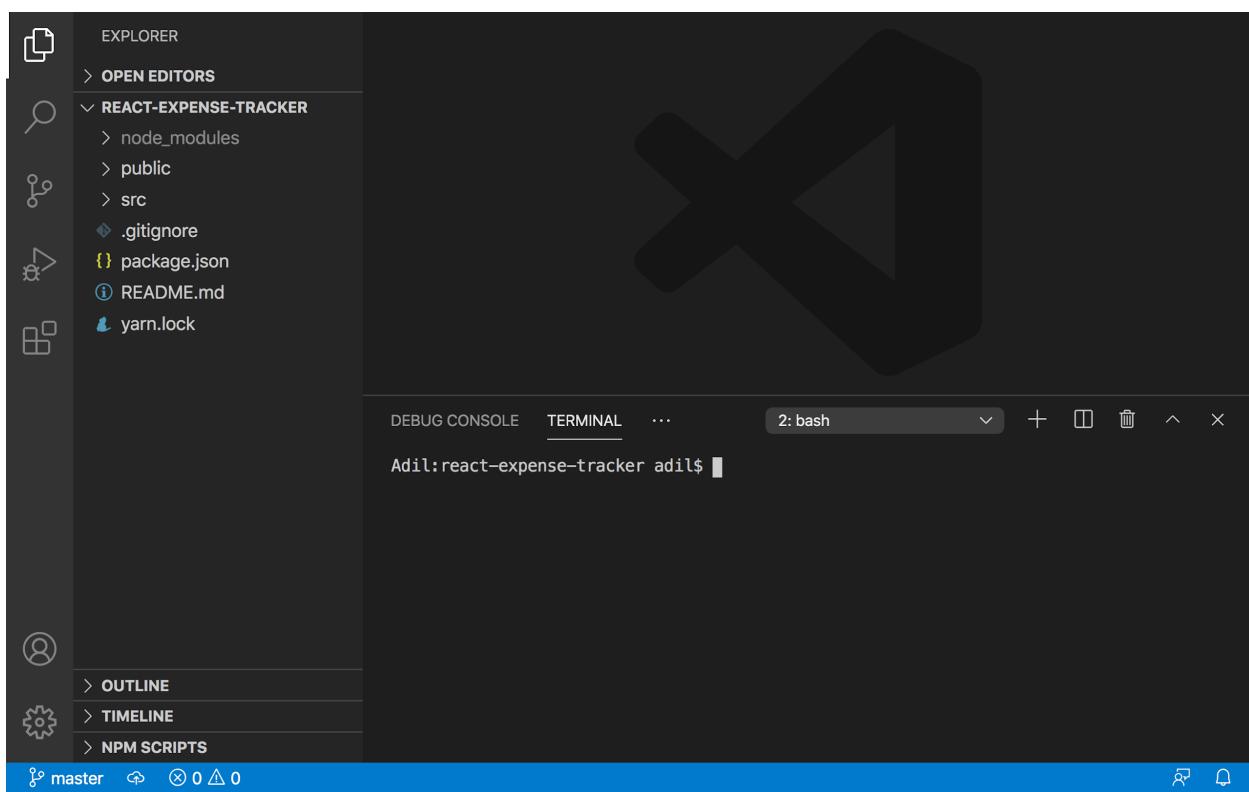
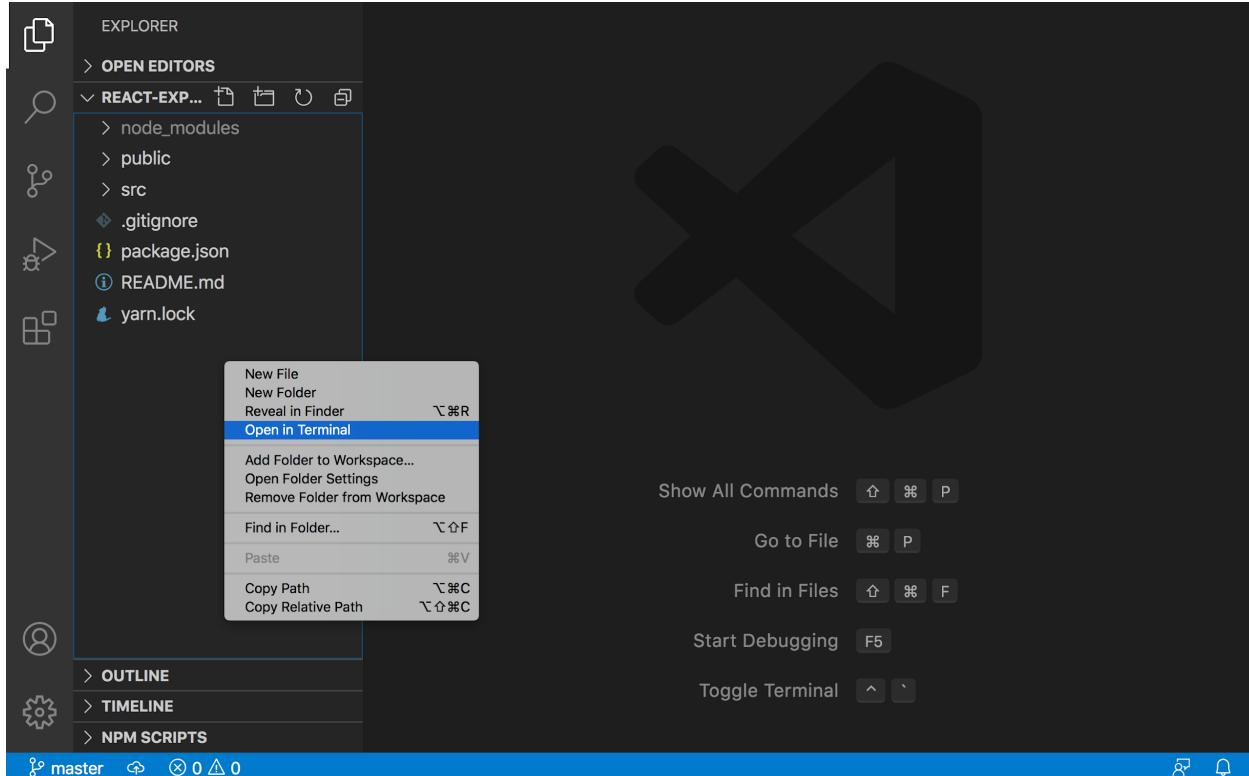
Part II - Open the Project with VS Code





Part III - View the Project in the Browser

First, let's open the project in the terminal



VS Code interface showing the terminal output of running the application:

```
Adil:react-expense-tracker adil$ yarn start
```

VS Code interface showing the terminal output of running the build command:

```
Compiled successfully!
```

You can now view **react-expense-tracker** in the browser.

```
Local: http://localhost:3000
On Your Network: http://192.168.1.4:3000
```

Note that the development build is not optimized.
To create a production build, use **yarn build**.



Part IV - Add Repository to GitHub with the CLI

Step 1: Create a new repository on GitHub

The screenshot shows the GitHub homepage. On the left, there's a sidebar with 'Repositories' and a search bar. Below it, under 'Working with a team?', there's a section about GitHub being built for collaboration and a 'Create an organization' button. The main area shows 'Recent activity' with notifications for completed assignments and followings from users like mnomanfarooq, owsy123, and RooshanAhmed. There are also promotional banners for 'GitHub Satellite' and 'GitHub is now free for teams'. On the right, there's an 'Explore repositories' section with links to repositories like jina-ai/jina, jnln-org/jnln, facebook/react, and Jina AI.

Do not initialize with a readme file, gitignore, or license:

The screenshot shows the 'Create a new repository' page. At the top, it says 'Create a new repository' and provides a brief description of what a repository is. Below that, the 'Owner' field is set to 'adil-innovation-lab' and the 'Repository name' field is 'react-expense-tracker'. A note says 'Great repository names are short and memorable. Need inspiration? How about redesigned-funicular?'. The 'Description (optional)' field contains 'Expense tracker app in react using functional components with hooks and context API'. Under 'Visibility', the 'Public' option is selected, with a note that anyone can see the repository. The 'Private' option is also available. A note at the bottom says 'Skip this step if you're importing an existing repository.' There are checkboxes for 'Initialize this repository with a README' (which is checked) and 'Add .gitignore: None'. At the bottom, there are buttons for 'Add a license: None' and a large green 'Create repository' button.

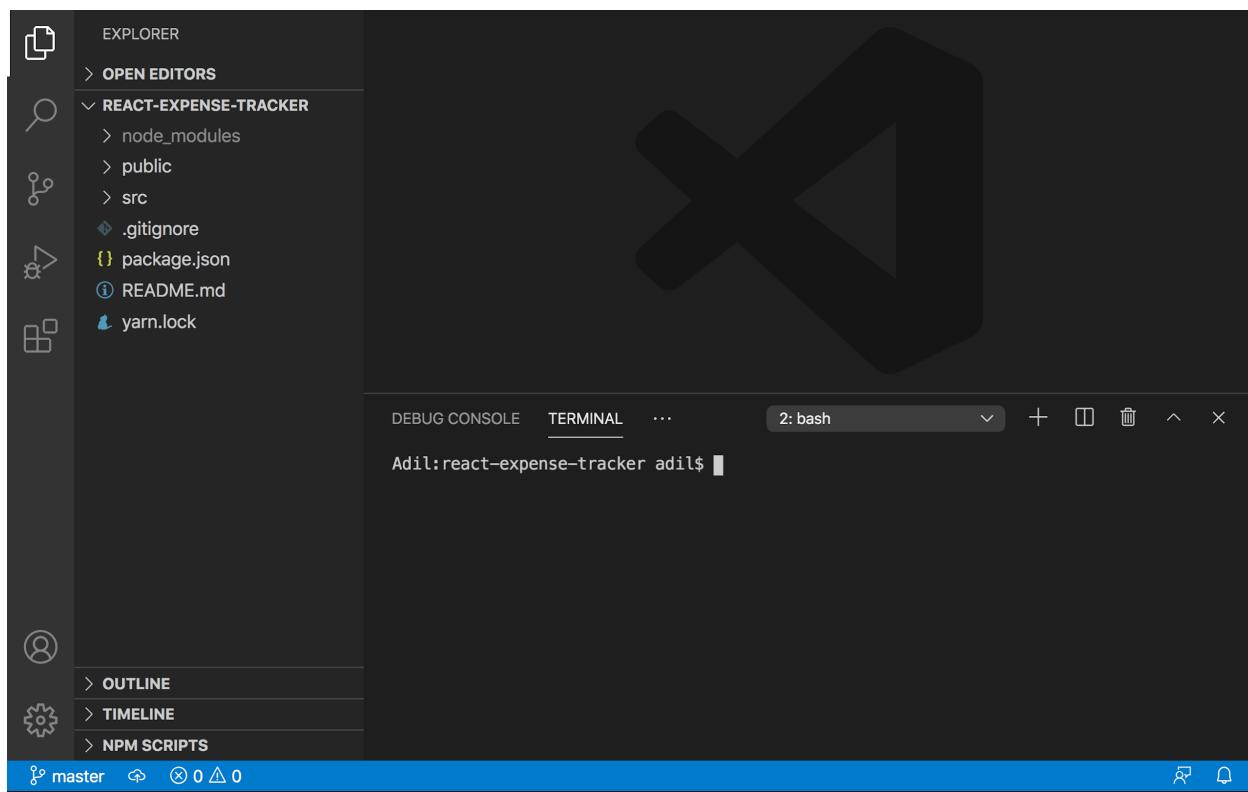
The screenshot shows a GitHub repository page for 'react-expense-tracker'. At the top, there are navigation links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the header, there's a search bar and a quick setup section with instructions for cloning the repository. It also provides command-line scripts for initializing a new repository or pushing an existing one.

```
echo "# react-expense-tracker" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/adil-innovation-lab/react-expense-tracker.git
git push -u origin master
```

```
git remote add origin https://github.com/adil-innovation-lab/react-expense-tracker.git
git push -u origin master
```

Step 2: Initialize the local directory as a git repository:

Open terminal in our local folder:



VS Code interface showing the Explorer, Terminal, and Activity Bar.

EXPLORER

> OPEN EDITORS

✓ REACT-EXPENSE-TRACKER

- > node_modules
- > public
- > src
- ◆ .gitignore
- { } package.json
- ⓘ README.md
- (blue icon) yarn.lock

TERMINAL

2: bash

```
Adil:react-expense-tracker adil$ git init
```

ACTIVITY BAR

master 0 △ 0

VS Code interface showing the Explorer, Terminal, and Activity Bar.

EXPLORER

> OPEN EDITORS

✓ REACT-EXPENSE-TRACKER

- > node_modules
- > public
- > src
- ◆ .gitignore
- { } package.json
- ⓘ README.md
- (blue icon) yarn.lock

TERMINAL

2: bash

```
Adil:react-expense-tracker adil$ git init
Reinitialized existing Git repository in /Users/adil/Documents/GitHub/react-expense-tracker/.git/
Adil:react-expense-tracker adil$
```

ACTIVITY BAR

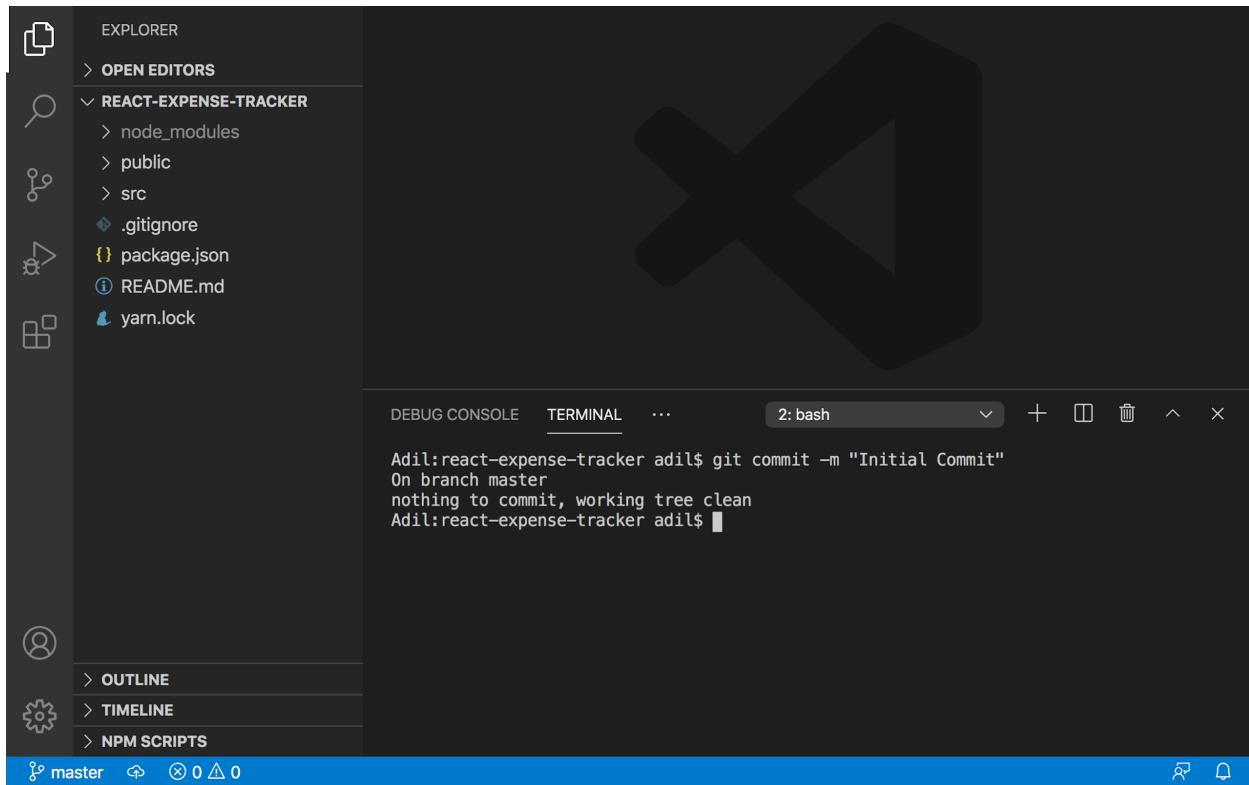
master 0 △ 0

Step 3: Add files from local directory to git staging area to prepare for commit

A screenshot of the Visual Studio Code interface. The left sidebar shows the Explorer view with a tree structure of a 'REACT-EXPENSE-TRACKER' project containing 'node_modules', 'public', 'src', '.gitignore', 'package.json', 'README.md', and 'yarn.lock'. The bottom status bar shows the current branch is 'master'. The main area is the Terminal tab, which displays the command 'Adil:react-expense-tracker adil\$ git add .' entered by the user.

Step 4: Commit files from staging area to local git repository

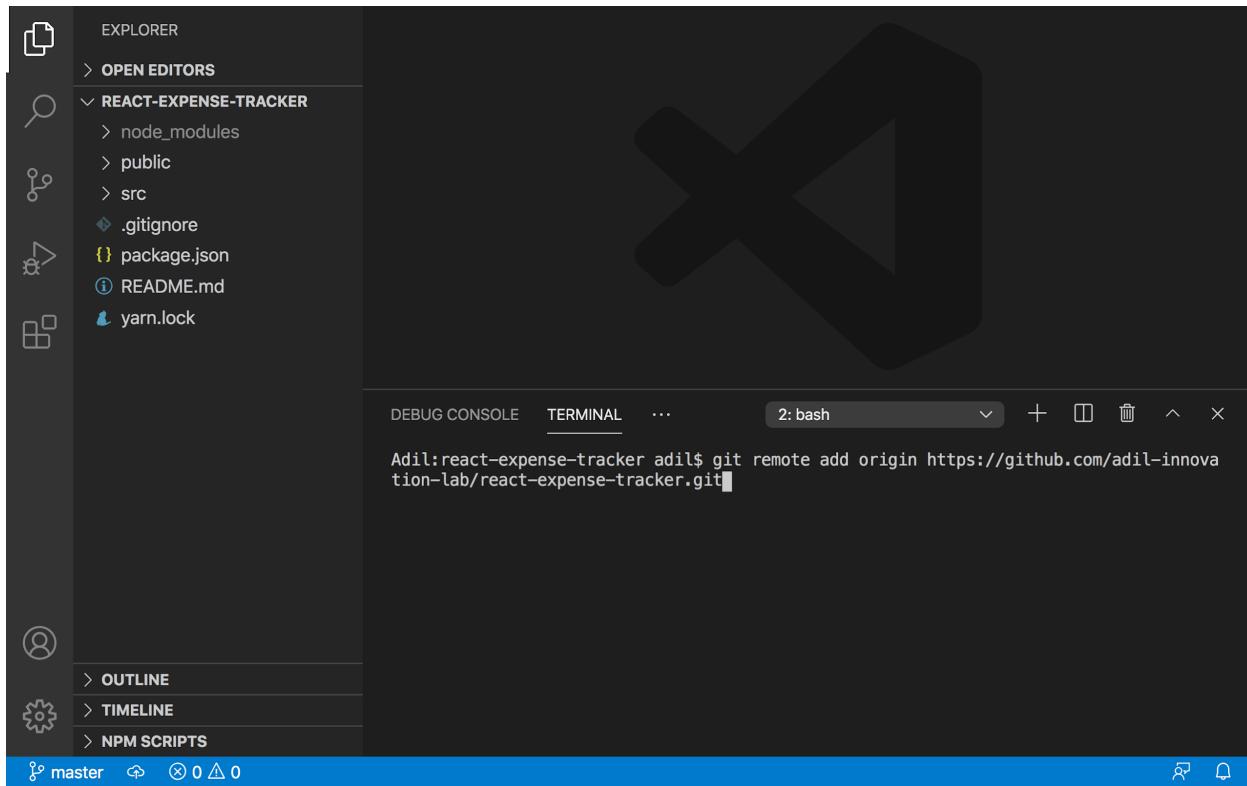
A screenshot of the Visual Studio Code interface, identical to the previous one but with a different terminal command. The Terminal tab now shows 'Adil:react-expense-tracker adil\$ git commit -m "Initial Commit"' entered by the user.



Step 5: Get the URL of the remote repository from GitHub:

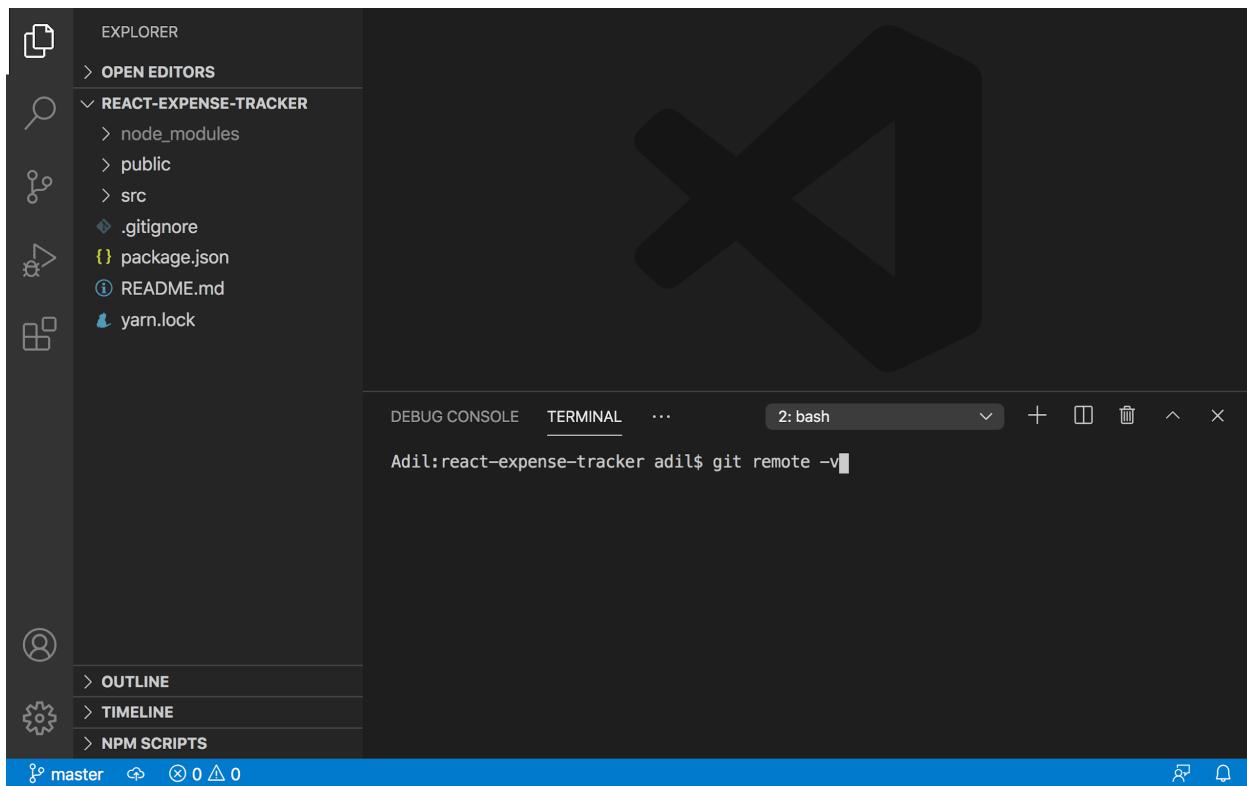
A screenshot of a GitHub repository page for 'adil-innovation-lab/react-expense-tracker'. The page includes navigation links for Pull requests, Issues, Marketplace, and Explore. Below the header, there's a summary of repository activity: 1 unwatched, 0 stars, 0 forks, and 0 issues. The main content area has sections for 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. A prominent feature is a 'Quick setup' box at the top with the URL 'https://github.com/adil-innovation-lab/react-expense-tracker.git'. Below it, there are instructions for creating a new repository on the command line and pushing an existing one.

Step 6: Add remote URL to git via the terminal so local repository can be pushed to the right place



The screenshot shows the VS Code interface with the terminal tab active. The terminal window displays the command: `Adil:react-expense-tracker adil$ git remote add origin https://github.com/adil-innovation-lab/react-expense-tracker.git`. The rest of the screen is dark, with the large X logo visible.

Verify the remote URL



The screenshot shows the VS Code interface with the terminal tab active. The terminal window displays the command: `Adil:react-expense-tracker adil$ git remote -v`. The rest of the screen is dark, with the large X logo visible.

The screenshot shows the Visual Studio Code interface with a dark theme. On the left is the Explorer sidebar, which lists a project named "REACT-EXPENSE-TRACKER" containing files like node_modules, public, src, .gitignore, package.json, README.md, and yarn.lock. Below the Explorer is the Activity Bar with sections for Outline, Timeline, and NPM Scripts. The main area is the Terminal, which displays the command "git remote -v" and its output: "origin https://github.com/adil-innovation-lab/react-expense-tracker.git (fetch)" and "origin https://github.com/adil-innovation-lab/react-expense-tracker.git (push)". The status bar at the bottom shows the branch is "master".

Step 7: Push the code from local git repository to the remote git repository

This screenshot is identical to the previous one, showing the Visual Studio Code interface with the same project structure in the Explorer, the same Activity Bar, and the same terminal output. The terminal shows the command "git push origin master" and its output, indicating the code has been successfully pushed to the remote repository.

The screenshot shows the VS Code interface with the following details:

- EXPLORER** sidebar: Shows the project structure for "REACT-EXPENSE-TRACKER" with files like node_modules, public, src, .gitignore, package.json, README.md, and yarn.lock.
- TERMINAL**: Displays the command-line output of a git push operation:

```
Adil:react-expense-tracker adil$ git push origin master
Enumerating objects: 22, done.
Counting objects: 100% (22/22), done.
Delta compression using up to 4 threads
Compressing objects: 100% (22/22), done.
Writing objects: 100% (22/22), 200.65 KiB | 5.57 MiB/s, done.
Total 22 (delta 0), reused 0 (delta 0)
To https://github.com/adil-innovation-lab/react-expense-tracker.git
 * [new branch]      master -> master
Adil:react-expense-tracker adil$
```
- DEBUG CONSOLE**, **TERMINAL**, and **...** tabs are visible at the top of the terminal area.
- STATUS BAR**: Shows the current branch as "master" and other status indicators.

Step 8: Verify changes pushed to remote git repository

The screenshot shows the GitHub repository page for "adil-innovation-lab / react-expense-tracker".

- Header:** Shows the repository URL: github.com/adil-innovation-lab/react-expense-tracker.
- Repository Summary:** Includes metrics: 1 commit, 1 branch, 0 packages, 0 releases, and 1 contributor.
- Alert:** A yellow box displays a security vulnerability message: "⚠ We found a potential security vulnerability in one of your dependencies. Only the owner of this repository can see this message." with a "View Dependabot alert" button.
- Branch Selection:** Set to "master".
- Actions:** Buttons for "New pull request", "Create new file", "Upload files", "Find file", and "Clone or download".
- Commit History:** Shows the initial project setup using Create React App, with commits for public, src, .gitignore, README.md, package.json, and yarn.lock, all made 12 minutes ago.

Part V - Deploy React App to Surge with a GitHub Workflow

Step 1: Create surge token and add a secret to the GitHub repository

The screenshot shows the GitHub repository settings for 'adil-innovation-lab / react-expense-tracker'. The left sidebar has 'Secrets' selected. The main area is titled 'Secrets' and contains the message: 'There are no secrets for this repository. Encrypted secrets allow you to store sensitive information, such as access tokens, in your repository.' A 'New secret' button is visible in the top right.

The screenshot shows the GitHub repository settings for 'adil-innovation-lab / react-expense-tracker'. The left sidebar has 'Secrets' selected. The main area is titled 'Secrets / New secret'. It shows a 'Name' field with 'SURGE_TOKEN' and a large 'Value' field containing a single vertical bar character. A green 'Add secret' button is at the bottom.

Step 2: Go to the Actions tab in the GitHub repository on github.com and click the set up a workflow yourself link

The screenshot shows the GitHub Actions setup page for a repository named "react-expense-tracker". At the top, there's a message: "Repository secret added." Below it, the repository name "adil-innovation-lab / react-expense-tracker" is shown. The "Actions" tab is selected. A large section titled "Get started with GitHub Actions" encourages users to build, test, and deploy their code. It features a "Set up this workflow yourself" button. Below this, a section titled "Workflows made for your JavaScript repository" lists two suggested workflows: "Publish Node.js Package" and "Node.js". Each workflow has its own "Set up this workflow" button and a preview of the YAML code. A URL at the bottom of the page is: https://github.com/adil-innovation-lab/react-expense-tracker/actions/new/master?filename=.github%2Fworkflows%2Fmain.yml&workflow_template=blank.

The screenshot shows the GitHub Actions editor for the repository "react-expense-tracker". The file being edited is ".github/workflows/DeploySurge.yml". The code is as follows:

```
1 # This is a basic workflow to help you get started with Actions
2
3 name: CI
4
5 # Controls when the action will run. Triggers the workflow on push or pull request
6 # events but only for the master branch
7 on:
8   push:
9     branches: [ master ]
10    pull_request:
11      branches: [ master ]
12
13 # A workflow run is made up of one or more jobs that can run sequentially or in parallel
14 jobs:
15   # This workflow contains a single job called "build"
16   build:
17     # The type of runner that the job will run on
18     runs-on: ubuntu-latest
19
20     # Steps represent a sequence of tasks that will be executed as part of the job
21     steps:
22       # Checks-out your repository under $GITHUB_WORKSPACE, so your job can access it
```

On the right side of the editor, there are tabs for "Marketplace" and "Documentation", and a sidebar for "Featured Actions" with three items: "Setup Node.js environment", "Upload a Build Artifact", and "Setup Java JDK".

Step 3: Write the YAML for the workflow

A screenshot of a GitHub repository page for 'adil-innovation-lab/react-expense-tracker'. The page shows the 'Code' tab selected, displaying the contents of the 'DeploySurge.yml' workflow file. The file contains YAML code for a GitHub Actions workflow. To the right of the code editor, there is a sidebar titled 'Marketplace' which lists several actions available for use in workflows.

```
20  # Steps represent a sequence of tasks that will be executed as part of the job
21  steps:
22  # Checks-out your repository under $GITHUB_WORKSPACE, so your job can access it
23  - uses: actions/checkout@v2
24
25  - name: Install NodeJS
26  uses: actions/setup-node@v2-beta
27  with:
28    node-version: 12
29
30  - uses: borales/actions-yarn@v2.0.0
31  with:
32    cmd: install # will run `yarn install` command
33
34  - name: Build React App
35  run: yarn build
36
37  - name: Install Surge
38  run: npm install --global surge
39
40  - name: Deploy to Surge
41  run: surge ./build react-expense-tracker-adil.surge.sh --token ${ secrets.SURGE_TOKEN }
```

Step 4: Commit the workflow to the master branch

A screenshot of the same GitHub repository page, but now the 'Commit' section is visible at the bottom of the workflow editor. A green 'Start commit' button is prominently displayed, indicating that the changes made to the workflow file are ready to be committed to the master branch.

```
20  # Steps represent a sequence of tasks that will be executed as part of the job
21  steps:
22  # Checks-out your repository under $GITHUB_WORKSPACE, so your job can access it
23  - uses: actions/checkout@v2
24
25  - name: Install NodeJS
26  uses: actions/setup-node@v2-beta
27  with:
28    node-version: 12
29
30  - uses: borales/actions-yarn@v2.0.0
31  with:
32    cmd: install # will run `yarn install` command
33
34  - name: Build React App
35  run: yarn build
36
37  - name: Install Surge
38  run: npm install --global surge
39
40  - name: Deploy to Surge
41  run: surge ./build react-expense-tracker-adil.surge.sh --token ${ secrets.SURGE_TOKEN }
```

Step 5: Check the Actions tab for the workflow to deploy successfully

The screenshot shows the GitHub Actions tab for the repository 'adil-innovation-lab/react-expense-tracker'. The 'Actions' tab is selected. A workflow named 'Create DeploySurge.yml' is listed under 'All workflows'. The most recent run is shown, triggered by a push to the 'master' branch. The status is 'Queued' with a timestamp of '10 seconds ago'. A 'Cancel workflow' button is visible.

The screenshot shows the detailed logs for the 'React App Deploy to Surge / build' step of the workflow. The log output is as follows:

```
1 1 Run borales/actions-yarn@v2.0.0
2 2 /usr/bin/docker run --name d39060775a57ef4572b2a4f2ab0142c08d 7c4340 --label 3888d3 --workdir /github/workspace
3 3 --rm -e INPUT_CMD -e INPUT_AUTH_TOKEN -e INPUT_REGISTRY_URL -e NPM_AUTH_TOKEN -e NPM_REGISTRY_URL -e HOME -e
4 4 GITHUB_JOB -e GITHUB_REF -e GITHUB_SHA -e GITHUB_REPOSITORY -e GITHUB_REPOSITORY_OWNER -e GITHUB_RUN_ID -e
5 5 GITHUB_RUN_NUMBER -e GITHUB_ACTOR -e GITHUB_WORKFLOW -e GITHUB_HEAD_REF -e GITHUB_BASE_REF -e GITHUB_EVENT_NAME
6 6 -e GITHUB_SERVER_URL -e GITHUB_API_URL -e GITHUB_GRAPHQL_URL -e GITHUB_WORKSPACE -e GITHUB_ACTION -e
7 7 GITHUB_EVENT_PATH -e RUNNER_OS -e RUNNER_TOOL_CACHE -e RUNNER_TEMP -e RUNNER_WORKSPACE -e ACTIONS_RUNTIME_URL -e
8 8 ACTIONS_RUNTIME_TOKEN -e ACTIONS_CACHE_URL -e GITHUB_ACTIONS=true -e CI=true -v
9 9 "/var/run/docker.sock":"/var/run/docker.sock" -v "/home/runner/work/_temp/_github_home":"/github/home" -v
10 10 "/home/runner/work/_temp/_github_workflow":"/github/workflow" -v "/home/runner/work/react-expense-tracker/react-expense-tracker":"/github/workspace" 3888d3:9060775a57ef4572b...
11 11 yarn install v1.17.3
12 12 [1/4] Resolving packages...
13 13 [2/4] Fetching packages...
14 14 ...
15 15 ...
16 16 ...
17 17 ...
18 18 ...
19 19 ...
20 20 ...
21 21 ...
22 22 ...
23 23 ...
24 24 ...
25 25 ...
26 26 ...
27 27 ...
28 28 ...
29 29 ...
30 30 ...
31 31 ...
32 32 ...
33 33 ...
34 34 ...
35 35 ...
36 36 ...
37 37 ...
38 38 ...
39 39 ...
40 40 ...
41 41 ...
42 42 ...
43 43 ...
44 44 ...
45 45 ...
46 46 ...
47 47 ...
48 48 ...
49 49 ...
50 50 ...
51 51 ...
52 52 ...
53 53 ...
54 54 ...
55 55 ...
56 56 ...
57 57 ...
58 58 ...
59 59 ...
60 60 ...
61 61 ...
62 62 ...
63 63 ...
64 64 ...
65 65 ...
66 66 ...
67 67 ...
68 68 ...
69 69 ...
70 70 ...
71 71 ...
72 72 ...
73 73 ...
74 74 ...
75 75 ...
76 76 ...
77 77 ...
78 78 ...
79 79 ...
80 80 ...
81 81 ...
82 82 ...
83 83 ...
84 84 ...
85 85 ...
86 86 ...
87 87 ...
88 88 ...
89 89 ...
90 90 ...
91 91 ...
92 92 ...
93 93 ...
94 94 ...
95 95 ...
96 96 ...
97 97 ...
98 98 ...
99 99 ...
100 100 ...
101 101 ...
102 102 ...
103 103 ...
104 104 ...
105 105 ...
106 106 ...
107 107 ...
108 108 ...
109 109 ...
110 110 ...
111 111 ...
112 112 ...
113 113 ...
114 114 ...
115 115 ...
116 116 ...
117 117 ...
118 118 ...
119 119 ...
120 120 ...
121 121 ...
122 122 ...
123 123 ...
124 124 ...
125 125 ...
126 126 ...
127 127 ...
128 128 ...
129 129 ...
130 130 ...
131 131 ...
132 132 ...
133 133 ...
134 134 ...
135 135 ...
136 136 ...
137 137 ...
138 138 ...
139 139 ...
140 140 ...
141 141 ...
142 142 ...
143 143 ...
144 144 ...
145 145 ...
146 146 ...
147 147 ...
148 148 ...
149 149 ...
150 150 ...
151 151 ...
152 152 ...
153 153 ...
154 154 ...
155 155 ...
156 156 ...
157 157 ...
158 158 ...
159 159 ...
160 160 ...
161 161 ...
162 162 ...
163 163 ...
164 164 ...
165 165 ...
166 166 ...
167 167 ...
168 168 ...
169 169 ...
170 170 ...
171 171 ...
172 172 ...
173 173 ...
174 174 ...
175 175 ...
176 176 ...
177 177 ...
178 178 ...
179 179 ...
180 180 ...
181 181 ...
182 182 ...
183 183 ...
184 184 ...
185 185 ...
186 186 ...
187 187 ...
188 188 ...
189 189 ...
190 190 ...
191 191 ...
192 192 ...
193 193 ...
194 194 ...
195 195 ...
196 196 ...
197 197 ...
198 198 ...
199 199 ...
200 200 ...
201 201 ...
202 202 ...
203 203 ...
204 204 ...
205 205 ...
206 206 ...
207 207 ...
208 208 ...
209 209 ...
210 210 ...
211 211 ...
212 212 ...
213 213 ...
214 214 ...
215 215 ...
216 216 ...
217 217 ...
218 218 ...
219 219 ...
220 220 ...
221 221 ...
222 222 ...
223 223 ...
224 224 ...
225 225 ...
226 226 ...
227 227 ...
228 228 ...
229 229 ...
230 230 ...
231 231 ...
232 232 ...
233 233 ...
234 234 ...
235 235 ...
236 236 ...
237 237 ...
238 238 ...
239 239 ...
240 240 ...
241 241 ...
242 242 ...
243 243 ...
244 244 ...
245 245 ...
246 246 ...
247 247 ...
248 248 ...
249 249 ...
250 250 ...
251 251 ...
252 252 ...
253 253 ...
254 254 ...
255 255 ...
256 256 ...
257 257 ...
258 258 ...
259 259 ...
260 260 ...
261 261 ...
262 262 ...
263 263 ...
264 264 ...
265 265 ...
266 266 ...
267 267 ...
268 268 ...
269 269 ...
270 270 ...
271 271 ...
272 272 ...
273 273 ...
274 274 ...
275 275 ...
276 276 ...
277 277 ...
278 278 ...
279 279 ...
280 280 ...
281 281 ...
282 282 ...
283 283 ...
284 284 ...
285 285 ...
286 286 ...
287 287 ...
288 288 ...
289 289 ...
290 290 ...
291 291 ...
292 292 ...
293 293 ...
294 294 ...
295 295 ...
296 296 ...
297 297 ...
298 298 ...
299 299 ...
300 300 ...
301 301 ...
302 302 ...
303 303 ...
304 304 ...
305 305 ...
306 306 ...
307 307 ...
308 308 ...
309 309 ...
310 310 ...
311 311 ...
312 312 ...
313 313 ...
314 314 ...
315 315 ...
316 316 ...
317 317 ...
318 318 ...
319 319 ...
320 320 ...
321 321 ...
322 322 ...
323 323 ...
324 324 ...
325 325 ...
326 326 ...
327 327 ...
328 328 ...
329 329 ...
330 330 ...
331 331 ...
332 332 ...
333 333 ...
334 334 ...
335 335 ...
336 336 ...
337 337 ...
338 338 ...
339 339 ...
340 340 ...
341 341 ...
342 342 ...
343 343 ...
344 344 ...
345 345 ...
346 346 ...
347 347 ...
348 348 ...
349 349 ...
350 350 ...
351 351 ...
352 352 ...
353 353 ...
354 354 ...
355 355 ...
356 356 ...
357 357 ...
358 358 ...
359 359 ...
360 360 ...
361 361 ...
362 362 ...
363 363 ...
364 364 ...
365 365 ...
366 366 ...
367 367 ...
368 368 ...
369 369 ...
370 370 ...
371 371 ...
372 372 ...
373 373 ...
374 374 ...
375 375 ...
376 376 ...
377 377 ...
378 378 ...
379 379 ...
380 380 ...
381 381 ...
382 382 ...
383 383 ...
384 384 ...
385 385 ...
386 386 ...
387 387 ...
388 388 ...
389 389 ...
390 390 ...
391 391 ...
392 392 ...
393 393 ...
394 394 ...
395 395 ...
396 396 ...
397 397 ...
398 398 ...
399 399 ...
400 400 ...
401 401 ...
402 402 ...
403 403 ...
404 404 ...
405 405 ...
406 406 ...
407 407 ...
408 408 ...
409 409 ...
410 410 ...
411 411 ...
412 412 ...
413 413 ...
414 414 ...
415 415 ...
416 416 ...
417 417 ...
418 418 ...
419 419 ...
420 420 ...
421 421 ...
422 422 ...
423 423 ...
424 424 ...
425 425 ...
426 426 ...
427 427 ...
428 428 ...
429 429 ...
430 430 ...
431 431 ...
432 432 ...
433 433 ...
434 434 ...
435 435 ...
436 436 ...
437 437 ...
438 438 ...
439 439 ...
440 440 ...
441 441 ...
442 442 ...
443 443 ...
444 444 ...
445 445 ...
446 446 ...
447 447 ...
448 448 ...
449 449 ...
450 450 ...
451 451 ...
452 452 ...
453 453 ...
454 454 ...
455 455 ...
456 456 ...
457 457 ...
458 458 ...
459 459 ...
460 460 ...
461 461 ...
462 462 ...
463 463 ...
464 464 ...
465 465 ...
466 466 ...
467 467 ...
468 468 ...
469 469 ...
470 470 ...
471 471 ...
472 472 ...
473 473 ...
474 474 ...
475 475 ...
476 476 ...
477 477 ...
478 478 ...
479 479 ...
480 480 ...
481 481 ...
482 482 ...
483 483 ...
484 484 ...
485 485 ...
486 486 ...
487 487 ...
488 488 ...
489 489 ...
490 490 ...
491 491 ...
492 492 ...
493 493 ...
494 494 ...
495 495 ...
496 496 ...
497 497 ...
498 498 ...
499 499 ...
500 500 ...
501 501 ...
502 502 ...
503 503 ...
504 504 ...
505 505 ...
506 506 ...
507 507 ...
508 508 ...
509 509 ...
510 510 ...
511 511 ...
512 512 ...
513 513 ...
514 514 ...
515 515 ...
516 516 ...
517 517 ...
518 518 ...
519 519 ...
520 520 ...
521 521 ...
522 522 ...
523 523 ...
524 524 ...
525 525 ...
526 526 ...
527 527 ...
528 528 ...
529 529 ...
530 530 ...
531 531 ...
532 532 ...
533 533 ...
534 534 ...
535 535 ...
536 536 ...
537 537 ...
538 538 ...
539 539 ...
540 540 ...
541 541 ...
542 542 ...
543 543 ...
544 544 ...
545 545 ...
546 546 ...
547 547 ...
548 548 ...
549 549 ...
550 550 ...
551 551 ...
552 552 ...
553 553 ...
554 554 ...
555 555 ...
556 556 ...
557 557 ...
558 558 ...
559 559 ...
560 560 ...
561 561 ...
562 562 ...
563 563 ...
564 564 ...
565 565 ...
566 566 ...
567 567 ...
568 568 ...
569 569 ...
570 570 ...
571 571 ...
572 572 ...
573 573 ...
574 574 ...
575 575 ...
576 576 ...
577 577 ...
578 578 ...
579 579 ...
580 580 ...
581 581 ...
582 582 ...
583 583 ...
584 584 ...
585 585 ...
586 586 ...
587 587 ...
588 588 ...
589 589 ...
590 590 ...
591 591 ...
592 592 ...
593 593 ...
594 594 ...
595 595 ...
596 596 ...
597 597 ...
598 598 ...
599 599 ...
600 600 ...
601 601 ...
602 602 ...
603 603 ...
604 604 ...
605 605 ...
606 606 ...
607 607 ...
608 608 ...
609 609 ...
610 610 ...
611 611 ...
612 612 ...
613 613 ...
614 614 ...
615 615 ...
616 616 ...
617 617 ...
618 618 ...
619 619 ...
620 620 ...
621 621 ...
622 622 ...
623 623 ...
624 624 ...
625 625 ...
626 626 ...
627 627 ...
628 628 ...
629 629 ...
630 630 ...
631 631 ...
632 632 ...
633 633 ...
634 634 ...
635 635 ...
636 636 ...
637 637 ...
638 638 ...
639 639 ...
640 640 ...
641 641 ...
642 642 ...
643 643 ...
644 644 ...
645 645 ...
646 646 ...
647 647 ...
648 648 ...
649 649 ...
650 650 ...
651 651 ...
652 652 ...
653 653 ...
654 654 ...
655 655 ...
656 656 ...
657 657 ...
658 658 ...
659 659 ...
660 660 ...
661 661 ...
662 662 ...
663 663 ...
664 664 ...
665 665 ...
666 666 ...
667 667 ...
668 668 ...
669 669 ...
670 670 ...
671 671 ...
672 672 ...
673 673 ...
674 674 ...
675 675 ...
676 676 ...
677 677 ...
678 678 ...
679 679 ...
680 680 ...
681 681 ...
682 682 ...
683 683 ...
684 684 ...
685 685 ...
686 686 ...
687 687 ...
688 688 ...
689 689 ...
690 690 ...
691 691 ...
692 692 ...
693 693 ...
694 694 ...
695 695 ...
696 696 ...
697 697 ...
698 698 ...
699 699 ...
700 700 ...
701 701 ...
702 702 ...
703 703 ...
704 704 ...
705 705 ...
706 706 ...
707 707 ...
708 708 ...
709 709 ...
710 710 ...
711 711 ...
712 712 ...
713 713 ...
714 714 ...
715 715 ...
716 716 ...
717 717 ...
718 718 ...
719 719 ...
720 720 ...
721 721 ...
722 722 ...
723 723 ...
724 724 ...
725 725 ...
726 726 ...
727 727 ...
728 728 ...
729 729 ...
730 730 ...
731 731 ...
732 732 ...
733 733 ...
734 734 ...
735 735 ...
736 736 ...
737 737 ...
738 738 ...
739 739 ...
740 740 ...
741 741 ...
742 742 ...
743 743 ...
744 744 ...
745 745 ...
746 746 ...
747 747 ...
748 748 ...
749 749 ...
750 750 ...
751 751 ...
752 752 ...
753 753 ...
754 754 ...
755 755 ...
756 756 ...
757 757 ...
758 758 ...
759 759 ...
760 760 ...
761 761 ...
762 762 ...
763 763 ...
764 764 ...
765 765 ...
766 766 ...
767 767 ...
768 768 ...
769 769 ...
770 770 ...
771 771 ...
772 772 ...
773 773 ...
774 774 ...
775 775 ...
776 776 ...
777 777 ...
778 778 ...
779 779 ...
780 780 ...
781 781 ...
782 782 ...
783 783 ...
784 784 ...
785 785 ...
786 786 ...
787 787 ...
788 788 ...
789 789 ...
790 790 ...
791 791 ...
792 792 ...
793 793 ...
794 794 ...
795 795 ...
796 796 ...
797 797 ...
798 798 ...
799 799 ...
800 800 ...
801 801 ...
802 802 ...
803 803 ...
804 804 ...
805 805 ...
806 806 ...
807 807 ...
808 808 ...
809 809 ...
810 810 ...
811 811 ...
812 812 ...
813 813 ...
814 814 ...
815 815 ...
816 816 ...
817 817 ...
818 818 ...
819 819 ...
820 820 ...
821 821 ...
822 822 ...
823 823 ...
824 824 ...
825 825 ...
826 826 ...
827 827 ...
828 828 ...
829 829 ...
830 830 ...
831 831 ...
832 832 ...
833 833 ...
834 834 ...
835 835 ...
836 836 ...
837 837 ...
838 838 ...
839 839 ...
840 840 ...
841 841 ...
842 842 ...
843 843 ...
844 844 ...
845 845 ...
846 846 ...
847 847 ...
848 848 ...
849 849 ...
850 850 ...
851 851 ...
852 852 ...
853 853 ...
854 854 ...
855 855 ...
856 856 ...
857 857 ...
858 858 ...
859 859 ...
860 860 ...
861 861 ...
862 862 ...
863 863 ...
864 864 ...
865 865 ...
866 866 ...
867 867 ...
868 868 ...
869 869 ...
870 870 ...
871 871 ...
872 872 ...
873 873 ...
874 874 ...
875 875 ...
876 876 ...
877 877 ...
878 878 ...
879 879 ...
880 880 ...
881 881 ...
882 882 ...
883 883 ...
884 884 ...
885 885 ...
886 886 ...
887 887 ...
888 888 ...
889 889 ...
890 890 ...
891 891 ...
892 892 ...
893 893 ...
894 894 ...
895 895 ...
896 896 ...
897 897 ...
898 898 ...
899 899 ...
900 900 ...
901 901 ...
902 902 ...
903 903 ...
904 904 ...
905 905 ...
906 906 ...
907 907 ...
908 908 ...
909 909 ...
910 910 ...
911 911 ...
912 912 ...
913 913 ...
914 914 ...
915 915 ...
916 916 ...
917 917 ...
918 918 ...
919 919 ...
920 920 ...
921 921 ...
922 922 ...
923 923 ...
924 924 ...
925 925 ...
926 926 ...
927 927 ...
928 928 ...
929 929 ...
930 930 ...
931 931 ...
932 932 ...
933 933 ...
934 934 ...
935 935 ...
936 936 ...
937 937 ...
938 938 ...
939 939 ...
940 940 ...
941 941 ...
942 942 ...
943 943 ...
944 944 ...
945 945 ...
946 946 ...
947 947 ...
948 948 ...
949 949 ...
950 950 ...
951 951 ...
952 952 ...
953 953 ...
954 954 ...
955 955 ...
956 956 ...
957 957 ...
958 958 ...
959 959 ...
960 960 ...
961 961 ...
962 962 ...
963 963 ...
964 964 ...
965 965 ...
966 966 ...
967 967 ...
968 968 ...
969 969 ...
970 970 ...
971 971 ...
972 972 ...
973 973 ...
974 974 ...
975 975 ...
976 976 ...
977 977 ...
978 978 ...
979 979 ...
980 980 ...
981 981 ...
982 982 ...
983 983 ...
984 984 ...
985 985 ...
986 986 ...
987 987 ...
988 988 ...
989 989 ...
990 990 ...
991 991 ...
992 992 ...
993 993 ...
994 994 ...
995 995 ...
996 996 ...
997 997 ...
998 998 ...
999 999 ...
1000 1000 ...
1001 1001 ...
1002 1002 ...
1003 1003 ...
1004 1004 ...
1005 1005 ...
1006 1006 ...
1007 1007 ...
1008 1008 ...
1009 1009 ...
1010 1010 ...
1011 1011 ...
1012 1012 ...
1013 1013 ...
1014 1014 ...
1015 1015 ...
1016 1016 ...
1017 1017 ...
1018 1018 ...
1019 1019 ...
1020 1020 ...
1021 1021 ...
1022 1022 ...
1023 1023 ...
1024 1024 ...
1025 1025 ...
1026 1026 ...
1027 1027 ...
1028 1028 ...
1029 1029 ...
1030 1030 ...
1031 1031 ...
1032 1032 ...
1033 1033 ...
1034 1034 ...
1035 1035 ...
1036 1036 ...
1037 1037 ...
1038 1038 ...
1039 1039 ...
1040 1040 ...
1041 1041 ...
1042 1042 ...
1043 1043 ...
1044 1044 ...
1045 1045 ...
1046 1046 ...
1047 1047 ...
1048 1048 ...
1049 1049 ...
1050 1050 ...
1051 1051 ...
1052 1052 ...
1053 1053 ...
1054 1054 ...
1055 1055 ...
1056 1056 ...
1057 1057 ...
1058 1058 ...
1059 1059 ...
1060 1060 ...
1061 1061 ...
1062 1062 ...
1063 1063 ...
1064 1064 ...
1065 1065 ...
1066 1066 ...
1067 1067 ...
1068 1068 ...
1069 1069 ...
1070 1070 ...
1071 1071 ...
1072 1072 ...
1073 1073 ...
1074 1074 ...
1075 1075 ...
1076 1076 ...
1077 1077 ...
1078 1078 ...
1079 1079 ...
1080 1080 ...
1081 1081 ...
1082 1082 ...
1083 1083 ...
1084 1084 ...
1085 1085 ...
1086 1086 ...
1087 1087 ...
1088 1088 ...
1089 1089 ...
1090 1090 ...
1091 1091 ...
1092 1092 ...
1093 1093 ...
1094 1094 ...
1095 1095 ...
1096 1096 ...
1097 1097 ...
1098 1098 ...
1099 1099 ...
1100 1100 ...
1101 1101 ...
1102 1102 ...
1103 1103 ...
1104 1104 ...
1105 1105 ...
1106 1106 ...
1107 1107 ...
1108 1108 ...
1109 1109 ...
1110 1110 ...
1111 1111 ...
1112 1112 ...
1113 1113 ...
1114 1114 ...
1115 1115 ...
1116 1116 ...
1117 1117 ...
1118 1118 ...
1119 1119 ...
1120 1120 ...
1121 1121 ...
1122 1122 ...
1123 1123 ...
1124 1124 ...
1125 1125 ...
1126 1126 ...
1127 1127 ...
1128 1128 ...
1129 1129 ...
1130 1130 ...
1131 1131 ...
1132 1132 ...
1133 1133 ...
1134 1134 ...
1135 1135 ...
1136 1136 ...
1137 1137 ...
1138 1138 ...
1139 1139 ...
1140 1140 ...
1141 1141 ...
1142 1142 ...
1143 1143 ...
1144 1144 ...
1145 1145 ...
1146 1146 ...
1147 1147 ...
1148 1148 ...
1149 1149 ...
1150 1150 ...
1151 1151 ...
1152 1152 ...
1153 1153 ...
1154 1154 ...
1155 1155 ...
1156 1156 ...
1157 1157 ...
1158 1158 ...
1159 1159 ...
1160 1160 ...
1161 1161 ...
1162 1162 ...
1163 1163 ...
1164 1164 ...
1165 1165 ...
1166 1166 ...
1167 1167 ...
1168 1168 ...
1169 1169 ...
1170 1170 ...
1171 1171 ...
1172 1172 ...
1173 1173 ...
1174 1174 ...
1175 1175 ...
1176 1176 ...
1177 1177 ...
1178 1178 ...
1179 1179 ...
1180 1180 ...
1181 1181 ...
1182 1182 ...
1183 1183 ...
1184 1184 ...
1185 1185 ...
1186 1186 ...
1187 1187 ...
1188 1188 ...
1189 1189 ...
1190 1190 ...
1191 1191 ...
1192 1192 ...
1193 1193 ...
1194 1194 ...
1195 1195 ...
1196 1196 ...
1197 1197 ...
1198 1198 ...
1199 1199 ...
1200 1200 ...
1201 1201 ...
1202 1202 ...
1203 1203 ...
1204 1204 ...
1205 1205 ...
1206 1206 ...
1207 1207 ...
1208 1208 ...
1209 1209 ...
1210 1210 ...
1211 1211 ...
1212 1212 ...
1213 1213 ...
1214 1214 ...
1215 1215 ...
1216 1216 ...
1217 1217 ...
1218 1218 ...
1219 1219 ...
1220 1220 ...
1221 1221 ...
1222 1222 ...
1223 1223 ...
1224 1224 ...
1225 1225 ...
1226 1226 ...
1227 1227 ...
1228 1228 ...
1229 1229 ...
1230 1230 ...
1231 1231 ...
1232 1232 ...
1233 1233 ...
1234 1234 ...
1235 1235 ...
1236 1236 ...
1237 1237 ...
1238 1238 ...
1239 1239 ...
1240 1240 ...
1241 1241 ...
1242 1242 ...
1243 1243 ...
1244 1244 ...
1245 1245 ...
1246 1246 ...
1247 1247 ...
1248 1248 ...
1249 1249 ...
1250 1250 ...
1251 1251 ...
1252 1252 ...
1253 1253 ...
1254 1254 ...
1255 1255 ...
1256 1256 ...
1257 1257 ...
1258 1258 ...
1259 1259 ...
1260 1260 ...
1261 1261 ...
1262 1262 ...
1263 1263 ...
1264 1264 ...
1265 1265 ...
1266 1266 ...
1267 1267 ...
1268 1268 ...
1269 1269 ...
1270 1270 ...
1271 1271 ...
1272 1272 ...
1273 1273 ...
1274 1274 ...
1275 1275 ...
1276 1276 ...
1277 1277 ...
1278 1278 ...
1279 1279 ...
1280 1280 ...
1281 1281 ...
1282 1282 ...
1283 1283 ...
1284 1284 ...
1285 1285 ...
1286 1286 ...
1287 1287 ...
1288 1288 ...
1289 1289 ...
1290 1290 ...
1291 1291 ...
1292 1292 ...
1293 1293 ...
1294 1294 ...
1295 1295 ...
1296 1296 ...
12
```

github.com/adil-innovation-lab/react-expense-tracker/runs/770647906

Search or jump to... Pull requests Issues Marketplace Explore

adil-innovation-lab / react-expense-tracker

Code Issues 0 Pull requests 1 Actions Projects 0 Wiki Security 1 Insights Settings

Create DeploySurge.yml

master a2030fa

Re-run jobs

React App Deploy to Surge / build succeeded now in 1m 1s

Search logs

build

- Set up job 4s
- Build borales/actions-yarn@v2.0.0 6s
- Run actions/checkout@v2 0s
- Install NodeJS 1s
- Run borales/actions-yarn@v2.0.0 33s
- Build React App 8s
- Install Surge 6s
- Deploy to Surge 3s
- Post Run actions/checkout@v2 0s
- Complete job 0s

Part VI - Create Components

Step 1: Create a components folder in the src folder

The screenshot shows the VS Code interface with the following details:

- EXPLORER View:** Shows the project structure. A new folder named "components" has been created under the "src" directory.
- Editor View:** The file "App.js" is open, displaying the following code:

```
1 import React from 'react';
2 import './App.css';
3
4 function App() {
5   return (
6     <div>
7       | Hello World
8     </div>
9   );
10
11
12 export default App;
13
```
- Terminal View:** Shows the command "4: node" and the message "You can now view react-expense-tracker in the browser." It also displays the local and network URLs for the application.
- Bottom Status Bar:** Shows the current branch as "master*", file count as 0, and other status indicators.

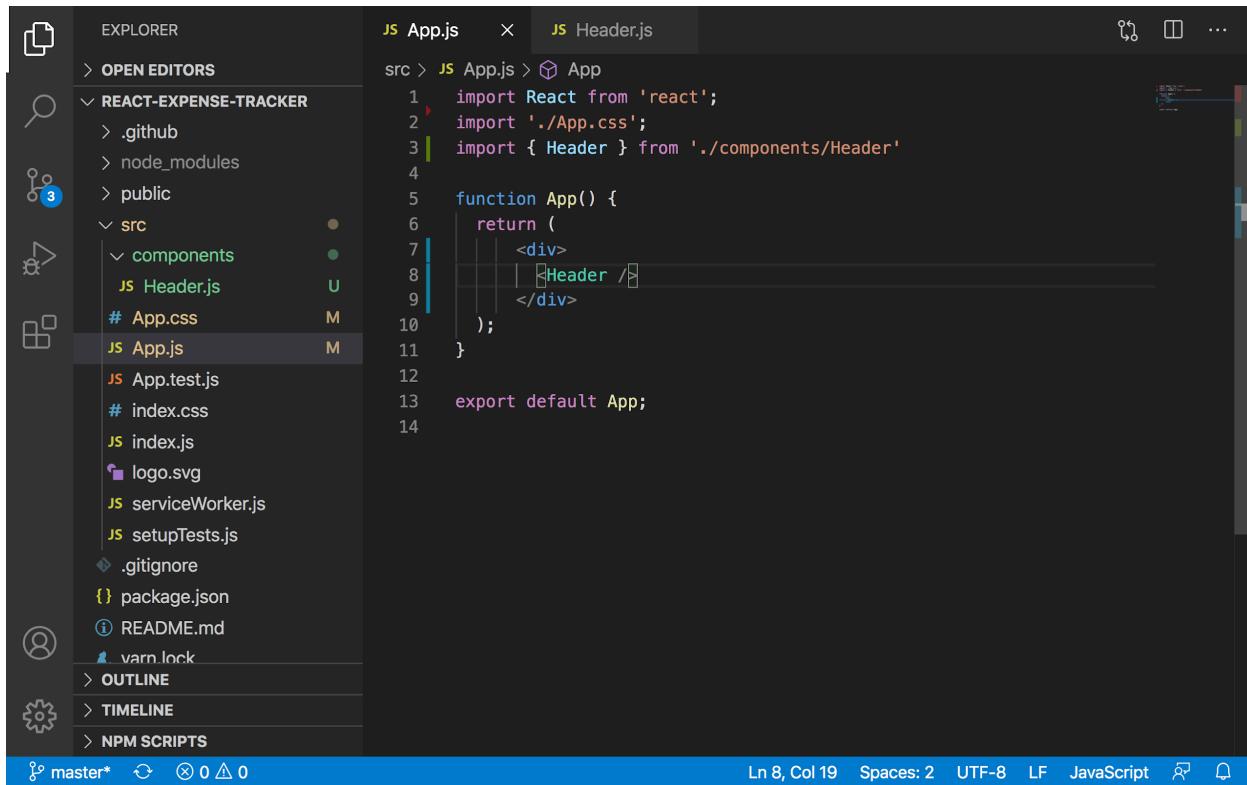
Step 2: Create Header Component

The screenshot shows the VS Code interface with the following details:

- EXPLORER View:** Shows the project structure. The "components" folder now contains a file named "Header.js".
- Editor View:** The file "Header.js" is open, displaying the following code:

```
1 import React from 'react';
2
3 export const Header = () => {
4   return (
5     <h1>
6       | React Expense Tracker by Adil Altaf
7     </h1>
8   );
9 }
```
- Bottom Status Bar:** Shows the current branch as "master*", file count as 0, and other status indicators.

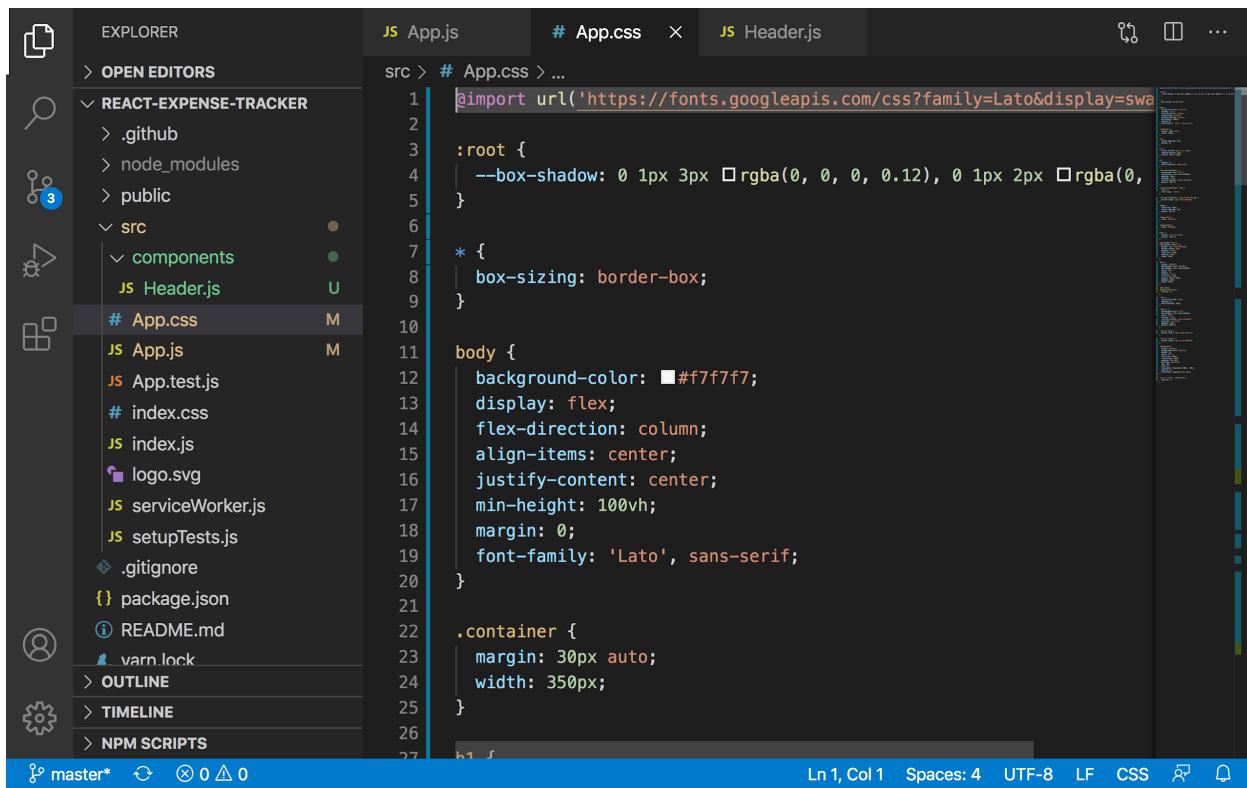
Step 3: Import the Header Component to App.js and add to App function



The screenshot shows the VS Code interface with the Explorer sidebar open, displaying the project structure of 'REACT-EXPENSE-TRACKER'. The 'src' folder contains 'components' (with 'Header.js') and other files like 'App.css', 'App.js', and 'index.css'. The 'App.js' file is selected in the Explorer and is being edited in the main editor area. The code imports 'React' and 'Header' from 'components/Header'. It defines an 'App' function that returns a div containing a Header component. The status bar at the bottom indicates the file is on 'master*' branch, has 0 changes, and is in 'JavaScript' mode.

```
src > JS App.js > App
1 import React from 'react';
2 import './App.css';
3 import { Header } from './components/Header'
4
5 function App() {
6   return (
7     <div>
8       |   <Header />
9     </div>
10  );
11}
12
13 export default App;
```

Step 4: Apply CSS to App.js from App.css



The screenshot shows the VS Code interface with the Explorer sidebar open, displaying the project structure of 'REACT-EXPENSE-TRACKER'. The 'src' folder contains 'components' (with 'Header.js') and other files like 'App.css', 'App.js', and 'index.css'. The 'App.css' file is selected in the Explorer and is being edited in the main editor area. The code imports a Google font CSS file. It defines a root selector with a box shadow and a body selector with a light gray background, flex display, and center alignment. It also defines a container selector with auto margin and a fixed width. The status bar at the bottom indicates the file is on 'master*' branch, has 0 changes, and is in 'CSS' mode.

```
src > # App.css > ...
1 @import url('https://fonts.googleapis.com/css?family=Lato&display=swap');
2
3 :root {
4   --box-shadow: 0 1px 3px 0 rgba(0, 0, 0, 0.12), 0 1px 2px 0 rgba(0, 0, 0, 0.08);
5 }
6
7 *
8   box-sizing: border-box;
9
10 body {
11   background-color: #f7f7f7;
12   display: flex;
13   flex-direction: column;
14   align-items: center;
15   justify-content: center;
16   min-height: 100vh;
17   margin: 0;
18   font-family: 'Lato', sans-serif;
19 }
20
21 .container {
22   margin: 30px auto;
23   width: 350px;
24 }
```

Step 5: Create Balance Component

The screenshot shows the VS Code interface with the following details:

- EXPLORER** sidebar: Shows the project structure under `REACT-EXPENSE-TRACKER`, including `src` and `components` folders containing `Balance.js`, `Header.js`, and `App.css`.
- OPEN EDITORS**: `App.js` is the active editor.
- Code Editor**: Displays the `Balance.js` file content:

```
1 import React from 'react'
2
3 export const Balance = () => {
4   return (
5     <div>
6       <h4>Current Balance</h4>
7       <h1 id="balance">$0.00</h1>
8     </div>
9   }
10 }
11 
```
- Bottom Status Bar**: Shows the file is 11 lines long, in Col 1, with 4 spaces, using UTF-8 encoding, and is a JavaScript file.

Step 6: Import the Balance Component to App.js and add to App function

The screenshot shows the VS Code interface with the following details:

- EXPLORER** sidebar: Shows the project structure under `REACT-EXPENSE-TRACKER`, including `src` and `components` folders containing `Balance.js`, `Header.js`, and `App.css`.
- OPEN EDITORS**: `App.js` is the active editor.
- Code Editor**: Displays the `App.js` file content:

```
1 import React from 'react';
2 import './App.css';
3 import { Header } from './components/Header';
4 import { Balance } from './components/Balance';
5
6 function App() {
7   return (
8     <div>
9       <Header />
10      <div className="container">
11        <Balance />
12      </div>
13    );
14 }
15
16
17 export default App;
18 
```
- Bottom Status Bar**: Shows the file is 18 lines long, in Col 1, with 2 spaces, using UTF-8 encoding, and is a JavaScript file.

Step 7: Create the AccountSummary Component for Income and Expense Summary

The screenshot shows the VS Code interface with the following details:

- EXPLORER** sidebar: Shows the project structure under `REACT-EXPENSE-TRACKER`, including `src` and `components` folders containing `AccountSummary.js`, `Balance.js`, and `Header.js`.
- EDITOR**: Two tabs are open: `App.js` and `AccountSummary.js`. The `AccountSummary.js` tab is active, displaying the following code:

```
1 import React from 'react'
2
3 export const AccountSummary = () => {
4     return (
5         <div className="inc-exp-container">
6             <div>
7                 <h4>Income</h4>
8                 <p className="money plus">
9                     +$0.00
10                </p>
11            </div>
12            <div>
13                <h4>Expense</h4>
14                <p className="money minus">
15                    -$0.00
16                </p>
17            </div>
18        </div>
19    )
20}
21
```

- STATUS BAR**: Shows the current branch is `master*`, and the file status is `U`.

Step 8: Import the AccountSummary Component to App.js and add to App function

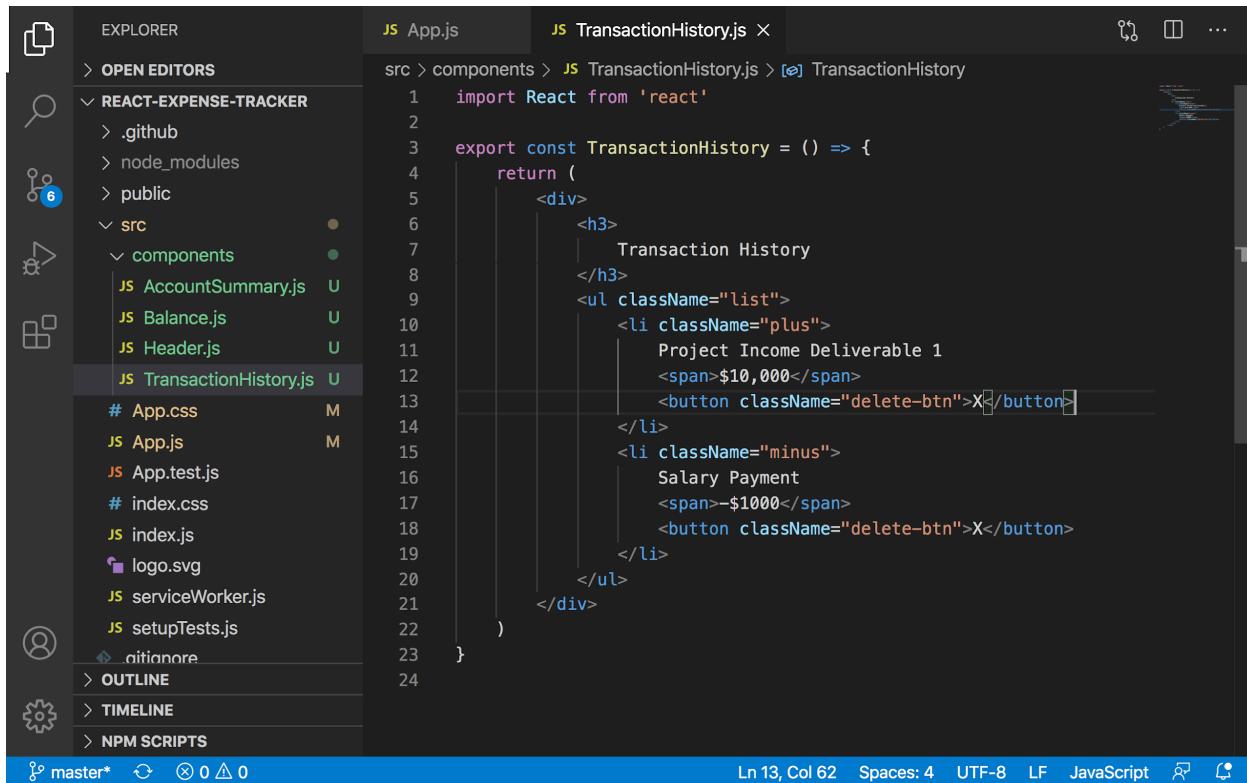
The screenshot shows the VS Code interface with the following details:

- EXPLORER** sidebar: Shows the project structure under `REACT-EXPENSE-TRACKER`, including `src` and `components` folders containing `AccountSummary.js`, `Balance.js`, and `Header.js`.
- EDITOR**: Two tabs are open: `App.js` and `AccountSummary.js`. The `App.js` tab is active, displaying the following code:

```
1 import React from 'react';
2 import './App.css';
3 import { Header } from './components/Header';
4 import { Balance } from './components/Balance';
5 import { AccountSummary } from './components/AccountSummary';
6
7 function App() {
8     return (
9         <div>
10            <Header />
11            <div className="container">
12                <Balance />
13                <AccountSummary />
14            </div>
15        </div>
16    );
17 }
18
19 export default App;
```

- STATUS BAR**: Shows the current branch is `master*`, and the file status is `M`.

Step 9: Create the TransactionHistory Component for list of transactions

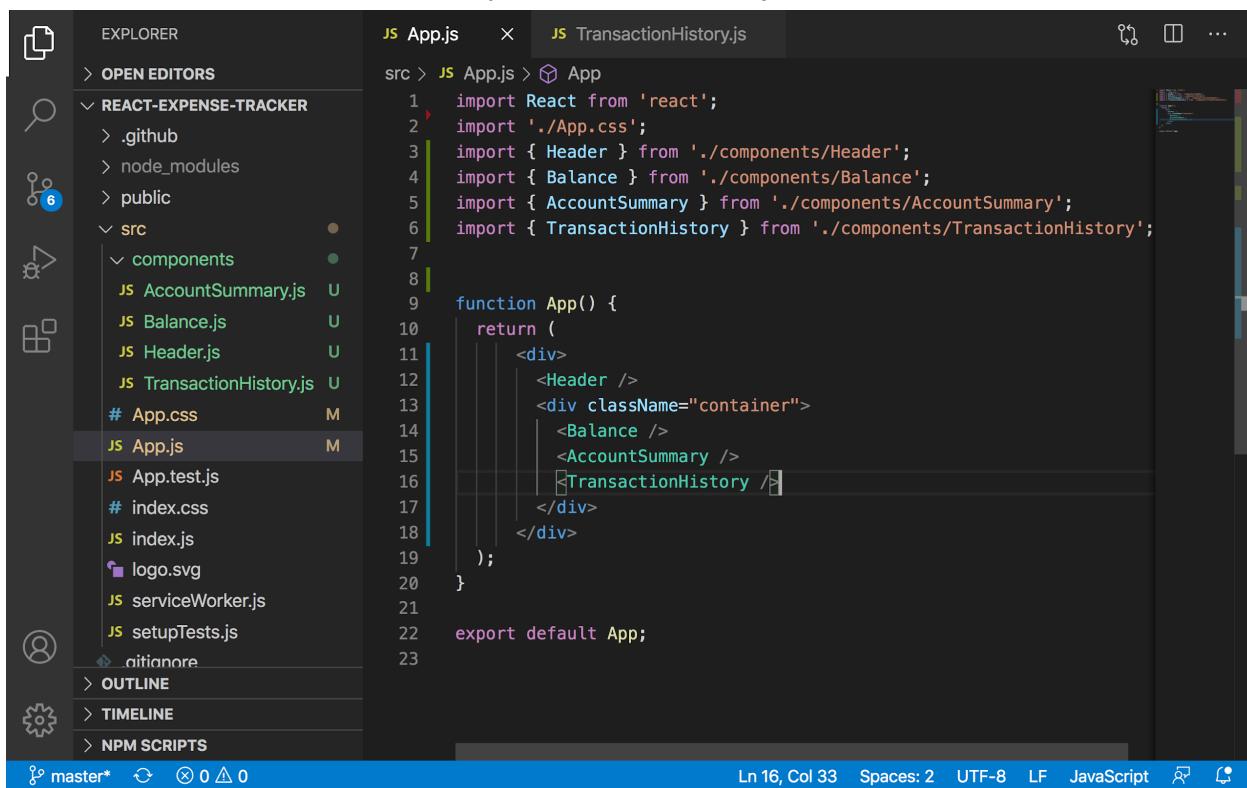


The screenshot shows the VS Code interface with the TransactionHistory.js file open in the editor. The code defines a functional component named TransactionHistory that returns a div containing a h3 and a ul with two li items. Each li item has a span and a button.

```
1 import React from 'react'
2
3 export const TransactionHistory = () => {
4     return (
5         <div>
6             <h3>
7                 | Transaction History
8             </h3>
9             <ul className="list">
10                <li className="plus">
11                    | Project Income Deliverable 1
12                    <span>$10,000</span>
13                    <button className="delete-btn">X</button>
14                </li>
15                <li className="minus">
16                    | Salary Payment
17                    <span>-$1000</span>
18                    <button className="delete-btn">X</button>
19                </li>
20            </ul>
21        </div>
22    )
23}
```

Ln 13, Col 62 Spaces: 4 UTF-8 LF JavaScript

Step 10: Import the TransactionHistory Component to App.js and add to App function

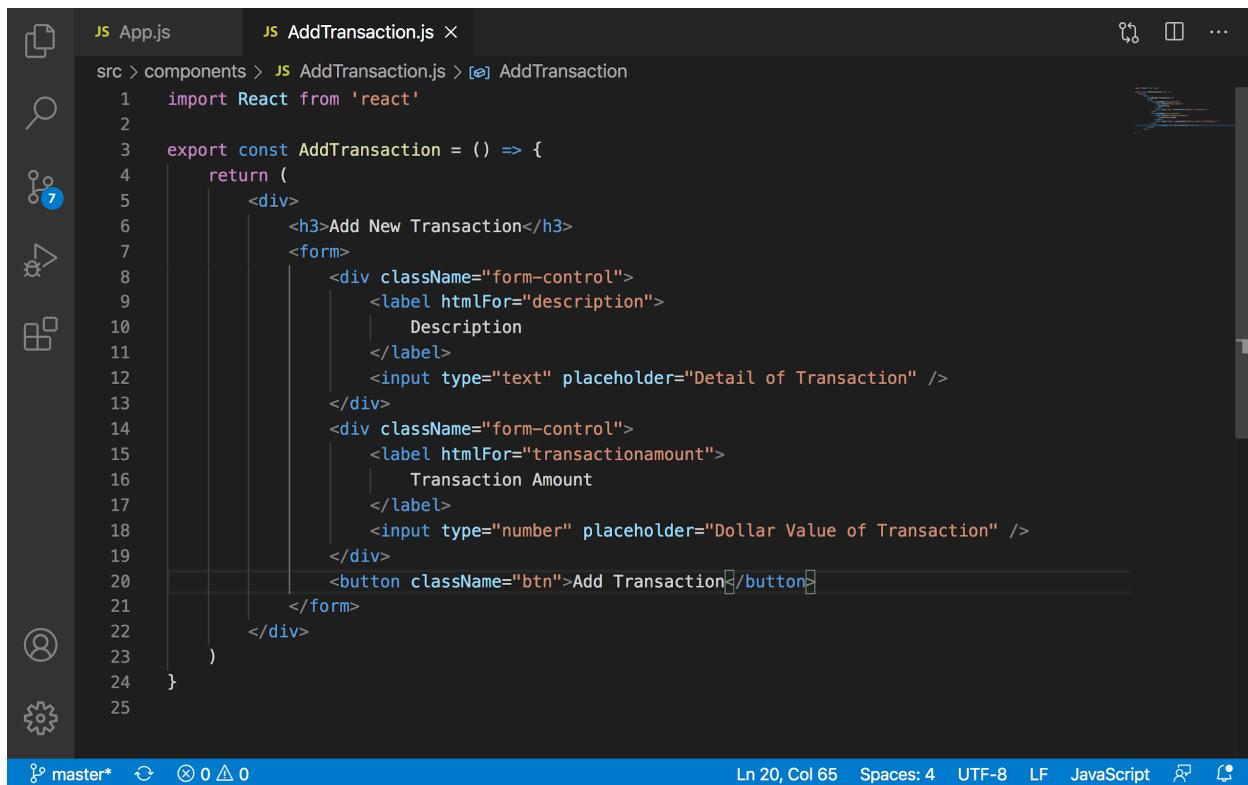


The screenshot shows the VS Code interface with the App.js file open in the editor. The code imports several components and defines an App function that returns a div containing Header, Balance, AccountSummary, and TransactionHistory components.

```
1 import React from 'react';
2 import './App.css';
3 import { Header } from './components/Header';
4 import { Balance } from './components/Balance';
5 import { AccountSummary } from './components/AccountSummary';
6 import { TransactionHistory } from './components/TransactionHistory';
7
8 function App() {
9     return (
10         <div>
11             <Header />
12             <div className="container">
13                 <Balance />
14                 <AccountSummary />
15                 <TransactionHistory />
16             </div>
17         </div>
18     );
19 }
20
21 export default App;
```

Ln 16, Col 33 Spaces: 2 UTF-8 LF JavaScript

Step 11: Create the AddTransaction Component for the form to add new transactions

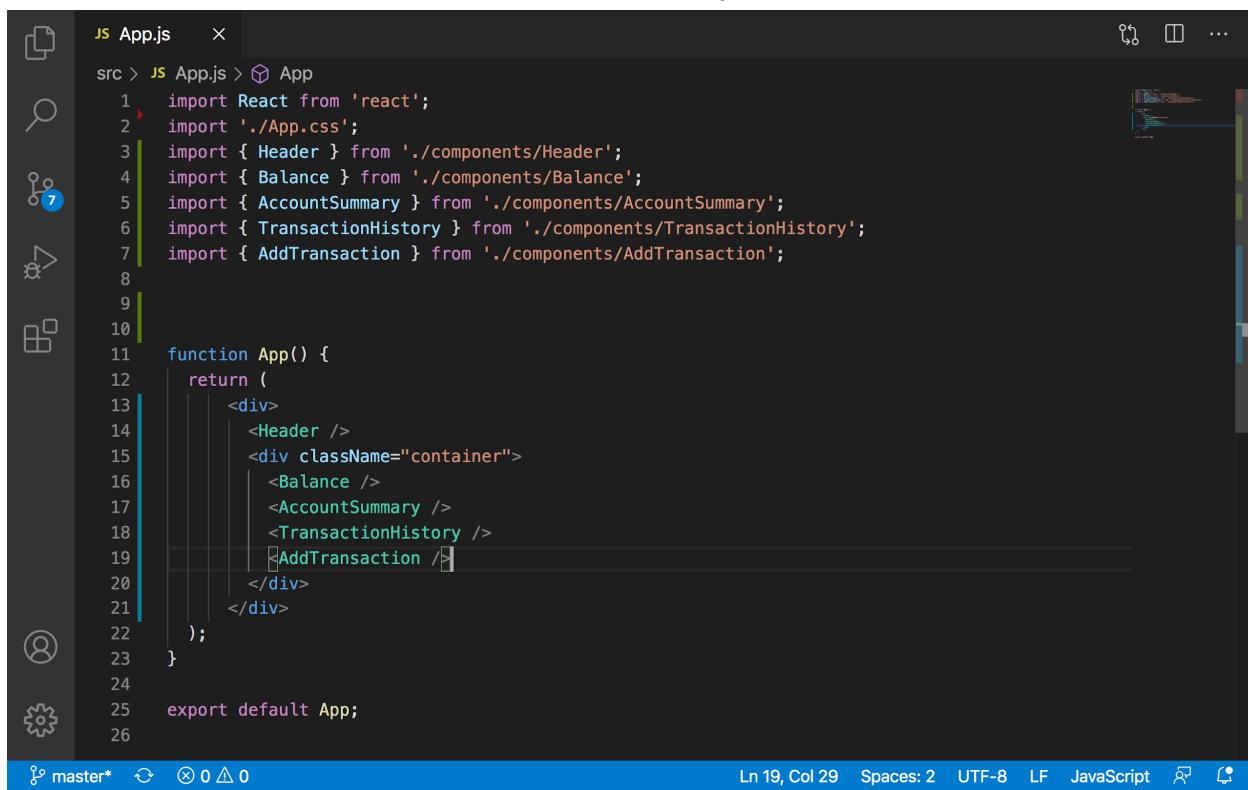


The screenshot shows the VS Code interface with the 'AddTransaction.js' file open. The code defines a functional component named 'AddTransaction' that creates a form for adding a new transaction. It includes fields for description and transaction amount, and a button to add the transaction.

```
JS App.js      JS AddTransaction.js ×
src > components > JS AddTransaction.js > [o] AddTransaction
1 import React from 'react'
2
3 export const AddTransaction = () => {
4     return (
5         <div>
6             <h3>Add New Transaction</h3>
7             <form>
8                 <div className="form-control">
9                     <label htmlFor="description">
10                         Description
11                     </label>
12                     <input type="text" placeholder="Detail of Transaction" />
13                 </div>
14                 <div className="form-control">
15                     <label htmlFor="transactionamount">
16                         Transaction Amount
17                     </label>
18                     <input type="number" placeholder="Dollar Value of Transaction" />
19                 </div>
20                 <button className="btn">Add Transaction</button>
21             </form>
22         </div>
23     )
24 }
25

Ln 20, Col 65  Spaces: 4  UTF-8  LF  JavaScript  ⚡  🔍
```

Step 12: Import the AddTransaction Component to App.js and add to App function



The screenshot shows the 'App.js' file in VS Code. The 'AddTransaction' component is imported and added to the 'App' function's return statement, rendering it within the main application container.

```
JS App.js ×
src > JS App.js > [o] App
1 import React from 'react';
2 import './App.css';
3 import { Header } from './components/Header';
4 import { Balance } from './components/Balance';
5 import { AccountSummary } from './components/AccountSummary';
6 import { TransactionHistory } from './components/TransactionHistory';
7 import { AddTransaction } from './components/AddTransaction';
8
9
10 function App() {
11     return (
12         <div>
13             <Header />
14             <div className="container">
15                 <Balance />
16                 <AccountSummary />
17                 <TransactionHistory />
18                 <AddTransaction />
19             </div>
20         </div>
21     );
22 }
23
24 export default App;

Ln 19, Col 29  Spaces: 2  UTF-8  LF  JavaScript  ⚡  🔍
```

Part VII - Create Hook for Updating State with Add Transaction

Step 1: Import useState

```
src > components > JS AddTransaction.js > ...
1 import React, { useState } from 'react'
2
3 export const AddTransaction = () => {
4     return (
5         <div>
6             <h3>Add New Transaction</h3>
7             <form>
8                 <div className="form-control">
9                     <label htmlFor="description">
10                         Description
11                     </label>
12                     <input type="text" placeholder="Detail of Transaction" />
13                 </div>
14                 <div className="form-control">
15                     <label htmlFor="transactionamount">
16                         Transaction Amount
17                     </label>
18                     <input type="number" placeholder="Dollar Value of Transaction" />
19                 </div>
20                 <button className="btn">Add Transaction</button>
21             </form>
22         </div>
23     )
24 }
25
```

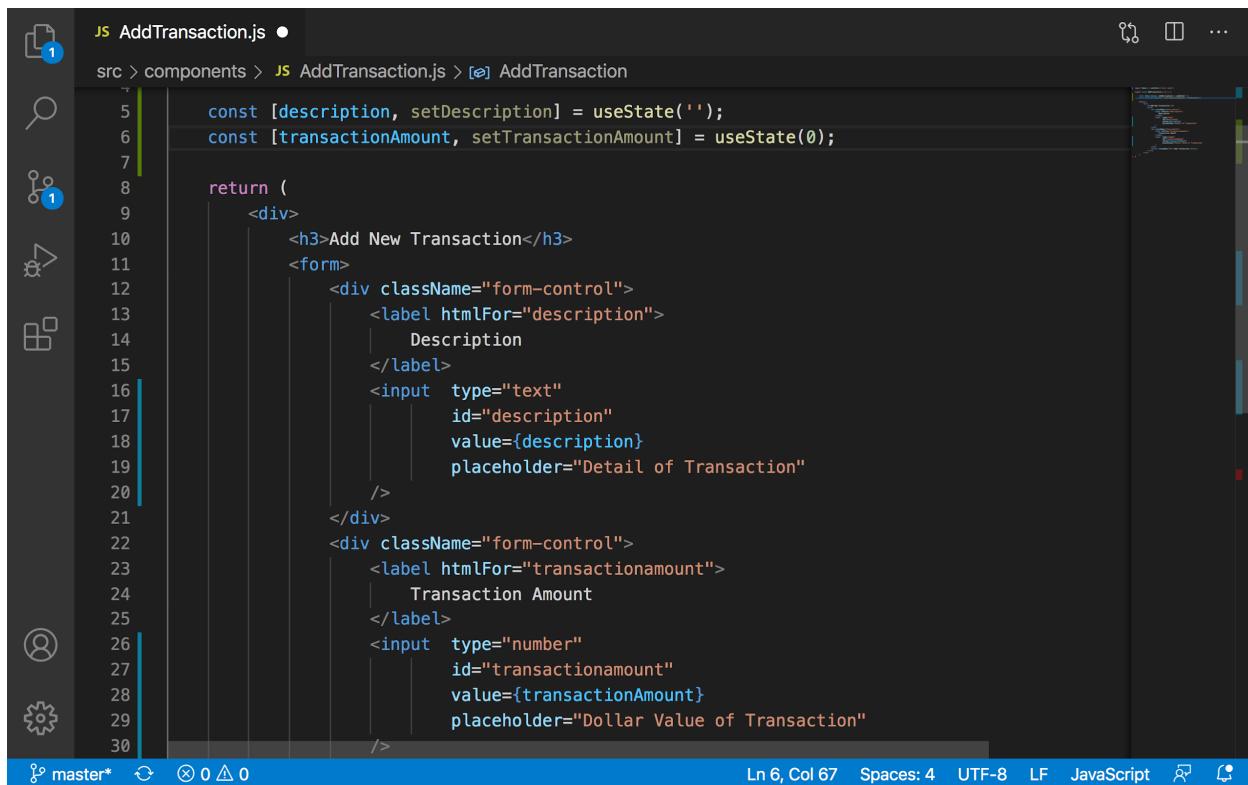
Step 2: Create state for description and transactionAmount

The screenshot shows the VS Code interface with the following details:

- Title Bar:** JS AddTransaction.js
- Sidebar:** Shows icons for file, search, navigation, and other development tools.
- Status Bar:** Displays the file path: src > components > JS AddTransaction.js > [e] AddTransaction.
- Code Area:** The code for the AddTransaction component is displayed. It uses React's useState hook to manage state for description and transactionAmount. The component returns a form with a title and a text input field labeled "Description".

```
1 import React, { useState } from 'react';
2
3 export const AddTransaction = () => [
4
5     const [description, setDescription] = useState('');
6     const [transactionAmount, setTransactionAmount] = useState(0);
7
8     return (
9         <div>
10            <h3>Add New Transaction</h3>
11            <form>
12                <div className="form-control">
13                    <label htmlFor="description">
14                        Description
15                    </label>
16                </div>
17            </form>
18        </div>
19    );
20}
```

Step 3: Link the state to the form with the value attribute

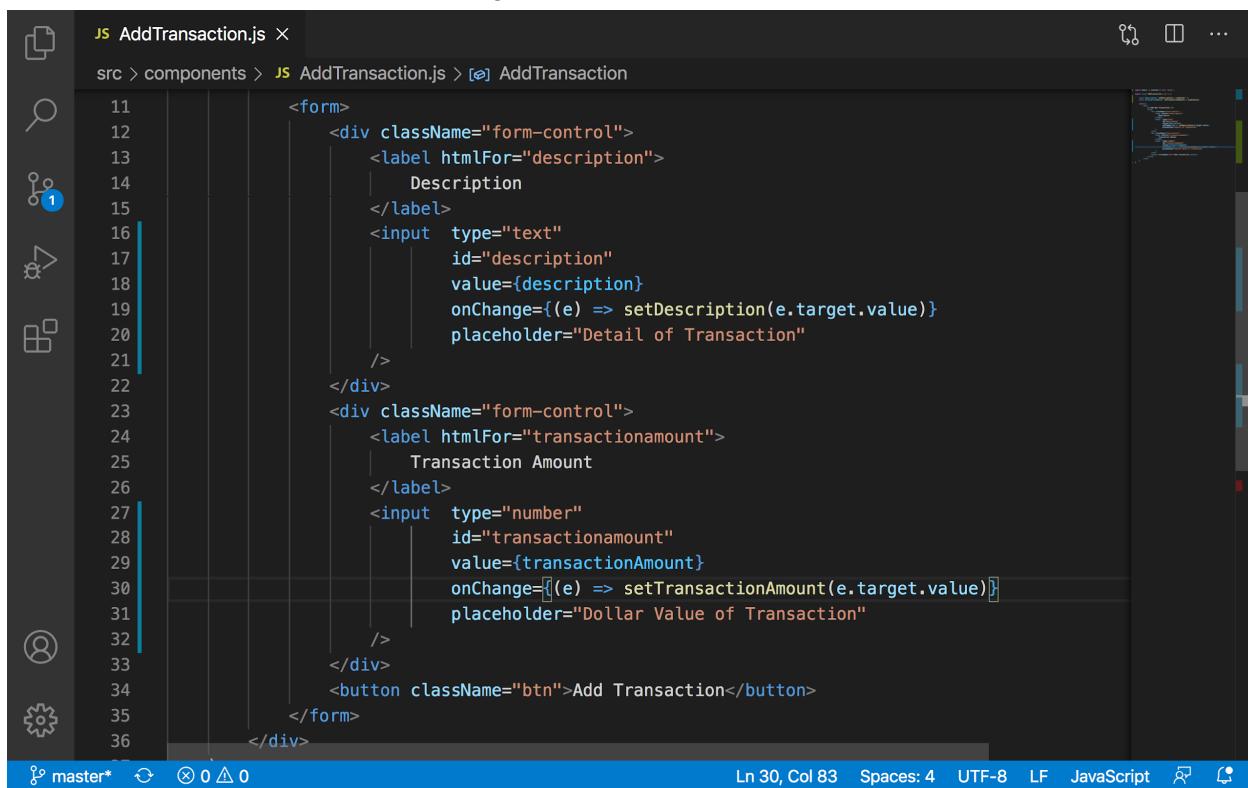


The screenshot shows the `AddTransaction.js` file in VS Code. The code defines two state variables: `description` and `transactionAmount`, both initialized to empty strings. It then creates a form with two input fields. The first input field has its `value` attribute set to the current value of `description`. The second input field has its `value` attribute set to the current value of `transactionAmount`.

```
JS AddTransaction.js ●
src > components > JS AddTransaction.js > [o] AddTransaction
  5 |   const [description, setDescription] = useState('');
  6 |   const [transactionAmount, setTransactionAmount] = useState(0);
  7 |
  8 |   return (
  9 |     <div>
10 |       <h3>Add New Transaction</h3>
11 |       <form>
12 |         <div className="form-control">
13 |           <label htmlFor="description">
14 |             Description
15 |           </label>
16 |           <input type="text"
17 |             id="description"
18 |             value={description}
19 |             placeholder="Detail of Transaction"
20 |           />
21 |         </div>
22 |         <div className="form-control">
23 |           <label htmlFor="transactionamount">
24 |             Transaction Amount
25 |           </label>
26 |           <input type="number"
27 |             id="transactionamount"
28 |             value={transactionAmount}
29 |             placeholder="Dollar Value of Transaction"
30 |           />
31 |         </div>
32 |       </form>
33 |     </div>
34 |   </div>
35 |
36 | 
```

Ln 6, Col 67 Spaces: 4 UTF-8 LF JavaScript

Step 4: Update the state with onChange event



The screenshot shows the `AddTransaction.js` file in VS Code after adding event listeners. The first input field now has an `onChange` event handler that updates the `description` state to the new value. Similarly, the second input field has an `onChange` event handler that updates the `transactionAmount` state.

```
JS AddTransaction.js ×
src > components > JS AddTransaction.js > [o] AddTransaction
  11 |   <form>
  12 |     <div className="form-control">
  13 |       <label htmlFor="description">
  14 |         Description
  15 |       </label>
  16 |       <input type="text"
  17 |         id="description"
  18 |         value={description}
  19 |         onChange={(e) => setDescription(e.target.value)}
  20 |         placeholder="Detail of Transaction"
  21 |       />
  22 |     </div>
  23 |     <div className="form-control">
  24 |       <label htmlFor="transactionamount">
  25 |         Transaction Amount
  26 |       </label>
  27 |       <input type="number"
  28 |         id="transactionamount"
  29 |         value={transactionAmount}
  30 |         onChange={(e) => setTransactionAmount(e.target.value)}
  31 |         placeholder="Dollar Value of Transaction"
  32 |       />
  33 |     </div>
  34 |     <button className="btn">Add Transaction</button>
  35 |
  36 |   </div>
  37 | 
```

Ln 30, Col 83 Spaces: 4 UTF-8 LF JavaScript

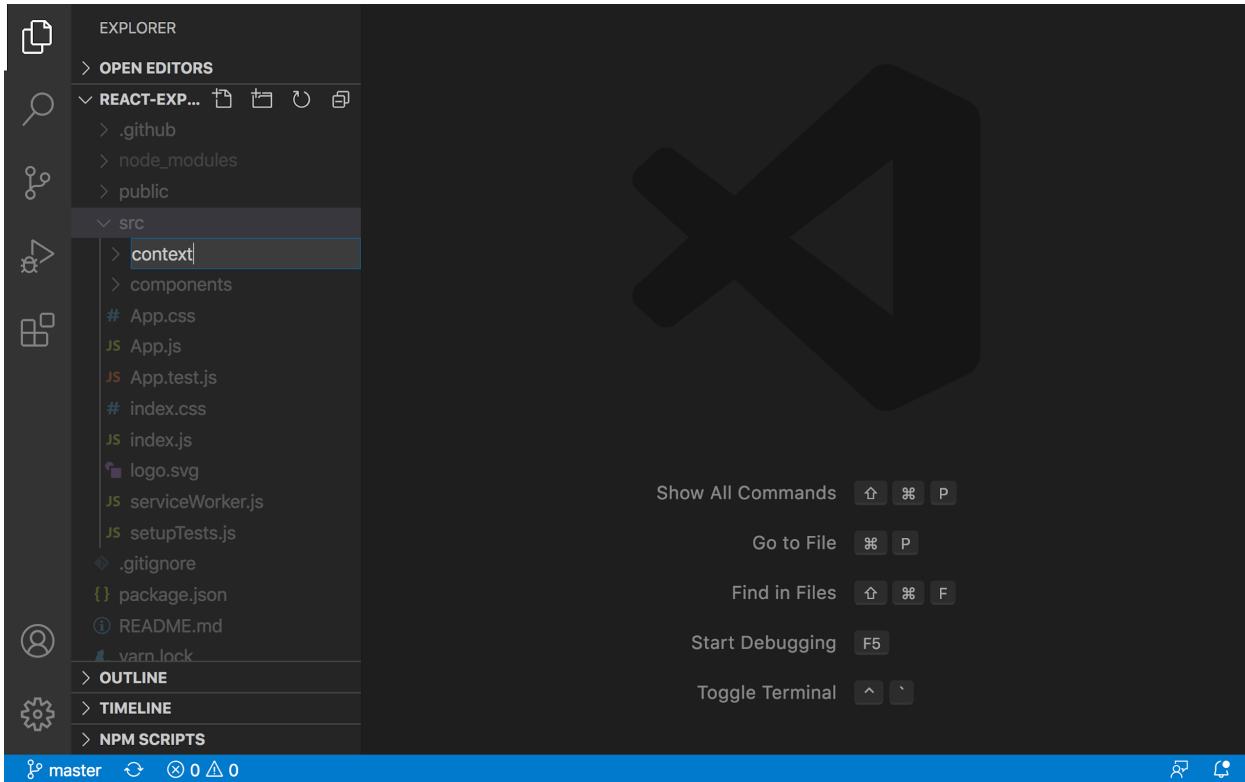
Step 5: Check the React Developer Tools in Chrome to see Components and check State

The screenshot shows a browser window with the title "React App" at "localhost:3000". The page displays an "Expense Tracker by Adil Altaf" application. On the left, there's a "CURRENT BALANCE" section showing "\$0.00". Below it is a summary box with "INCOME" (\$+\$0.00) and "EXPENSE" (-\$0.00). The main area shows a "Transaction History" table with two rows: "Project Income Deliverable 1" (\$10,000) and "Salary Payment" (-\$1000). At the bottom, there's an "Add New Transaction" form with fields for "Description" ("Project Income Deliverable 2") and "Transaction Amount" (10000), followed by a purple "Add Transaction" button.

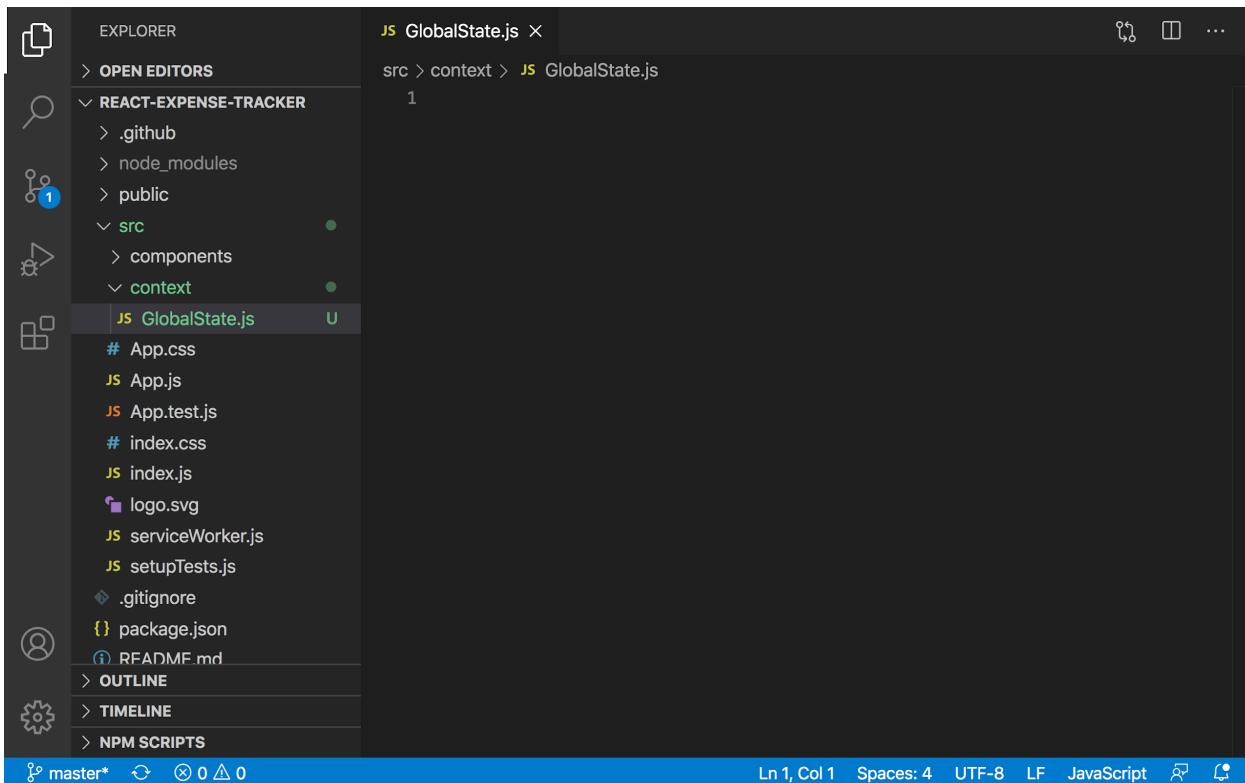
The right side of the screenshot shows the "Components" tab in the React DevTools. It lists components under the "App" tree: Header, Balance, AccountSummary, TransactionHistory, and AddTransaction. The "AddTransaction" component is currently selected. The DevTools pane shows its props (new prop: ""), hooks (State: "Project Income Deliverable 2", State: "10000"), and the fact that it is rendered by the "App" component.

Part VIII - Create Context and Reducer

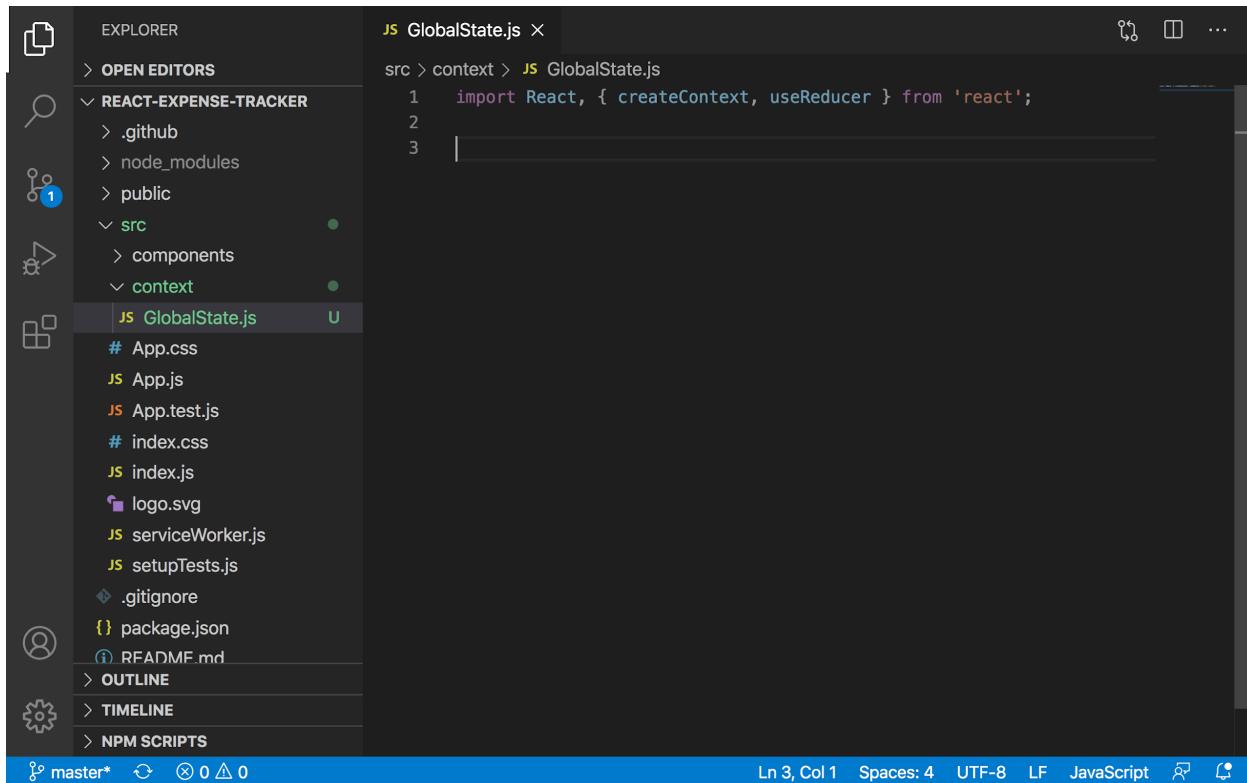
Step 1: Create a context folder in the src folder



Step 2: Create GlobalState.js file for the context

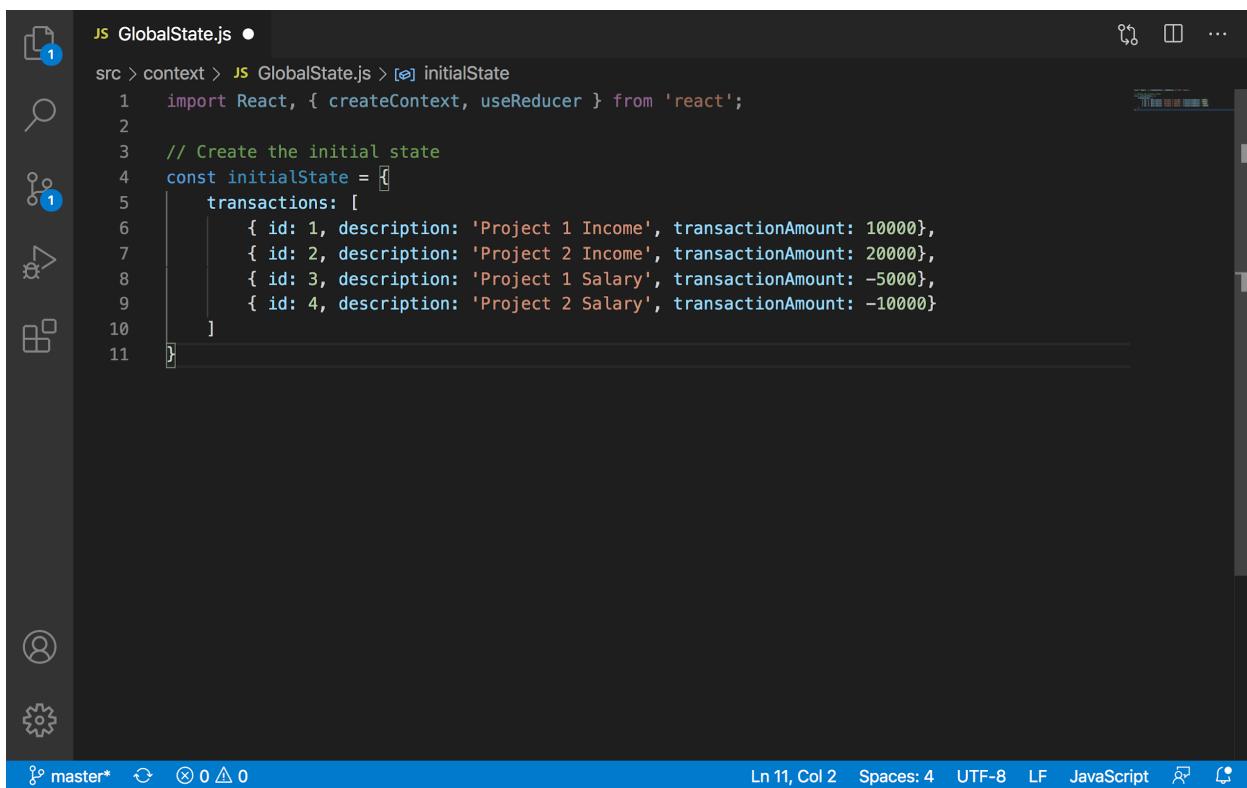


Step 3: Import React, { createContext, useReducer } from 'react'



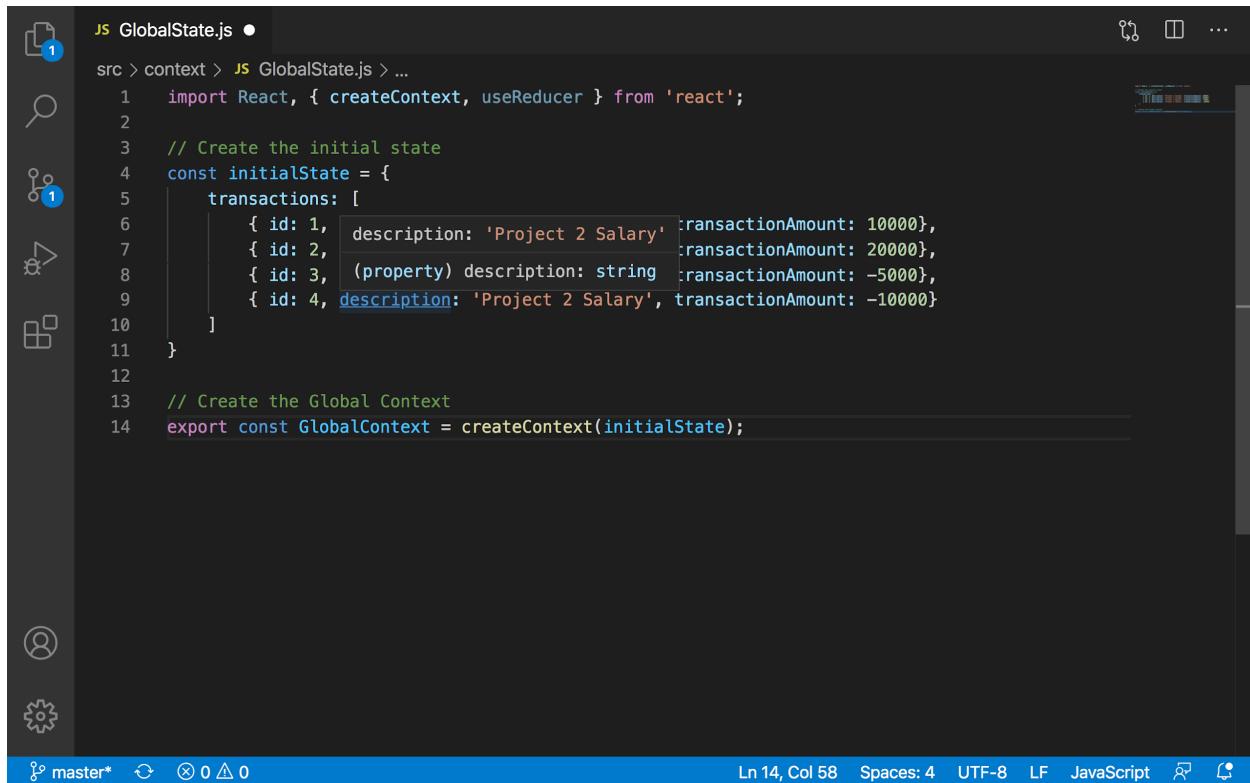
```
JS GlobalState.js ×
src > context > JS GlobalState.js
1 import React, { createContext, useReducer } from 'react';
2
3
```

Step 4: Create the Initial State with placeholder transactions



```
JS GlobalState.js ●
src > context > JS GlobalState.js > [o] initialState
1 import React, { createContext, useReducer } from 'react';
2
3 // Create the initial state
4 const initialState = [
5   transactions: [
6     { id: 1, description: 'Project 1 Income', transactionAmount: 10000},
7     { id: 2, description: 'Project 2 Income', transactionAmount: 20000},
8     { id: 3, description: 'Project 1 Salary', transactionAmount: -5000},
9     { id: 4, description: 'Project 2 Salary', transactionAmount: -10000}
10   ]
11 }
```

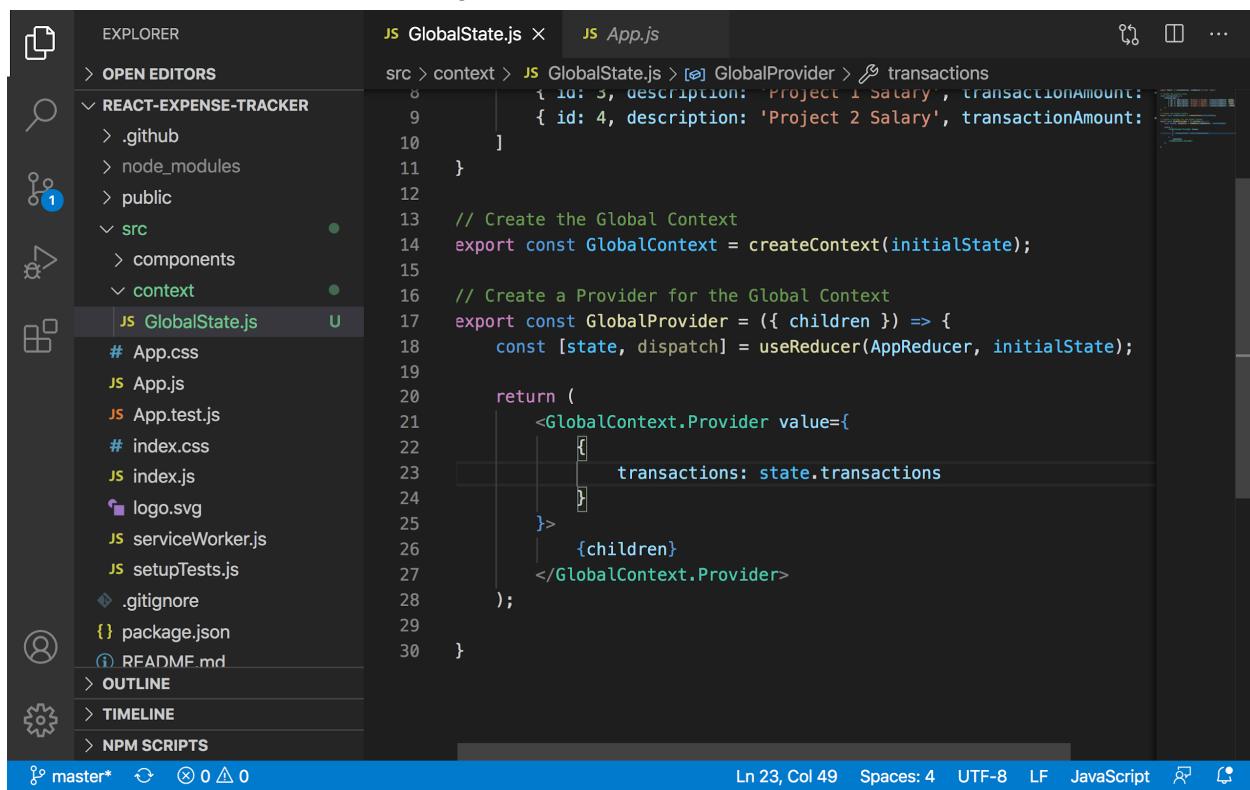
Step 5: Create the global context with the initialState as the default value:



The screenshot shows the GlobalState.js file in VS Code. The code defines a global context with an initial state containing four transactions. The transactions are represented as objects with properties like id, description, and transactionAmount.

```
JS GlobalState.js ●
src > context > JS GlobalState.js > ...
1 import React, { createContext, useReducer } from 'react';
2
3 // Create the initial state
4 const initialState = {
5   transactions: [
6     { id: 1, description: 'Project 1 Salary', transactionAmount: 10000 },
7     { id: 2, description: 'Project 2 Salary', transactionAmount: 20000 },
8     { id: 3, (property) description: string transactionAmount: -5000 },
9     { id: 4, description: 'Project 2 Salary', transactionAmount: -10000 }
10  ]
11 }
12
13 // Create the Global Context
14 export const GlobalContext = createContext(initialState);
```

Step 6: Create the provider for the global context



The screenshot shows the GlobalState.js and App.js files in the VS Code Explorer. The GlobalState.js file contains the implementation of the GlobalProvider component, which wraps its children in the GlobalContext.Provider. The App.js file imports the GlobalProvider and uses it to wrap the rest of the application's components.

```
EXPLORER
OPEN EDITORS
REACT-EXPENSE-TRACKER
  .github
  node_modules
  public
  src
    components
    context
      JS GlobalState.js
      # App.css
      JS App.js
      JS App.test.js
      # index.css
      JS index.js
      logo.svg
      JS serviceWorker.js
      JS setupTests.js
      .gitignore
      package.json
      RFADMF.md
  OUTLINE
  TIMELINE
  NPM SCRIPTS

JS GlobalState.js X JS App.js
src > context > JS GlobalState.js > [e] GlobalProvider > transactions
  1 id: 1, description: 'Project 1 Salary', transactionAmount: 10000
  2   {
  3     id: 2, description: 'Project 2 Salary', transactionAmount: 20000
  4   }
  5   {
  6     id: 3, (property) description: string transactionAmount: -5000
  7   }
  8   {
  9     id: 4, description: 'Project 2 Salary', transactionAmount: -10000
10  }
11 }
12
13 // Create the Global Context
14 export const GlobalContext = createContext(initialState);
15
16 // Create a Provider for the Global Context
17 export const GlobalProvider = ({ children }) => {
18   const [state, dispatch] = useReducer(AppReducer, initialState);
19
20   return (
21     <GlobalContext.Provider value={

      [
        {
          transactions: state.transactions
        }
      ]
    }
  )
22   >
23     {children}
24   </GlobalContext.Provider>
25 );
26
27
28
29
30 }
```

Step 7: Create the AppReducer.js file in the context folder

The screenshot shows the VS Code interface with the following details:

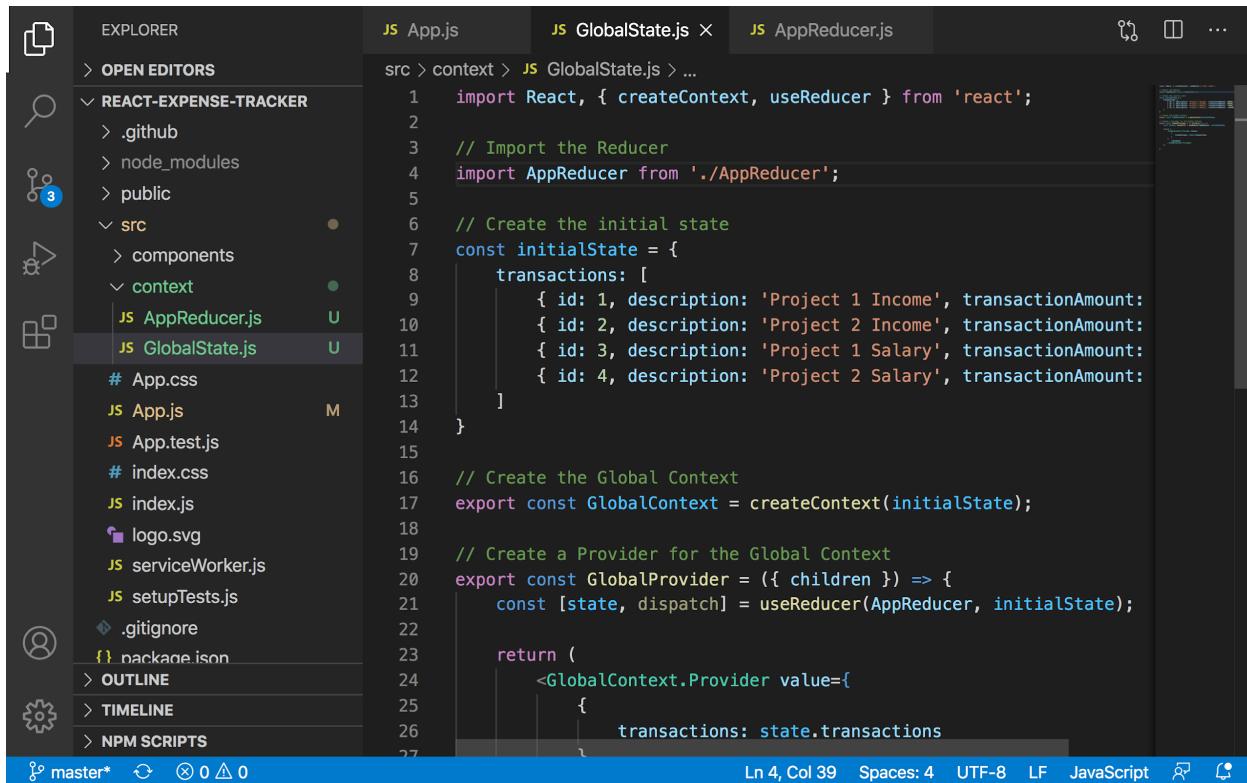
- Explorer View:** Shows the project structure under "REACT-EXPENSE-TRACKER". The "src" folder contains "components" and "context". Inside "context", there are files: "AppReducer.js" (selected), "GlobalState.js", "# App.css", "App.js", "App.test.js", "# index.css", "index.js", "logo.svg", "serviceWorker.js", and "setupTests.js". There are also ".gitignore" and "package.json" files.
- Editor View:** The file "AppReducer.js" is open. The code is currently empty: "1 |".
- Bottom Status Bar:** Shows "Ln 1, Col 1" and "Spaces: 4" and "UTF-8 LF JavaScript".

Step 8: Create the basic reducer to use action.type to return a the state if no action is performed

The screenshot shows the VS Code interface with the following details:

- Explorer View:** Same as Step 7, showing the project structure and files.
- Editor View:** The file "AppReducer.js" is open. The code starts with a default export and a switch statement for action type:1 export default (state, action) => {
2 switch(action.type) {
3 default:
4 return state;
5 }
6 }
- Bottom Status Bar:** Shows "Ln 6, Col 2" and "Spaces: 4" and "UTF-8 LF JavaScript".

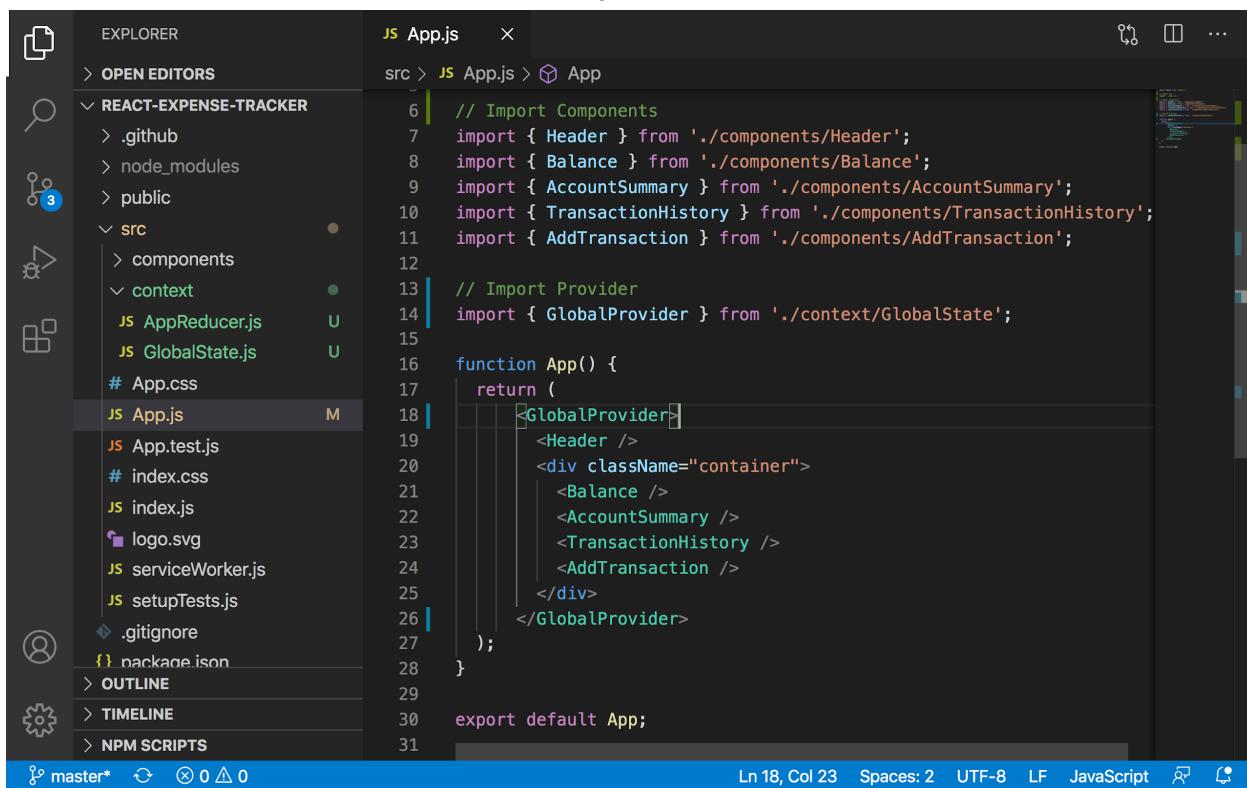
Step 9: Import the AppReducer into the GlobalState.js file



```
src > context > JS GlobalState.js > ...
1 import React, { createContext, useReducer } from 'react';
2
3 // Import the Reducer
4 import AppReducer from './AppReducer';
5
6 // Create the initial state
7 const initialState = {
8     transactions: [
9         { id: 1, description: 'Project 1 Income', transactionAmount: 100 },
10        { id: 2, description: 'Project 2 Income', transactionAmount: 200 },
11        { id: 3, description: 'Project 1 Salary', transactionAmount: 500 },
12        { id: 4, description: 'Project 2 Salary', transactionAmount: 400 }
13    ]
14 }
15
16 // Create the Global Context
17 export const GlobalContext = createContext(initialState);
18
19 // Create a Provider for the Global Context
20 export const GlobalProvider = ({ children }) => {
21     const [state, dispatch] = useReducer(AppReducer, initialState);
22
23     return (
24         <GlobalContext.Provider value={

25             {
26                 transactions: state.transactions
27             }
28         }
29     );
30 }
31
32 export default GlobalProvider;
```

Step 10: Import the GlobalProvider to the App.js and wrap components with the GlobalProvider



```
src > JS App.js > App
1 // Import Components
2 import { Header } from './components/Header';
3 import { Balance } from './components/Balance';
4 import { AccountSummary } from './components/AccountSummary';
5 import { TransactionHistory } from './components/TransactionHistory';
6 import { AddTransaction } from './components/AddTransaction';
7
8 // Import Provider
9 import { GlobalProvider } from './context/GlobalState';
10
11 function App() {
12     return (
13         <GlobalProvider>
14             <Header />
15             <div className="container">
16                 <Balance />
17                 <AccountSummary />
18                 <TransactionHistory />
19                 <AddTransaction />
20             </div>
21         </GlobalProvider>
22     );
23 }
24
25 export default App;
```

Step 11: Check the React Components in the Browser to see if reducer is present in the hooks and you can see the transactions from the initialState

The screenshot shows a browser window with two tabs both titled "React App". The URL is "localhost:3000". The main content area displays the "Expense Tracker by Adil Altaf" application. It shows a "CURRENT BALANCE" of "\$0.00". Below it, there's a summary section with "INCOME" of "+\$0.00" and "EXPENSE" of "-\$0.00". A "Transaction History" section lists "Project Income Deliverable 1" with a value of "\$10,000" and "Salary Payment" with a value of "-\$1000". An "Add New Transaction" form is present, with fields for "Description" containing "Detail of Transaction" and "Transaction Amount" containing "0". A purple "Add Transaction" button is at the bottom. To the right of the browser is the React DevTools interface. The "Components" tab is selected. The component tree under "App" shows the following structure:

- GlobalProvider
 - Context.Provider
 - Header
 - Balance
 - AccountSummary
 - TransactionHistory
 - AddTransaction
- GlobalProvider

Below the component tree, there are sections for "props", "hooks", and "rendered by".

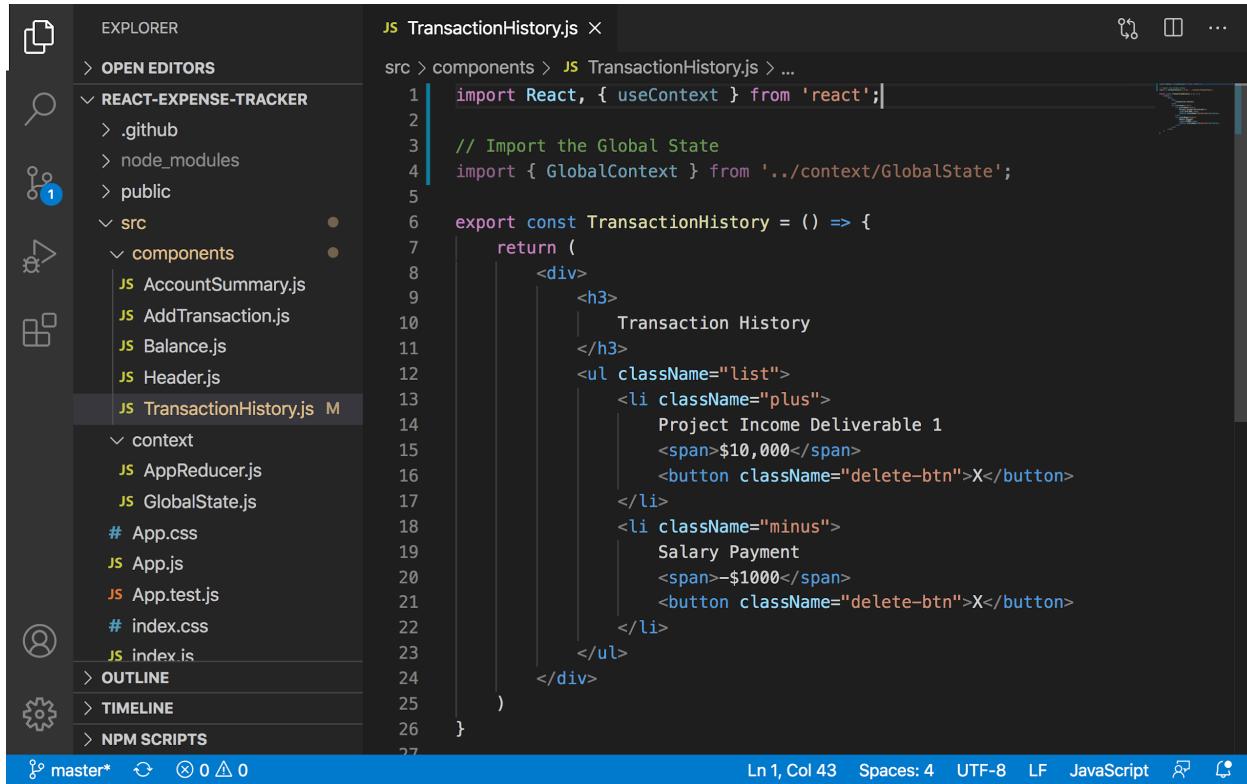
Props:
children: [<Header />, <div />]
new prop: ""

Hooks:
Reducer: {transactions: Array(4)}

Rendered By:

Part IX - Show Transactions from State the Transaction History

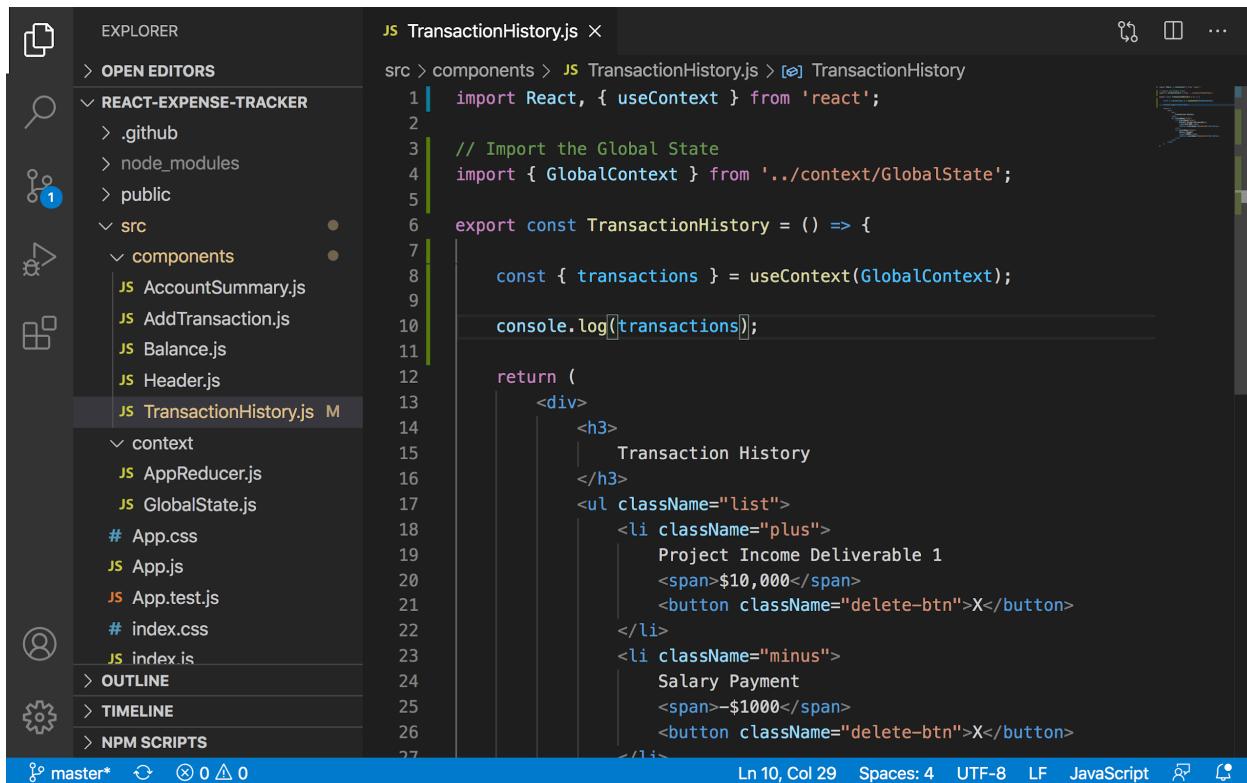
Step 1: Import the Global State from the GlobalContext file and the useContext hook so we can get the data from the context API



The screenshot shows the VS Code interface with the TransactionHistory.js file open in the editor. The code imports React and the useContext hook from react, and the GlobalContext from a context folder. It then defines a TransactionHistory component that returns a div containing a h3 and a list of transactions. Each transaction is represented by a li element with a plus or minus class, a span with a value, and a delete button.

```
JS TransactionHistory.js ×
src > components > JS TransactionHistory.js > ...
1 import React, { useContext } from 'react';
2
3 // Import the Global State
4 import { GlobalContext } from '../context/GlobalState';
5
6 export const TransactionHistory = () => {
7   return (
8     <div>
9       <h3>
10          Transaction History
11        </h3>
12        <ul className="list">
13          <li className="plus">
14            Project Income Deliverable 1
15            <span>$10,000</span>
16            <button className="delete-btn">X</button>
17          </li>
18          <li className="minus">
19            Salary Payment
20            <span>-$1000</span>
21            <button className="delete-btn">X</button>
22          </li>
23        </ul>
24      </div>
25    )
26  }
27 }
```

Step 2: Get access to transactions in the state using the GlobalContext



The screenshot shows the VS Code interface with the TransactionHistory.js file open in the editor. The code has been modified to use the useContext hook to get the transactions from the GlobalContext. The console.log statement is still present to verify the data.

```
JS TransactionHistory.js ×
src > components > JS TransactionHistory.js > [o] TransactionHistory
1 import React, { useContext } from 'react';
2
3 // Import the Global State
4 import { GlobalContext } from '../context/GlobalState';
5
6 export const TransactionHistory = () => {
7
8   const { transactions } = useContext(GlobalContext);
9
10  console.log(transactions);
11
12  return (
13    <div>
14      <h3>
15        Transaction History
16      </h3>
17      <ul className="list">
18        <li className="plus">
19          Project Income Deliverable 1
20          <span>$10,000</span>
21          <button className="delete-btn">X</button>
22        </li>
23        <li className="minus">
24          Salary Payment
25          <span>-$1000</span>
26          <button className="delete-btn">X</button>
27        </li>
28      </ul>
29    </div>
30  )
31 }
```

Step 3: Create the Transaction Component

The screenshot shows the VS Code interface with the Transaction.js file open. The code defines a Transaction component that takes a transaction object and returns a list item with the description and amount, plus a delete button.

```
JS TransactionHistory.js ● JS Transaction.js ×
src > components > JS Transaction.js > ...
1 import React from 'react'
2
3 export const Transaction = ({ transaction }) => {
4     return (
5         <li className="plus">
6             {transaction.description}
7             <span>{transaction.transactionAmount}</span>
8             <button className="delete-btn">X</button>
9         </li>
10    )
11 }
12
```

VS Code status bar: master* Ln 12, Col 1 Spaces: 4 UTF-8 LF JavaScript

Step 4: Import the Transaction component and use the map function to iterate over the transactions and display them

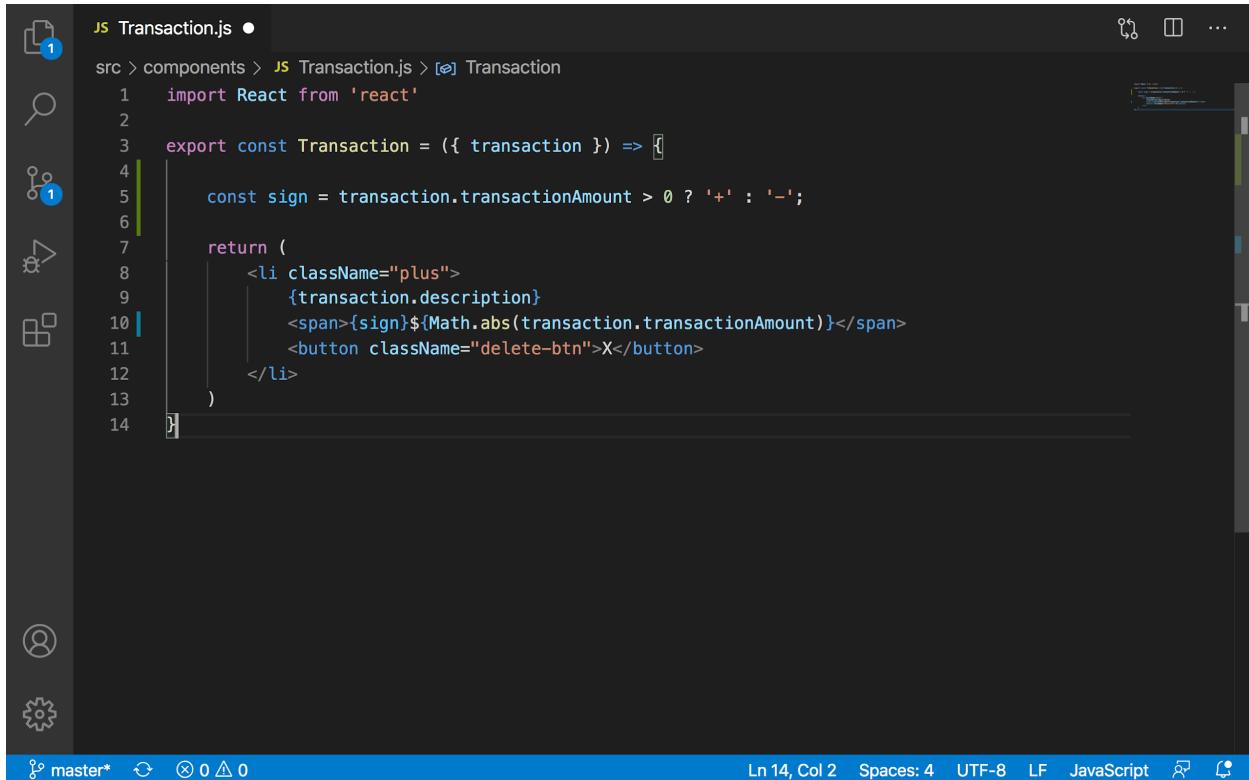
The screenshot shows the TransactionHistory.js file open in VS Code. It imports the Transaction component and uses the map function to iterate over the transactions, rendering each one in a list.

```
JS TransactionHistory.js ×
src > components > JS TransactionHistory.js > [o] TransactionHistory
1 import React, { useContext } from 'react';
2
3 // Import Transaction Component
4 import { Transaction } from './Transaction';
5
6 // Import the Global State
7 import { GlobalContext } from '../context/GlobalState';
8
9 export const TransactionHistory = () => {
10
11     const { transactions } = useContext(GlobalContext);
12
13     console.log(transactions);
14
15     return (
16         <div>
17             <h3>
18                 Transaction History
19             </h3>
20             <ul className="list">
21                 {transactions.map(transaction => (
22                     <Transaction key={transaction.id} transaction={transaction} />
23                 ))}
24             </ul>
25         </div>
26     )
27 }
```

VS Code status bar: master* Ln 25, Col 15 Spaces: 4 UTF-8 LF JavaScript

Part X - Differentiating between Income and Expense

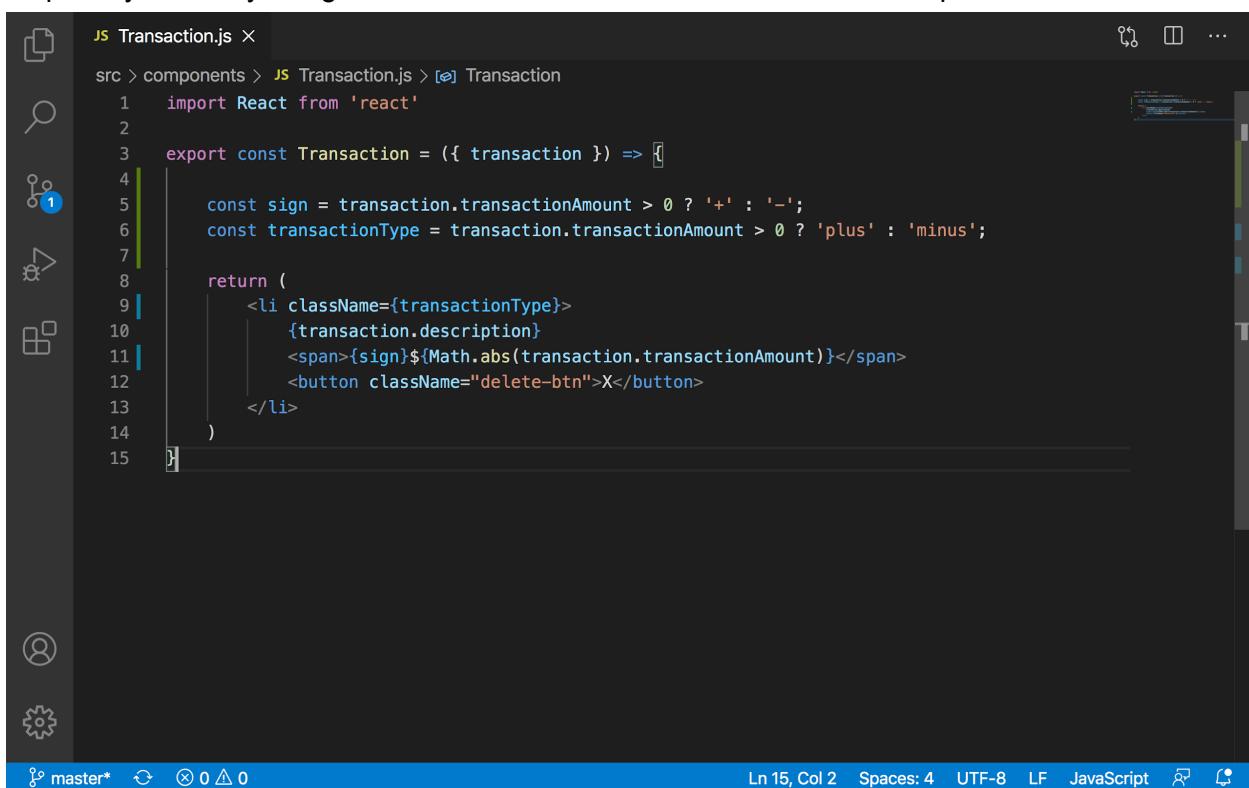
Step 1: Using a ternary operator determine if the transaction amount is positive or negative and use the type to signify positive or negative transaction amount in the display



```
JS Transaction.js ●
src > components > JS Transaction.js > [o] Transaction
1 import React from 'react'
2
3 export const Transaction = ({ transaction }) => [
4
5   const sign = transaction.transactionAmount > 0 ? '+' : '-';
6
7   return (
8     <li className="plus">
9       {transaction.description}
10      <span>{sign}${Math.abs(transaction.transactionAmount)}</span>
11      <button className="delete-btn">X</button>
12    </li>
13  )
14 ]
```

Ln 14, Col 2 Spaces: 4 UTF-8 LF JavaScript ⚙️

Step 2: Dynamically assign class to differentiate between income and expense

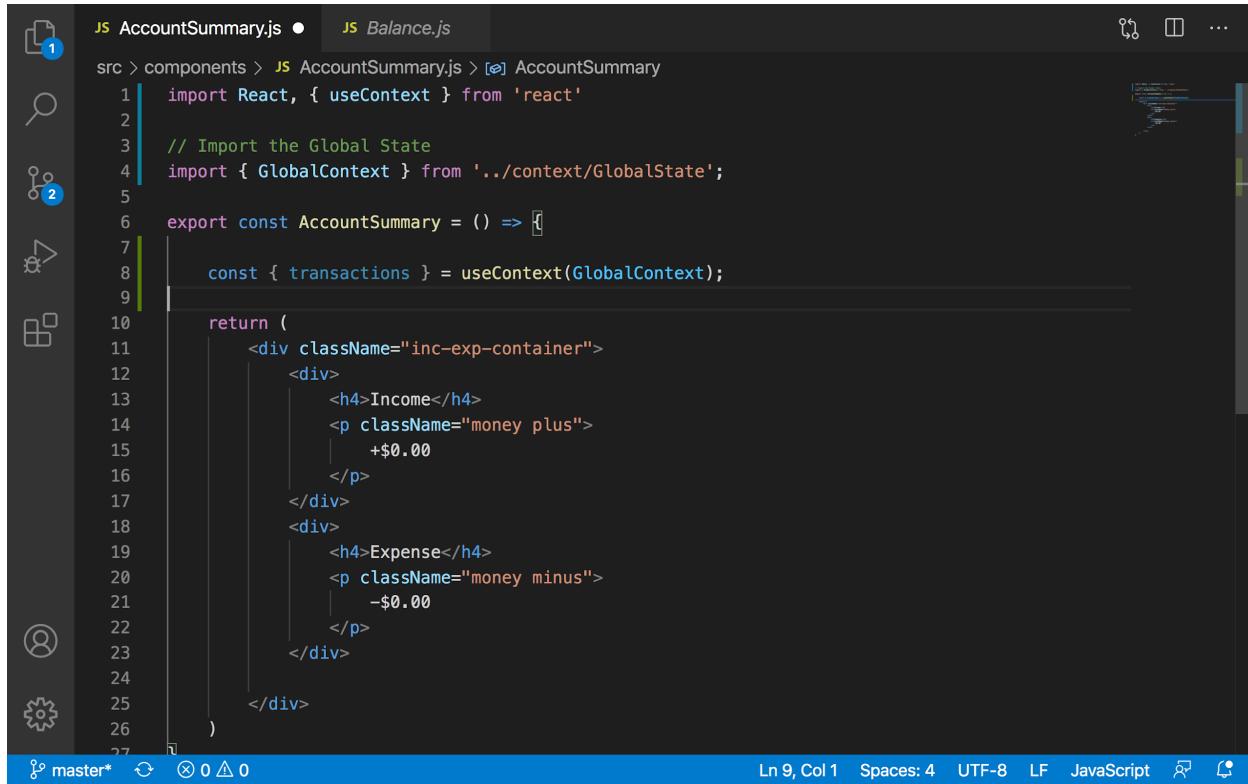


```
JS Transaction.js ✘
src > components > JS Transaction.js > [o] Transaction
1 import React from 'react'
2
3 export const Transaction = ({ transaction }) => [
4
5   const sign = transaction.transactionAmount > 0 ? '+' : '-';
6   const transactionType = transaction.transactionAmount > 0 ? 'plus' : 'minus';
7
8   return (
9     <li className={transactionType}>
10       {transaction.description}
11       <span>{sign}${Math.abs(transaction.transactionAmount)}</span>
12       <button className="delete-btn">X</button>
13     </li>
14   )
15 ]
```

Ln 15, Col 2 Spaces: 4 UTF-8 LF JavaScript ⚙️

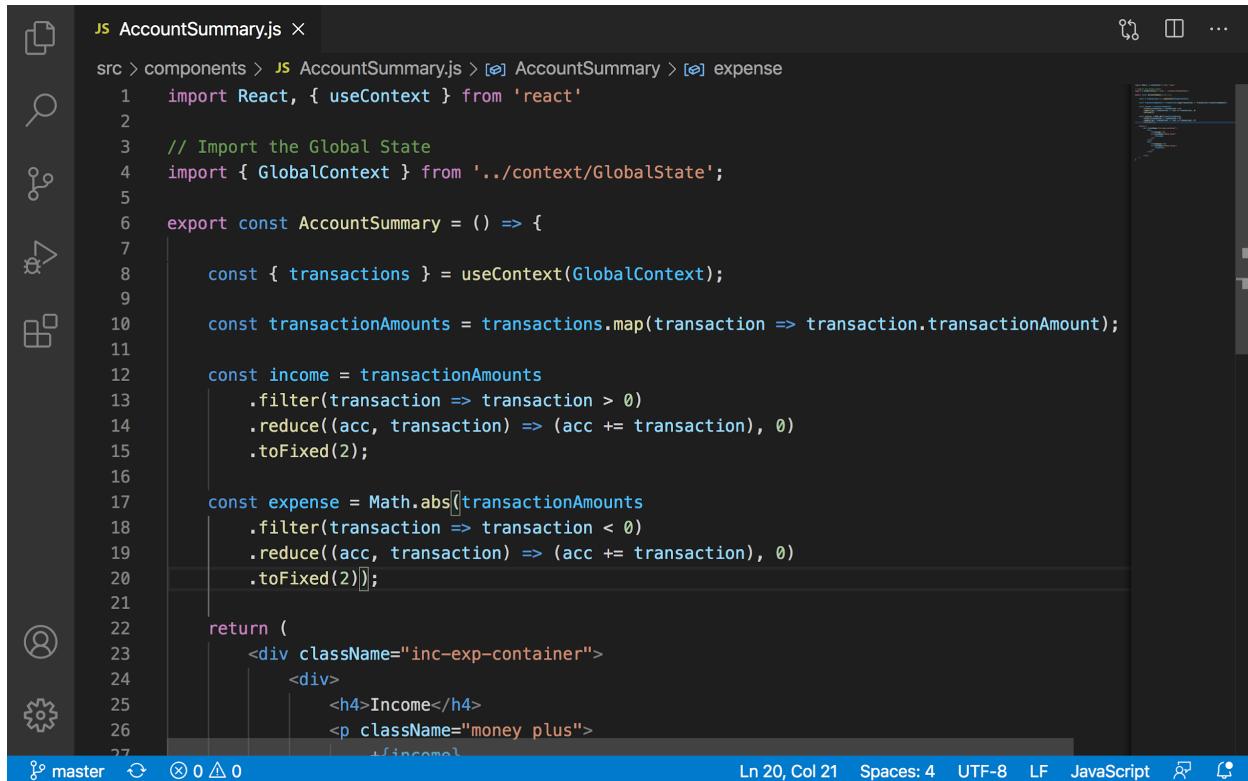
Part X - Updating the Account Summary

Step 1: Import the useContext, GlobalContext and get the transactions



```
JS AccountSummary.js ● JS Balance.js
src > components > JS AccountSummary.js > [o] AccountSummary
1 import React, { useContext } from 'react'
2
3 // Import the Global State
4 import { GlobalContext } from '../context/GlobalState';
5
6 export const AccountSummary = () => [
7
8     const { transactions } = useContext(GlobalContext);
9
10    return (
11        <div className="inc-exp-container">
12            <div>
13                <h4>Income</h4>
14                <p className="money plus">
15                    +$0.00
16                </p>
17            </div>
18            <div>
19                <h4>Expense</h4>
20                <p className="money minus">
21                    -$0.00
22                </p>
23            </div>
24        </div>
25    )
26)
27
Ln 9, Col 1 Spaces: 4 UTF-8 LF JavaScript ⚡ 🎨
```

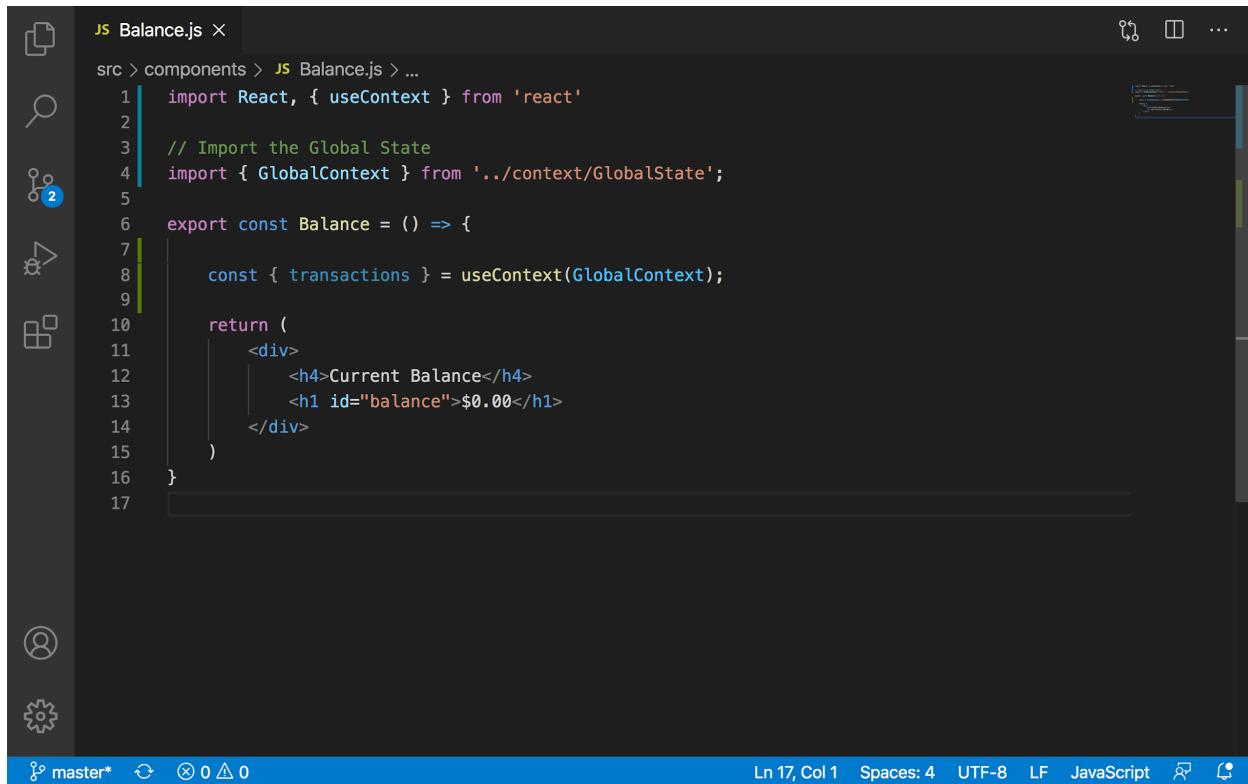
Step 2: Calculate the income and expense from the transactions array



```
JS AccountSummary.js ×
src > components > JS AccountSummary.js > [o] AccountSummary > [o] expense
1 import React, { useContext } from 'react'
2
3 // Import the Global State
4 import { GlobalContext } from '../context/GlobalState';
5
6 export const AccountSummary = () => {
7
8     const { transactions } = useContext(GlobalContext);
9
10    const transactionAmounts = transactions.map(transaction => transaction.transactionAmount);
11
12    const income = transactionAmounts
13        .filter(transaction => transaction > 0)
14        .reduce((acc, transaction) => (acc += transaction), 0)
15        .toFixed(2);
16
17    const expense = Math.abs(transactionAmounts
18        .filter(transaction => transaction < 0)
19        .reduce((acc, transaction) => (acc += transaction), 0)
20        .toFixed(2));
21
22    return (
23        <div className="inc-exp-container">
24            <div>
25                <h4>Income</h4>
26                <p className="money plus">
27                    $100.00
28                </p>
29            </div>
30            <div>
31                <h4>Expense</h4>
32                <p className="money minus">
33                    -$50.00
34                </p>
35            </div>
36        </div>
37    )
38)
39
Ln 20, Col 21 Spaces: 4 UTF-8 LF JavaScript ⚡ 🎨
```

Part XI - Updating the Balance

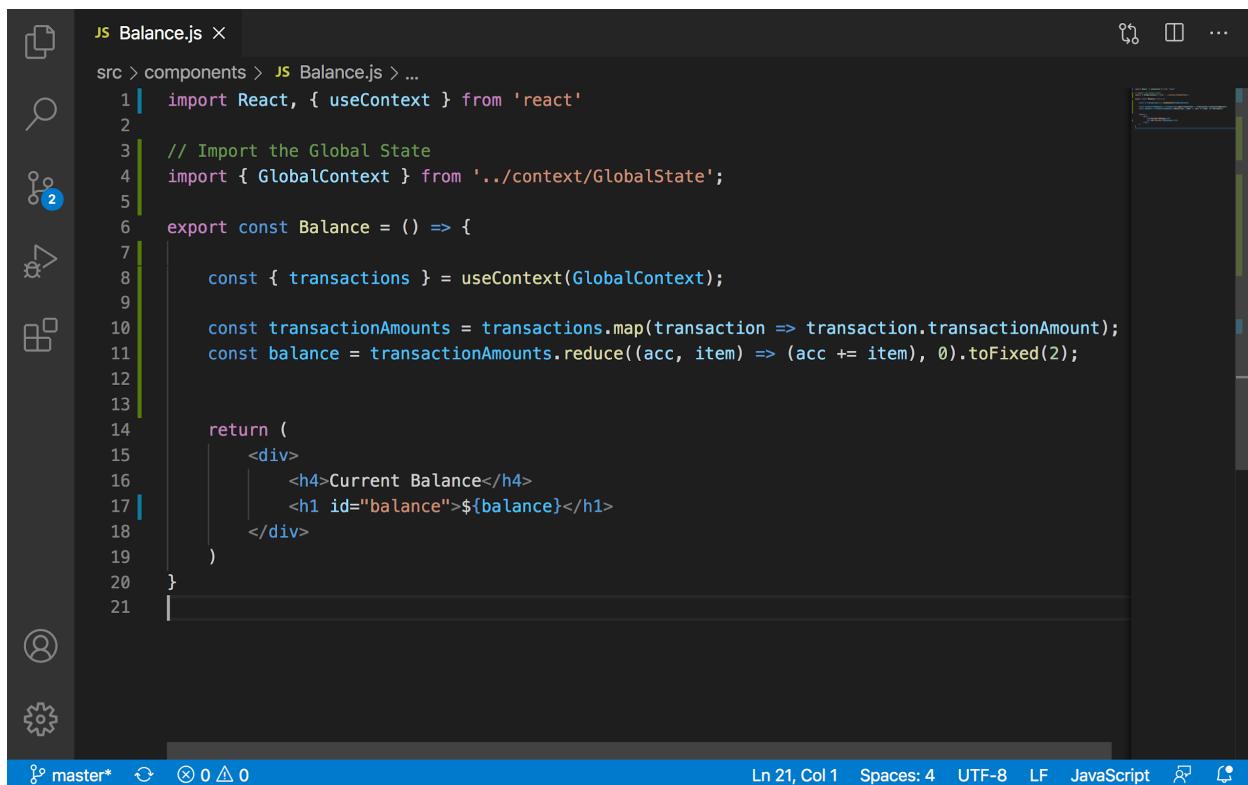
Step 1: Import the useContext, GlobalContext and get the transactions



```
JS Balance.js ×
src > components > JS Balance.js > ...
1 import React, { useContext } from 'react'
2
3 // Import the Global State
4 import { GlobalContext } from '../context/GlobalState';
5
6 export const Balance = () => {
7
8     const { transactions } = useContext(GlobalContext);
9
10    return (
11        <div>
12            <h4>Current Balance</h4>
13            <h1 id="balance">$0.00</h1>
14        </div>
15    )
16}
17
```

Ln 17, Col 1 Spaces: 4 UTF-8 LF JavaScript ⚡ 🎯

Step 2: Add the transaction amounts to get total balance using the reduce method

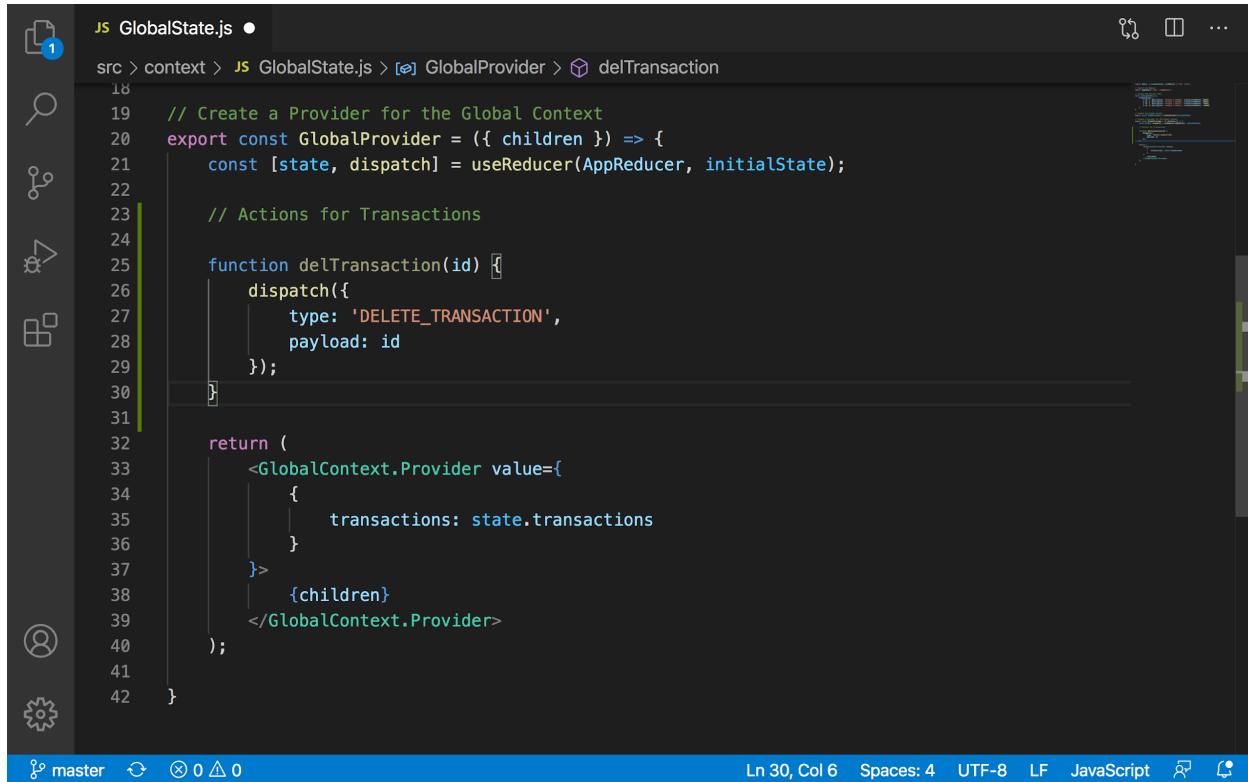


```
JS Balance.js ×
src > components > JS Balance.js > ...
1 import React, { useContext } from 'react'
2
3 // Import the Global State
4 import { GlobalContext } from '../context/GlobalState';
5
6 export const Balance = () => {
7
8     const { transactions } = useContext(GlobalContext);
9
10    const transactionAmounts = transactions.map(transaction => transaction.transactionAmount);
11    const balance = transactionAmounts.reduce((acc, item) => (acc += item), 0).toFixed(2);
12
13
14    return (
15        <div>
16            <h4>Current Balance</h4>
17            <h1 id="balance">${balance}</h1>
18        </div>
19    )
20}
21
```

Ln 21, Col 1 Spaces: 4 UTF-8 LF JavaScript ⚡ 🎯

Part XII - Add an Action to Delete Existing Transactions

Step 1: Create an action in the GlobalProvider for the delTransaction feature

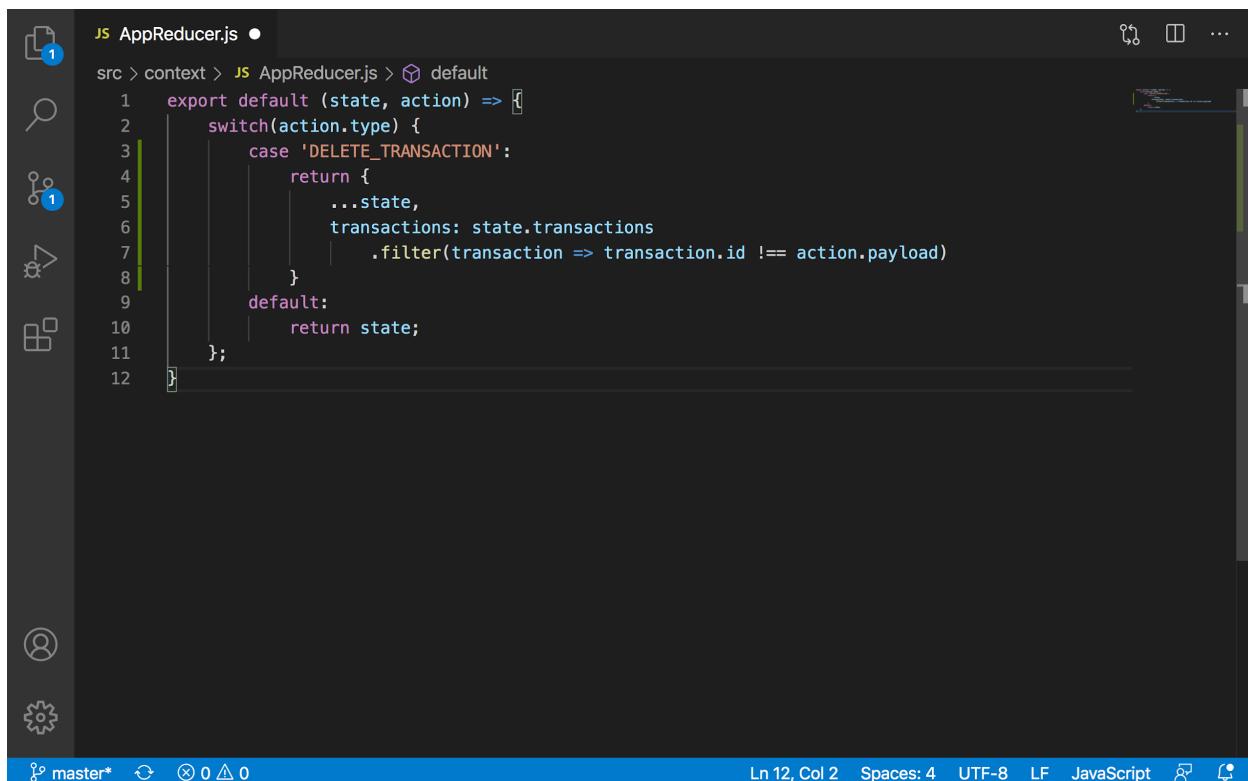


The screenshot shows the GlobalState.js file in a dark-themed code editor. The file defines a GlobalProvider component that uses the useReducer hook to manage state and dispatch actions. A new action, delTransaction, is added to handle the deletion of transactions.

```
JS GlobalState.js ●
src > context > JS GlobalState.js > [o] GlobalProvider > ⚡ delTransaction
18
19 // Create a Provider for the Global Context
20 export const GlobalProvider = ({ children }) => {
21   const [state, dispatch] = useReducer(AppReducer, initialState);
22
23   // Actions for Transactions
24
25   function delTransaction(id) {
26     dispatch({
27       type: 'DELETE_TRANSACTION',
28       payload: id
29     });
30   }
31
32   return (
33     <GlobalContext.Provider value={
34       {
35         transactions: state.transactions
36       }
37     }>
38     {children}
39   </GlobalContext.Provider>
40 );
41
42 }
```

At the bottom of the editor, the status bar shows: master, 0 changes, 0 issues, Ln 30, Col 6, Spaces: 4, UTF-8, LF, JavaScript.

Step 2: Create a case for the delete transaction in the AppReducer to create a new state

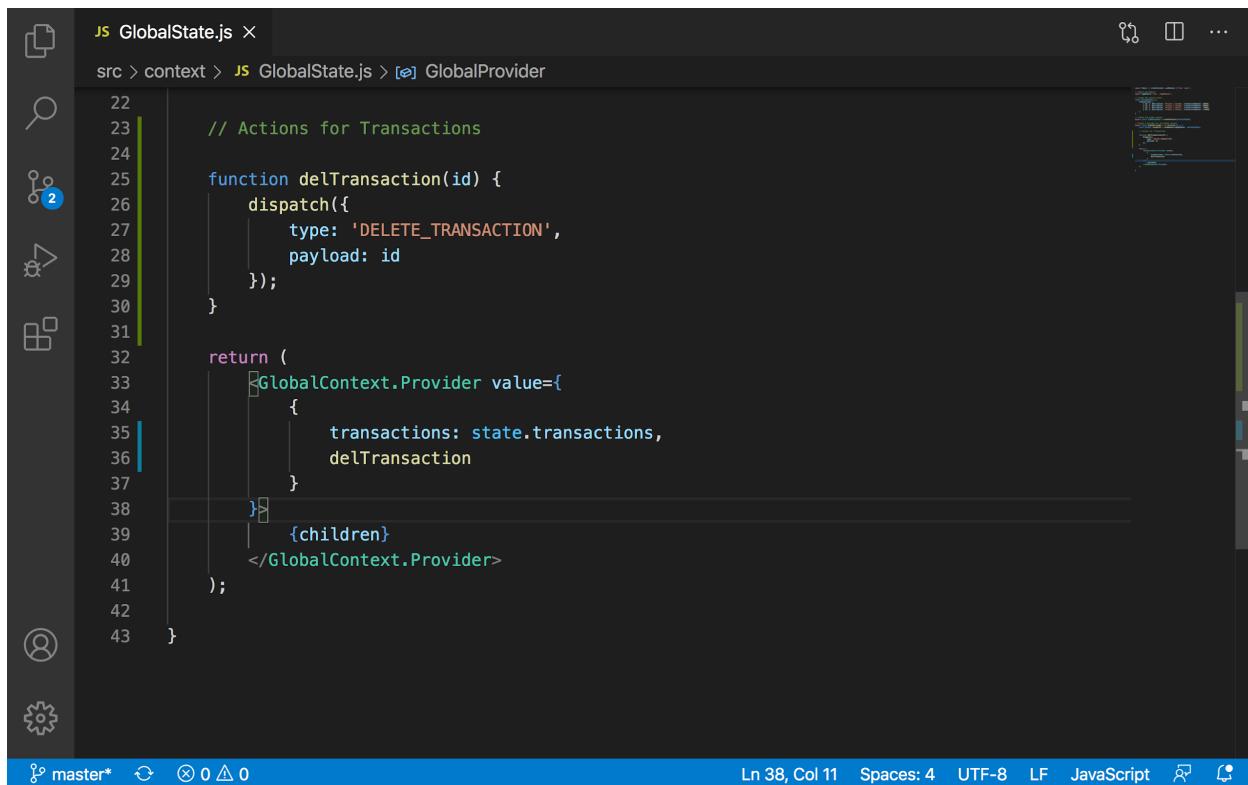


The screenshot shows the AppReducer.js file in a dark-themed code editor. It contains a default export that is a reducer function. The switch statement handles the 'DELETE_TRANSACTION' action by filtering out the transaction with the specified ID from the state's transactions array.

```
JS AppReducer.js ●
src > context > JS AppReducer.js > ⚡ default
1  export default (state, action) => [
2    switch(action.type) {
3      case 'DELETE_TRANSACTION':
4        return {
5          ...state,
6          transactions: state.transactions
7            .filter(transaction => transaction.id !== action.payload)
8        }
9      default:
10        return state;
11    };
12 ]
```

At the bottom of the editor, the status bar shows: master*, 0 changes, 0 issues, Ln 12, Col 2, Spaces: 4, UTF-8, LF, JavaScript.

Step 3: Update the GlobalProvider to pass new delTransaction function to components



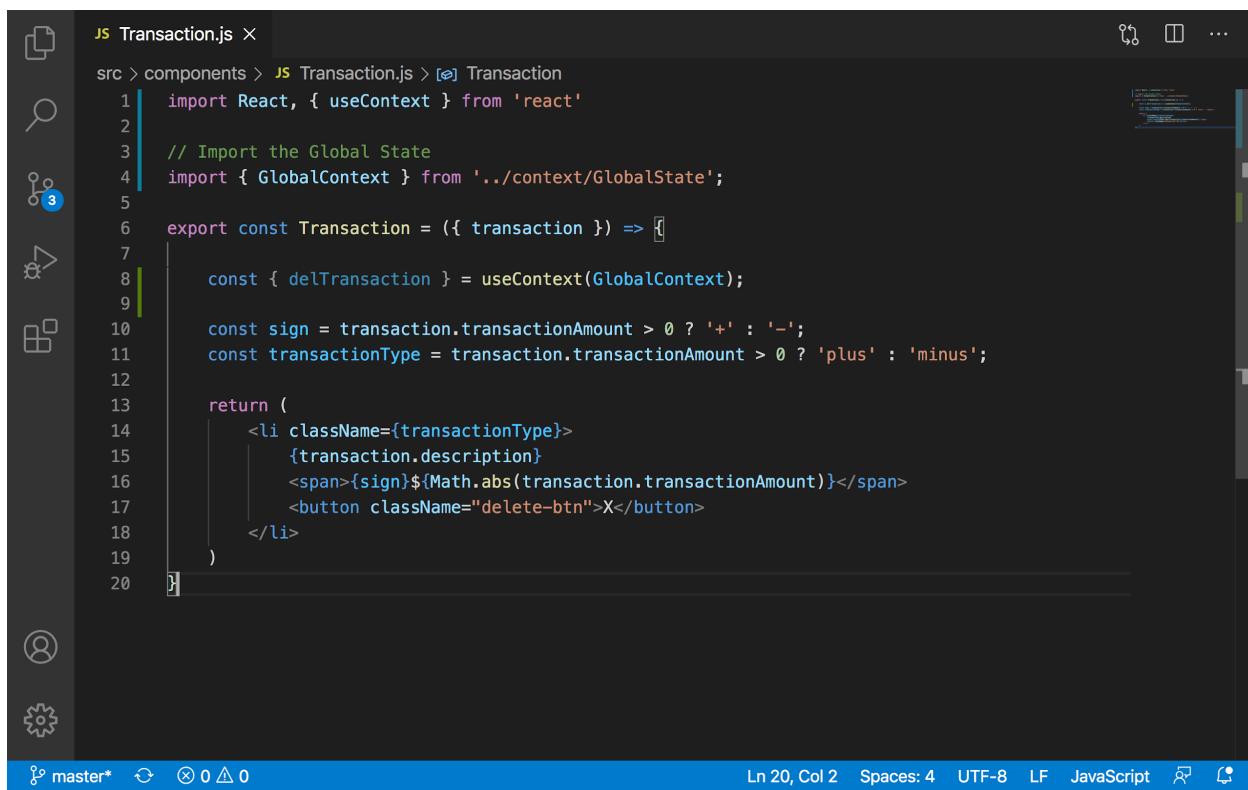
The screenshot shows the GlobalState.js file in VS Code. The code defines a GlobalProvider component that wraps its children with a GlobalContext.Provider. It includes a delTransaction function that dispatches an action to delete a transaction by ID.

```
JS GlobalState.js ×
src > context > JS GlobalState.js > [o] GlobalProvider
22
23     // Actions for Transactions
24
25     function delTransaction(id) {
26         dispatch({
27             type: 'DELETE_TRANSACTION',
28             payload: id
29         });
30     }
31
32     return (
33         <GlobalContext.Provider value={
34             {
35                 transactions: state.transactions,
36                 delTransaction
37             }
38         }>
39         {children}
40         </GlobalContext.Provider>
41     );
42
43 }
```

Step 4: Update the Transaction component with delTransaction

Import the useContext and GlobalContext

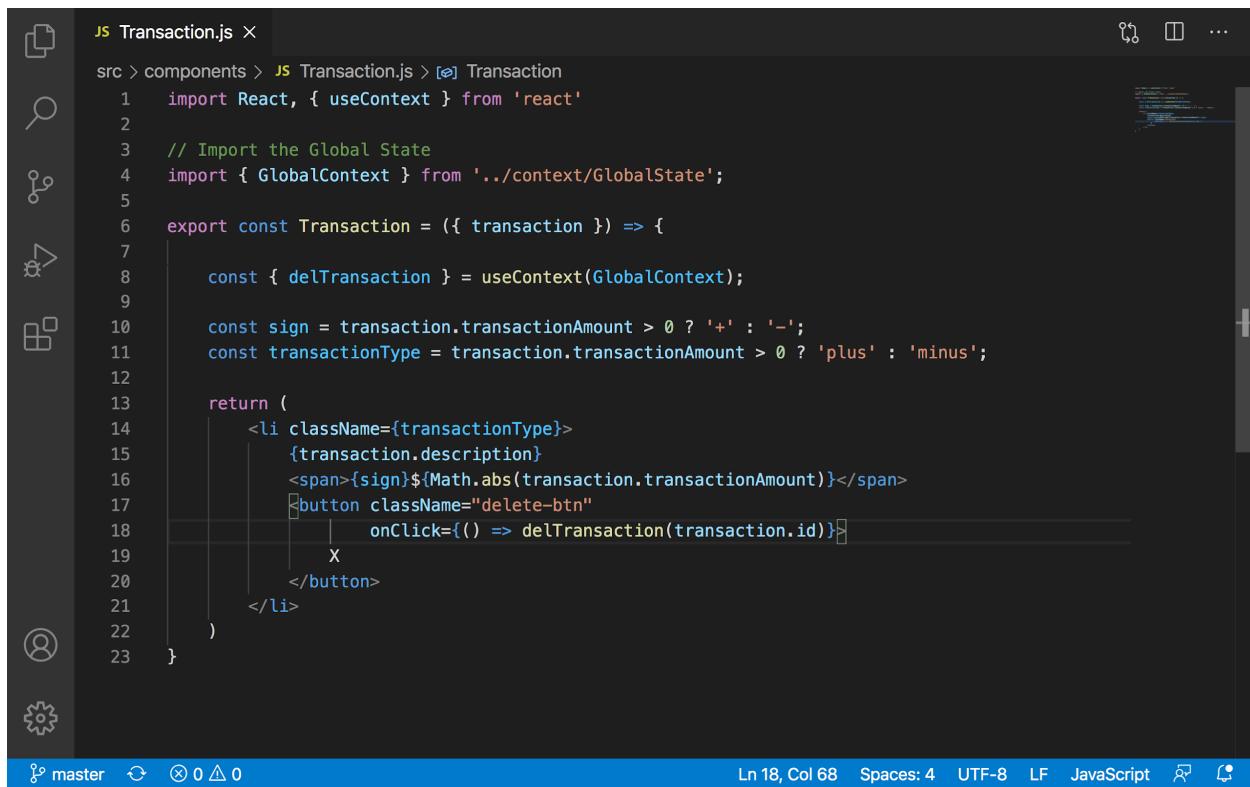
Get the delTransaction from the GlobalContext



The screenshot shows the Transaction.js file in VS Code. It imports React and the GlobalContext provider. The Transaction component uses the useContext hook to get the delTransaction function from the GlobalContext. It then renders a list item with the transaction's description, its absolute amount, and a delete button.

```
JS Transaction.js ×
src > components > JS Transaction.js > [o] Transaction
1 import React, { useContext } from 'react'
2
3 // Import the Global State
4 import { GlobalContext } from '../context/GlobalState';
5
6 export const Transaction = ({ transaction }) => [
7
8     const { delTransaction } = useContext(GlobalContext);
9
10    const sign = transaction.transactionAmount > 0 ? '+' : '-';
11    const transactionType = transaction.transactionAmount > 0 ? 'plus' : 'minus';
12
13    return (
14        <li className={transactionType}>
15            {transaction.description}
16            <span>{sign}${Math.abs(transaction.transactionAmount)}</span>
17            <button className="delete-btn">X</button>
18        )
19    ]
20 ]
```

Step 5: Call the delTransaction function when clicking the x button



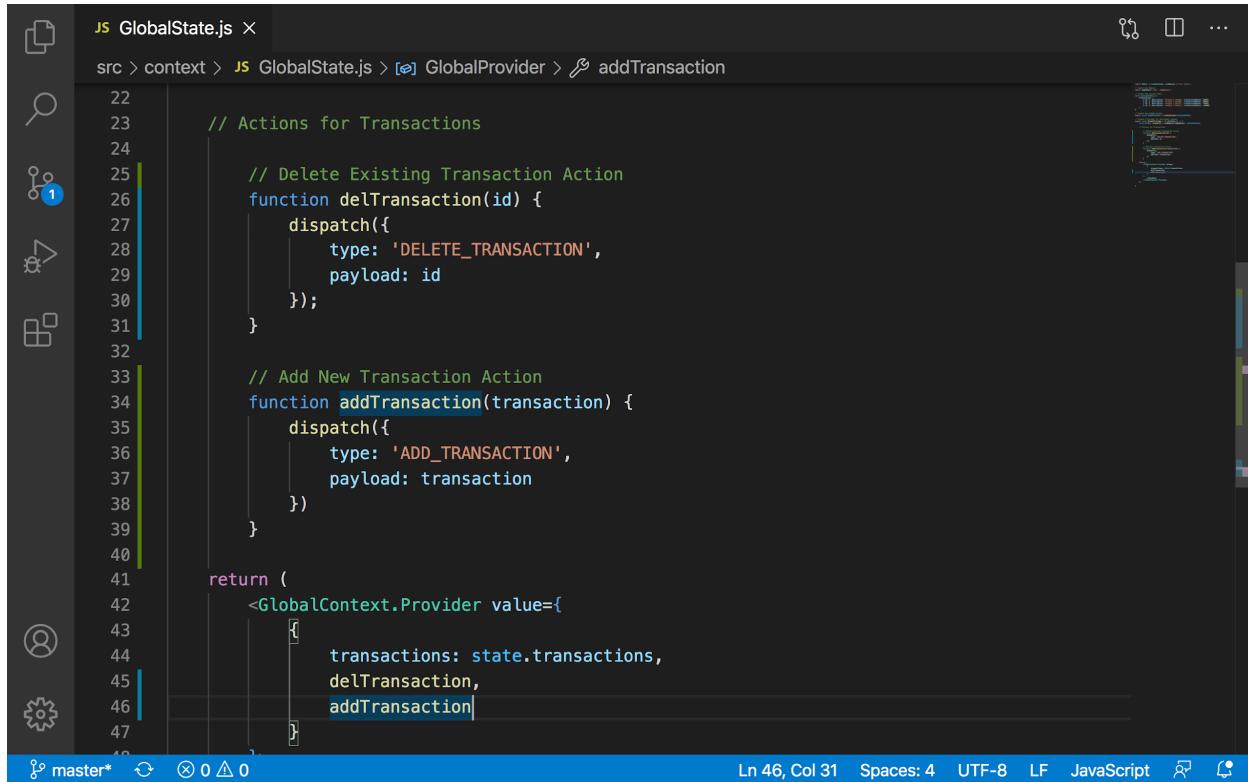
The screenshot shows a code editor window with the file `Transaction.js` open. The code is a React component named `Transaction` that takes a single argument `transaction`. It imports `React` and the `useContext` hook from `react`, and the `GlobalContext` provider from a context file. The component uses the `transactionAmount` from the `transaction` prop to determine the sign (+ or -) and type ('plus' or 'minus'). It then renders a list item (``) with the transaction's description and amount, followed by a button labeled 'X' with a click handler that calls the `delTransaction` function with the transaction's `id`.

```
JS Transaction.js ×
src > components > JS Transaction.js > [e] Transaction
1 import React, { useContext } from 'react'
2
3 // Import the Global State
4 import { GlobalContext } from '../context/GlobalState';
5
6 export const Transaction = ({ transaction }) => {
7
8     const { delTransaction } = useContext(GlobalContext);
9
10    const sign = transaction.transactionAmount > 0 ? '+' : '-';
11    const transactionType = transaction.transactionAmount > 0 ? 'plus' : 'minus';
12
13    return (
14        <li className={transactionType}>
15            {transaction.description}
16            <span>{sign}${Math.abs(transaction.transactionAmount)}</span>
17            <button className="delete-btn"
18                  onClick={() => delTransaction(transaction.id)}>
19                X
20            </button>
21        </li>
22    )
23}
```

Ln 18, Col 68 Spaces: 4 UTF-8 LF JavaScript ⌂ ⌂

Part XIII - Add an Action to Add New Transactions

Step 1: Create an action in the GlobalProvider for the addTransaction feature and add to the Provider

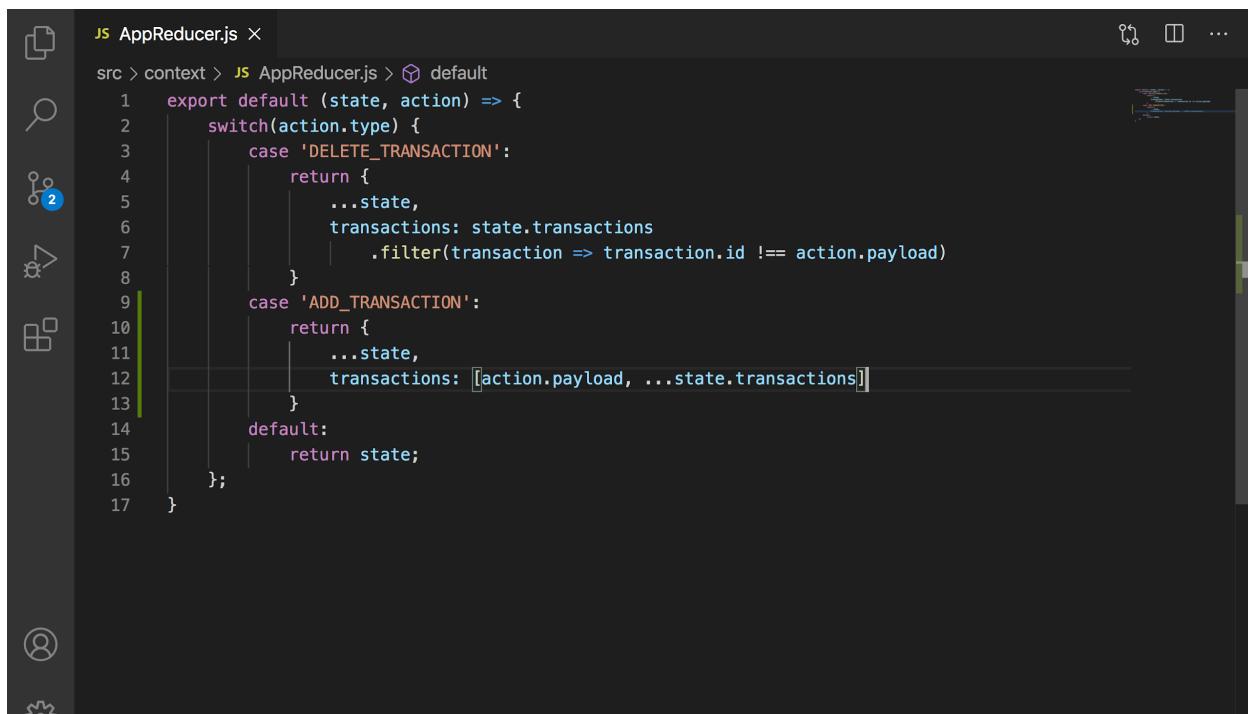


```
JS GlobalState.js ×
src > context > JS GlobalState.js > [o] GlobalProvider > ⚡ addTransaction

22
23     // Actions for Transactions
24
25     // Delete Existing Transaction Action
26     function delTransaction(id) {
27         dispatch({
28             type: 'DELETE_TRANSACTION',
29             payload: id
30         });
31     }
32
33     // Add New Transaction Action
34     function addTransaction(transaction) {
35         dispatch({
36             type: 'ADD_TRANSACTION',
37             payload: transaction
38         });
39     }
40
41     return (
42         <GlobalContext.Provider value={
43             [
44                 {
45                     transactions: state.transactions,
46                     delTransaction,
47                     addTransaction
48                 }
49             ]
50         }>
51     );
52 }

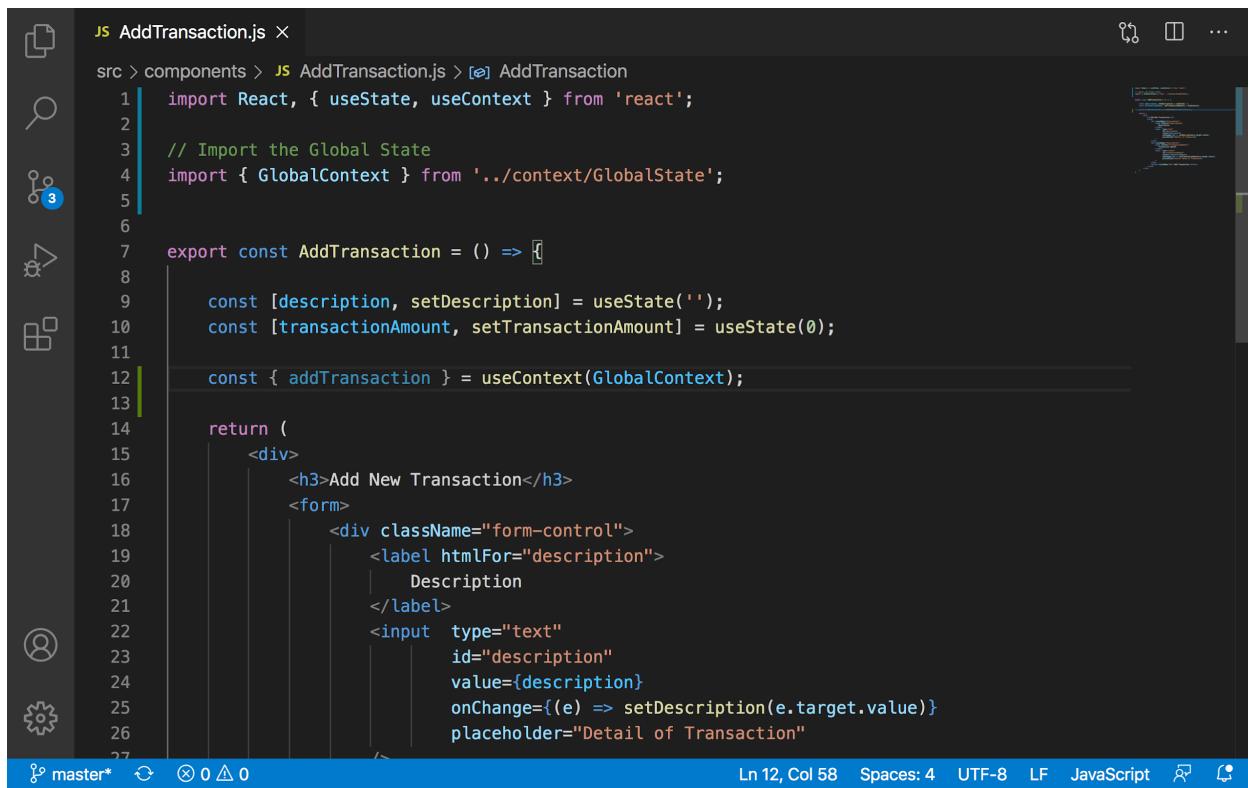
Ln 46, Col 31  Spaces: 4  UTF-8  LF  JavaScript  ⌂  ⌂
```

Step 2: Create a case for the add transaction in the AppReducer to create a new state



```
JS AppReducer.js ×
src > context > JS AppReducer.js > ⚡ default
1  export default (state, action) => {
2      switch(action.type) {
3          case 'DELETE_TRANSACTION':
4              return {
5                  ...state,
6                  transactions: state.transactions
7                      .filter(transaction => transaction.id !== action.payload)
8              }
9          case 'ADD_TRANSACTION':
10             return {
11                 ...state,
12                 transactions: [action.payload, ...state.transactions]
13             }
14         default:
15             return state;
16     };
17 }
```

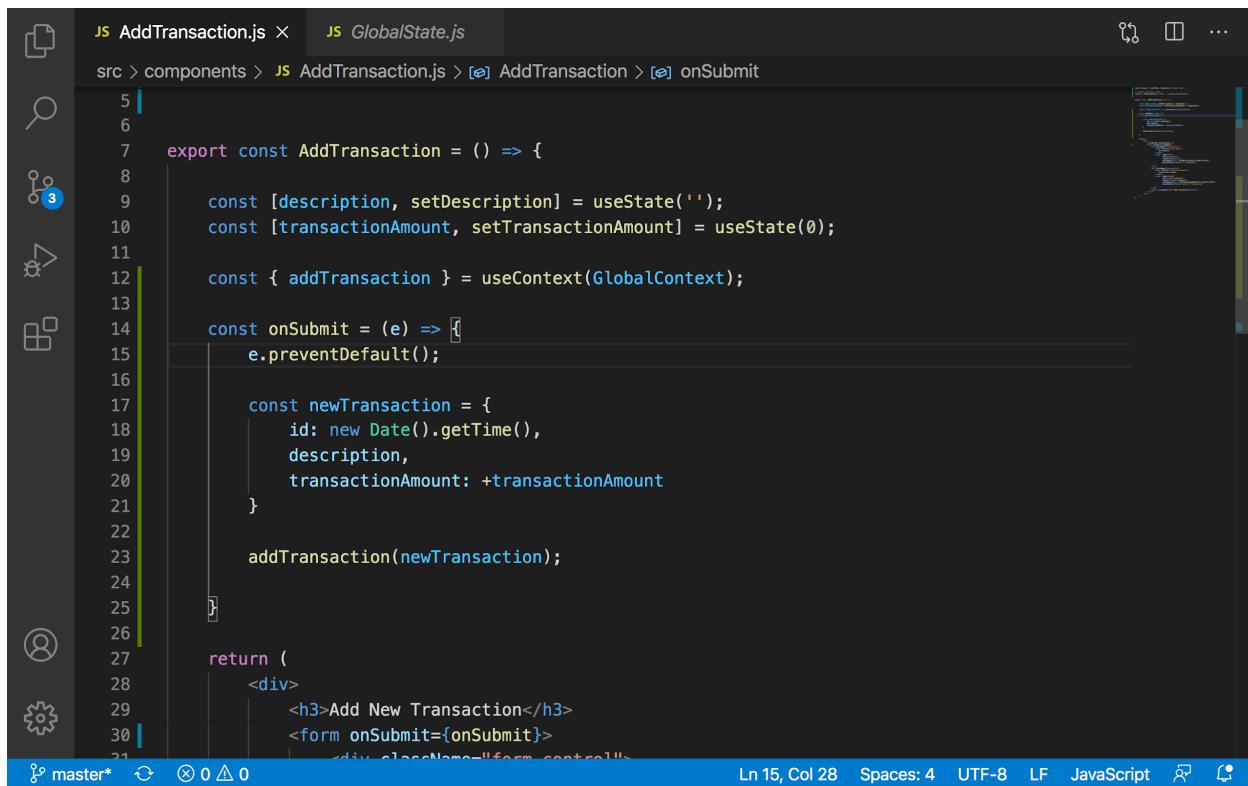
Step 3: Update the AddTransaction component with the new action to add new transactions
Import useContext and the GlobalContext and create the addTransaction const



```
JS AddTransaction.js ×
src > components > JS AddTransaction.js > [o] AddTransaction
1 import React, { useState, useContext } from 'react';
2
3 // Import the Global State
4 import { GlobalContext } from '../context/GlobalState';
5
6
7 export const AddTransaction = () => [
8
9     const [description, setDescription] = useState('');
10    const [transactionAmount, setTransactionAmount] = useState(0);
11
12    const { addTransaction } = useContext(GlobalContext);
13
14    return (
15        <div>
16            <h3>Add New Transaction</h3>
17            <form>
18                <div className="form-control">
19                    <label htmlFor="description">
20                        Description
21                    </label>
22                    <input type="text"
23                        id="description"
24                        value={description}
25                        onChange={(e) => setDescription(e.target.value)}
26                        placeholder="Detail of Transaction"
27                </div>
28            </form>
29        </div>
30    );
31
```

Ln 12, Col 58 Spaces: 4 UTF-8 LF JavaScript

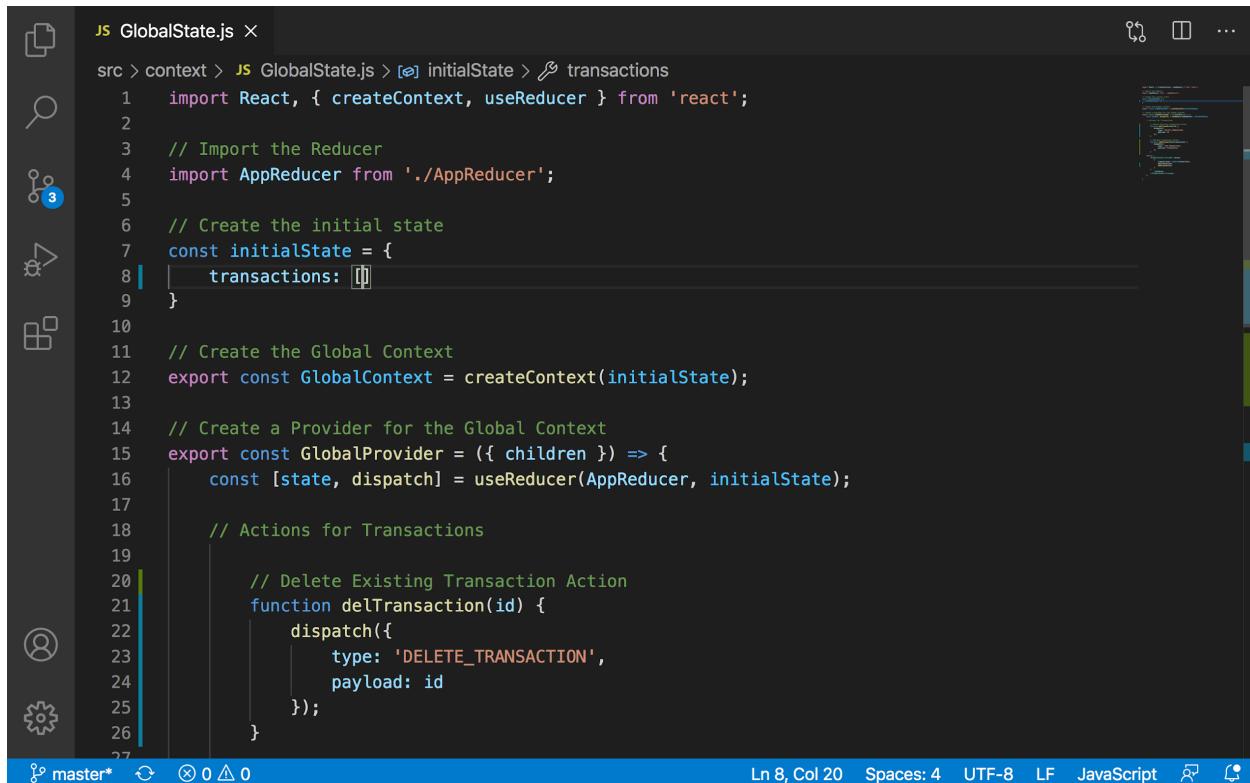
Step 4: Create the onSubmit function and apply the onSubmit event in the form



```
JS AddTransaction.js × JS GlobalState.js
src > components > JS AddTransaction.js > [o] AddTransaction > [o] onSubmit
5
6
7 export const AddTransaction = () => {
8
9     const [description, setDescription] = useState('');
10    const [transactionAmount, setTransactionAmount] = useState(0);
11
12    const { addTransaction } = useContext(GlobalContext);
13
14    const onSubmit = (e) => {
15        e.preventDefault();
16
17        const newTransaction = {
18            id: new Date().getTime(),
19            description,
20            transactionAmount: +transactionAmount
21        }
22
23        addTransaction(newTransaction);
24
25    }
26
27    return (
28        <div>
29            <h3>Add New Transaction</h3>
30            <form onSubmit={onSubmit}>
31                <div className="form-control">
32
```

Ln 15, Col 28 Spaces: 4 UTF-8 LF JavaScript

Step 5: Remove initialState transactions from GlobalState



The screenshot shows a code editor window with the file `GlobalState.js` open. The code is written in JavaScript and defines a global state context. It imports `React` and `useReducer` from `'react'`. It creates an initial state object `initialState` with a `transactions` array. It then creates a `GlobalContext` using `createContext` and a `GlobalProvider` using `useReducer`. Below this, it defines actions for transactions, specifically a `delTransaction` action that dispatches a payload with type `'DELETE_TRANSACTION'` and payload `id`.

```
JS GlobalState.js ×
src > context > JS GlobalState.js > initialState > transactions
1 import React, { createContext, useReducer } from 'react';
2
3 // Import the Reducer
4 import AppReducer from './AppReducer';
5
6 // Create the initial state
7 const initialState = {
8   transactions: []
9 }
10
11 // Create the Global Context
12 export const GlobalContext = createContext(initialState);
13
14 // Create a Provider for the Global Context
15 export const GlobalProvider = ({ children }) => {
16   const [state, dispatch] = useReducer(AppReducer, initialState);
17
18   // Actions for Transactions
19
20   // Delete Existing Transaction Action
21   function delTransaction(id) {
22     dispatch({
23       type: 'DELETE_TRANSACTION',
24       payload: id
25     });
26   }
27
```

Ln 8, Col 20 Spaces: 4 UTF-8 LF JavaScript ⚡ 🔍

