TablrPage after refresh:

environment: development

App.tsx:20 dev\_server: http://localhost:3001

App.tsx:21 prod\_server: https://server.tamarprofile.site

App.tsx:22 checkEnvironment: http://localhost:3001

App.tsx:25 serverUrl: http://localhost:3001

App.tsx:30 Server URL: http://localhost:3001

TablePage.tsx:33 🛠️ Inside TablePage Component

TablePage.tsx:53 Context values: {serverUrl: 'http://localhost:3001', tableContext: {…}}

TablePage.tsx:75 Current rowIndexesArr: []

TablePage.tsx:76 Current cells: []

TablePage.tsx:86 Current tableId: 67c04ae8ec6d6f00c6f51ec2

TablePage.tsx:87 All tables in context: []

TablePage.tsx:91 Table with ID 67c04ae8ec6d6f00c6f51ec2 not found. Waiting for tables to load...

tableContext.tsx:40 useEffect triggered in TableContext

tableContext.tsx:41 Tables before fetching: []

tableContext.tsx:45 Fetching tables...

docApi.tsx:118 at DocumentRestAPIMethods.get the response is: {data: Array(3), status: 200, statusText: 'OK', headers: AxiosHeaders, config: {…}, …}

getAllUserTables.tsx:22 at getAllUserTables the tablesData (3) [{…}, {…}, {…}]

TablePage.tsx:33 🛠️ Inside TablePage Component

TablePage.tsx:53 Context values: {serverUrl: 'http://localhost:3001', tableContext: {…}}

TablePage.tsx:75 Current rowIndexesArr: []

TablePage.tsx:76 Current cells: []

TablePage.tsx:86 Current tableId: 67c04ae8ec6d6f00c6f51ec2

TablePage.tsx:87 All tables in context: (3) [{…}, {…}, {…}]

TablePage.tsx:101 Current tableName: t3h

TablePage.tsx:102 Current tableIndex: 3

generateCellsForPlot.tsx:4 generateCellsForPlot called with: {rowIndexesArr: Array(0), cells: Array(0)}

TablePage.tsx:106 After calling generateCellsForPlot displayArr is: []

Exception: Error at describeNativeComponentFrame (http://localhost:5173/node\_modules/.vite/deps/chunk-6BKLQ22S.js?v=84e5a9f8:1451:23) at describeFunctionComponentFrame (http://localhost:5173/node\_modules/.vite/deps/chunk-6BKLQ22S.js?v=84e5a9f8:1514:20) at describeFiber (http://localhost:5173/node\_modules/.vite/deps/chunk-6BKLQ22S.js?v=84e5a9f8:1573:22) at getStackByFiberInDevAndProd (http://localhost:5173/node\_modules/.vite/deps/chunk-6BKLQ22S.js?v=84e5a9f8:1587:23) at getCurrentFiberStackInDev (http://localhost:5173/node\_modules/.vite/deps/chunk-6BKLQ22S.js?v=84e5a9f8:1752:20) at ReactDebugCurrentFrame.getStackAddendum (http://localhost:5173/node\_modules/.vite/deps/chunk-DRWLMN53.js?v=84e5a9f8:85:24) at printWarning (http://localhost:5173/node\_modules/.vite/deps/chunk-6BKLQ22S.js?v=84e5a9f8:512:49) at error (http://localhost:5173/node\_modules/.vite/deps/chunk-6BKLQ22S.js?v=84e5a9f8:505:15) at warnOnHookMismatchInDev (http://localhost:5173/node\_modules/.vite/deps/chunk-6BKLQ22S.js?v=84e5a9f8:11495:17) at updateHookTypesDev (http://localhost:5173/node\_modules/.vite/deps/chunk-6BKLQ22S.js?v=84e5a9f8:11465:17)

function describeNativeComponentFrame(fn, construct) { fn= f TablePage(), construct - false

// If something asked for a stack inside a fake render, it should get ignored.

if ( !fn || reentry) { fn = f TablePage()

return '';

}

{

var frame = componentFrameCache.get(fn); frame = undefined, fn = f TablePage()

if (frame !== undefined) { frame = undefined

return frame;

}

}

var control; control = undefined

reentry = true;

var previousPrepareStackTrace = Error.prepareStackTrace; // $FlowFixMe It does accept undefined.

Error.prepareStackTrace = undefined;

var previousDispatcher; previousDispatcher = {readContext: f , useCallback: f, useContex: f, useEffect: f, useImperativeHandle: f, …}

{

previousDispatcher = ReactCurrentDispatcher.current; // Set the dispatcher in DEV because this might be call in the render function previousDispatcher = {readContext: f , useCallback: f, useContex: f, useEffect: f, useImperativeHandle: f, …}

// for warnings.

ReactCurrentDispatcher.current = null;

disableLogs();

}

try {

// This should throw.

if (construct) { construct = false

// Something should be setting the props in the constructor.

var Fake = function () { Fake = undefined

throw Error();

}; // $FlowFixMe

Object.defineProperty(Fake.prototype, 'props', {

set: function () {

// We use a throwing setter instead of frozen or non-writable props

// because that won't throw in a non-strict mode function.

throw Error();

}

});

if (typeof Reflect === 'object' && Reflect.construct) {

// We construct a different control for this case to include any extra

// frames added by the construct call.

try {

Reflect.construct(Fake, []); Fake = undefined

} catch (x) {

control = x; control = undefined

}

Reflect.construct(fn, [], Fake); fn = f TablePage(), Fake = undefined

} else {

try {

Fake.call(); Fake = undefined

} catch (x) {

control = x; control = undefined

}

fn.call(Fake.prototype); fn = f TablePage(), Fake = undefined

}

} else {

try {

throw Error();

} catch (x) {

control = x;

}

fn();

}