

# 2023-05-23 Frontend [WIP]

Breef (Curated)

Exported on 07/04/2023

## Table of Contents

No headings included in this document

<b>1. Date</b>	23 Мая 2023 г.	Notes
<b>2. Repo</b>	<b>frontend</b>	<a href="https://gitlab.light-it.tools/breef/frontend">https://gitlab.light-it.tools/breef/frontend</a>
<b>Repository setup(Gitflow)</b>		
<b>Sonar</b>		
<b>Following the defined architecture</b>		
<b>Churn</b>	-	
<b>Cognitive complexity</b>		
<b>Cyclomatic complexity</b>		

## Code duplication (DRY)

**OK**

- We can encapsulate duplicated code in separate place (done)

```

TermsOfUseProject.tsx
...
TermsOfUseStandard.tsx
...
TermsOfUseBrief.tsx

```

- duplication of type definition (done)

```

useTeamMemberController.ts
TeamMember.tsx
useTeamMemberRemoveInvitationController.ts
useTeamMemberRemoveInvitation.ts

```

- two identical components (done)

```

Collapse.tsx
Accordion.tsx

```

- duplication (done)

```

ActionTip.tsx
ActionTip.ts
projectType.ts
Answers.ts
Answers.tsx

```

## KISS

**OK**

Following standards	Has issues
	<ul style="list-style-type: none"> <li>We can improve this solution (original on the left side) (done)           <ol style="list-style-type: none"> <li>1 Introduced the StorageType type alias to represent the storage type ('cookie' or 'local').</li> <li>2 Introduced the StorageMethod interface to represent the shape of storage methods</li> <li>3 Created the getStorageMethod function that returns the appropriate storage method object based on the storage type</li> <li>4 Remove unnecessary "Data" from name functions</li> </ol> <pre>storageController.ts frontend &gt; libs &gt; shared &gt; utils &gt; src &gt; lib &gt; storage-service &gt; tests &gt; setStorageData 13  export const setStorageData =  14    T extends Record&lt;string, string   object&gt;, 15  &gt;( 16    storage: 'cookie'   'local', 17    key: string, 18    data: T, 19    options?: object, 20  ) =&gt; ( 21    if (options === undefined) { 22      options = { domain: APP_URL_COOKIE }; 23    } 24    switch (storage) { 25      case 'cookie': 26        return setCookie(key, data, options); 27      case 'local': 28        return setLocalStorage(key, data); 29    } 30  ); 31  export const getStorageData = (storage: 'cookie'   'local', key: string) =&gt; ( 32    switch (storage) { 33      case 'cookie': 34        return getCookie(key); 35      case 'local': 36        return getLocalStorage(key); 37    } 38  ); 39  export const removeStorageData = ( 40    storage: 'cookie'   'local', 41    key: string, 42    options?: object, 43  ) =&gt; ( 44    if (options === undefined) { 45      options = { domain: APP_URL_COOKIE }; 46    } 47    switch (storage) { 48      case 'cookie': 49        return removeCookie(key, options); 50      case 'local': 51        return removeLocalStorage(key); 52    } 53  ); 54  frontend &gt; libs &gt; shared &gt; utils &gt; src &gt; lib &gt; storage-service &gt; tests &gt; getFromStorage 55  type StorageType = 'cookie'   'local'; 56  interface StorageMethod { 57    set(key: string, data: any, options?: object): void; 58    get(key: string): any; 59    remove(key: string, options?: object): void; 60  } 61  const getStorageMethod = (storage: StorageType): StorageMethod =&gt; { 62    switch (storage) { 63      case 'cookie': 64        return { 65          set: (key, data, options) =&gt; setCookie(key, data, options), 66          get: (key) =&gt; getCookie(key), 67          remove: (key, options) =&gt; removeCookie(key, options), 68        }; 69      case 'local': 70        return { 71          set: (key, data) =&gt; setLocalStorage(key, data), 72          get: (key) =&gt; getLocalStorage(key), 73          remove: (key) =&gt; removeLocalStorage(key), 74        }; 75    } 76  } 77  export const setToStorage = (storage: StorageType, key: string, data: any, options?: object): void =&gt; { 78    const storageMethod = getStorageMethod(storage); 79    const storageOptions = options ?? { domain: APP_URL_COOKIE }; 80    storageMethod.set(key, data, storageOptions); 81  } 82  export const removeFromStorage = (storage: StorageType, key: string): void =&gt; { 83    const storageMethod = getStorageMethod(storage); 84    const storageOptions = options ?? { domain: APP_URL_COOKIE }; 85    storageMethod.remove(key, storageOptions); 86  } 87</pre> </li> <li>we can eliminate the redundant <code>return false</code> statement and we can use <code>key</code> property instead of <code>keyCode</code> property of event object. Check import please. (done)</li> </ul> <pre>preventSubmittedInput.ts frontend &gt; libs &gt; shared &gt; utils &gt; src &gt; lib &gt; preventSubmittedInput &gt; preventSubmittedInput 1  export const handlePreventSubmittedInput = ( 2    e: React.KeyboardEvent&lt;HTMLInputElement&gt;, 3  ) =&gt; { 4    if (e.keyCode === 13) { <span style="color:red">Original</span> 5      e.preventDefault(); 6      return false; 7    } 8    return false; 9  }; 10 11 export const handlePreventSubmittedInput = ( 12   e: React.KeyboardEvent&lt;HTMLInputElement&gt;, 13 ) =&gt; { 14   if (e.key === "Enter") { <span style="color:red">Proposition</span> 15     e.preventDefault(); 16   } 17   return false; 18 }; 19</pre>

- we can remove unnecessary if/else statement and redundant return. Also I propose to change the name of the function for example it can be `FilterLinksByType` (done)

```

1 import { SocialLinks } from '@breef/shared/types';
2
3 export const replaceOtherLinks = (
4   links: SocialLinks[],
5   type: 'website' | 'portfolio',
6 ) => {
7   if (!links.length) {
8     return [];
9   } else if (links.length) {
10     const link = links.find(
11       item => (item as SocialLinks).title.toLowerCase() === type,
12     );
13     return link ? [link] : [];
14   }
15   return;
16 };
17
18 export const replaceOtherLinks = (           original
19   links: SocialLinks[],                      proposition
20   type: 'website' | 'portfolio',
21 ): SocialLinks[] => {
22   const link = links.find(item => item.title.toLowerCase() === type);
23   return link ? [link] : [];
24 };

```

- we need to simplify this function and then it will be easier for understanding and more readable also we need to change the name to convertAdvantages (done)
- 

```

17
18 export const replaceToStringClientAdvantages = (           original
19   advantages: { name: string }[],                         proposition
20 ) => {
21   if (!advantages.length) {
22     return '';
23   } else {
24     return advantages.reduce((acc, current, idx) => {
25       const str =
26         `${acc} +
27         (idx < advantages.length - 1
28          ? `${current.name}, `
29          : `${current.name}`);
30       return str;
31     }, '');
32   }
33 };
34
35 export const convertAdvantages = (advantages: { name: string }[]): string => {           proposition
36   return advantages.length === 0 ? '' : advantages.map(advantage => advantage.name).join(', ');
37 };
38

```

- 1 we need to change type name for my opinion it'll be great to use Phone or PhoneNumber
- 2 change name of the function parsePhoneNumber or processPhoneNumber
- 3 optional parameter is not correct
- 4 we can combine it in one identificator and reuse it
- 5 Why are we not interested in error?
- (done)

```

parsePhoneNumberToObj.ts 1 ×
frontend > libs > shared > utils > src > lib > phone-number-transformers > parsePhoneNumberToObj.ts > ...
1 import { parsePhoneNumberWithError } from 'libphonenumber-js';
2
3 type ParsedPhoneNumberObjType = {
4   number: string;
5   code: string;
6   numberWithoutCountryCode: string;
7 };
8
9 export const parsePhoneNumberToObj = (
10   phoneNumber?: string, // 3
11 ): ParsedPhoneNumberObjType => {
12   try {
13     const parsedPhoneNumber = parsePhoneNumberWithError(phoneNumber || '');
14     const countryCallingCode = '+' + parsedPhoneNumber.countryCallingCode;
15     return {
16       number: parsedPhoneNumber.number, // 4
17       code: countryCallingCode,
18       numberWithoutCountryCode: parsedPhoneNumber.number.replace(
19         countryCallingCode,
20         '',
21       ),
22     };
23   } catch (error) { // 5
24     return {
25       number: phoneNumber || '',
26       code: '',
27       numberWithoutCountryCode: '',
28     };
29   }
30 };
31

```

- I propose to change function name, don't use unnecessary "?" operator, simplify conditions
- (done)

```

formatText.ts 1 ●
libs > shared > utils > src > lib > format-text > formatText.ts > convertShortBudgetToLongBudget
1 import numeral from 'numeral';
2
3 export const shortBudgetToLongBudget = ({ value }: { value: string }): { value: string } => {
4   const valueArray = value?.split('-');
5   if (valueArray?.length < 2) {
6     return (
7       valueArray &&
8       `${numeral(valueArray[0].trim()).replace('+', '')}.format(
9         '$0,00',
10       )`;
11     );
12   }
13   return (
14     valueArray &&
15     `${numeral(valueArray[0].trim()).format('$0,0')} - ${numeral(
16       valueArray[1].trim(),
17     ).format('$0,0')}`;
18   );
19 };
20
21 export const convertShortBudgetToLongBudget = ({ value }: { value: string }): string => {
22   const budgetRange = value.split('-');
23
24   if (budgetRange.length < 2) {
25     return `${numeral(budgetRange[0].trim()).replace('+', '')}.format('$0,00')`;
26   }
27
28   return `${numeral(budgetRange[0].trim()).format('$0,0')} - ${numeral(budgetRange[1].trim()).format('$0,0')}`;
29 };
30

```

- 1 change name of the function 2 change name of ValueArray 3 remove "?" 4 I believe we don't need to use second if instead of it we need to use else
- (done)

- 

```

formatText.ts 1
libs > shared > utils > src > lib > format-text > formatText.ts > shotBudgetToMinMax
29  };
30  1
31  export const shotBudgetToMinMax = (value: string) => {
32  2   if (!value) return { min: 0, max: 0 };
33  3   const valueArray = value?.split('-');
34  Let maxBudget = 0;
35  Let minBudget = 0;      3
36
37  if (valueArray?.length < 2) {
38    minBudget = numeral(valueArray[0].trim().replace('+', '')).value() || 0;
39    maxBudget = minBudget && minBudget * 1.2;
40  4
41  if (valueArray?.length >= 2) {
42    minBudget = numeral(valueArray[0].trim()).value() || 0;
43    maxBudget = numeral(valueArray[1].trim()).value() || 0;
44  }
45
46  return {
47    min: minBudget,
48    max: maxBudget,
49  };
50 }

```

- It will be great to use enum or typing (now we use "magic string")
- (done)

- 

```

formatBrandDocumentsIcon.tsx 1
frontend > libs > shared > utils > src > lib > formatBrandDocuments > formatBrandDocumentsIcon.tsx > ...
1 import { FileIcon, PdfIcon } from '@breef/shared/assets';
2
3 export const getBrandDocumentIcon = (linkTitle: string) => {
4   const linkType = linkTitle.split('.');
5
6   if (linkType[linkType.length - 1] === 'pdf')
7     return <PdfIcon className="pdf-icon" data-testid="pdf-icon" />;
8   }
9   if (linkType[linkType.length - 1] === 'doc') {
10     return <FileIcon className="pdf-icon" data-testid="file-icon" />;
11   }
12   if (linkType[linkType.length - 1] === 'docx') {
13     return <FileIcon className="pdf-icon" data-testid="file-icon" />;
14   }
15   return null;
16 };
17

```

- We need to change name SocialLinks to SocialLink also take into account one of possible solution of refactoring. It is necessary to change the name of function
- (done)

```

replaceSocialLinks.ts • sharingTypes
frontend > libs > shared > utils > src > lib > formatLinks > replaceSocialLinks.ts ...
17
18 import { SocialLinks } from '@breef/shared/types';
19
20 export const replaceSocialLinks = ({
21   socialLink,
22 }: {
23   socialLink: SocialLinks[];
24 }) => {
25   if (!socialLink.length) {
26     return socialLinksDefault;
27   }
28   const tiktokLink = socialLink.find(item => item.title === 'tiktok');
29   const twitterLink = socialLink.find(item => item.title === 'twitter');
30   const instagramLink = socialLink.find(item => item.title === 'instagram');
31
32   return [
33     {
34       title: 'tiktok',
35       link: tiktokLink?.link || '',
36     },
37     {
38       title: 'twitter',
39       link: twitterLink?.link || '',
40     },
41     {
42       title: 'instagram',
43       link: instagramLink?.link || '',
44     },
45   ];
46 };

```

```

tests •
frontend > libs > shared > utils > src > lib > formatLinks > test.ts > completeSocialLinks.ts ...
17
18 export const completeSocialLinks = ({
19   socialLink,
20 }: {
21   socialLink: SocialLinks[];
22 }) => {
23
24   if (!socialLink.length) {
25     return socialLinksDefault;
26   }
27
28   const linkLookup: Record<string, string> = socialLink.reduce(
29     (acc, item) => {
30       acc[item.title as keyof typeof acc] = item.link;
31       return acc;
32     },
33   ) as Record<string, string>;
34
35
36   const updatedSocialLinks = socialLinksDefault.map((defaultLink) => {
37     const title = defaultLink.title;
38     const link = linkLookup[defaultLink.title] || '';
39   });
40
41   return updatedSocialLinks;
42 };

```

- It'll be better to change name function name. What is 4 (magic number. We need to use understandable identifier for it) ?

(done)

```

validationPayments.ts 1
frontend > libs > shared > utils > src > lib > payments-functions > validationPayments.ts > isFirstMilestoneAmountValid.ts ...
2 import moment from 'moment/moment';
3
4 export const checkFirstMilestoneAmount = (
5   milestones: MilestonePaymentType[],
6   summaryMilestones: number,
7 ) => {
7   if (milestones.length > 1) {
8     const amount = milestones[0].amount || 0;
9     if (amount < summaryMilestones / 4) {
10       return false;
11     }
12   }
13   return true;
14 };
15
16 export const isFirstMilestoneAmountValid = (
17   milestones: MilestonePaymentType[],
18   summaryMilestones: number,
19 ): boolean => {
20   if (milestones.length <= 1) {
21     return true;
22   }
23
24   const amount = milestones[0].amount ?? 0;
25   return amount >= summaryMilestones / 4;
26 };
27

```

- add typings and make function without inner conditions. We need to change the name of the function

(done)

```

frontend > libs > shared > utils > src > lib > payments-functions > validationPayments.ts > ...
18  export const checkFirstMilestoneMonth = (
19    milestones: MilestonePaymentType[],
20  ) => {
21    if (milestones.length > 1) {
22      const milestonesDates = milestones.map(item =>
23        moment(item.invoiceDate),
24      );
25      const minDate = moment.min(milestonesDates);
26      if (milestonesDates[0] > minDate) {
27        return false;
28      }
29    }
30    return true;
31  };
32
33  export const checkFirstMilestoneMonth = (milestones: MilestonePaymentType[]): boolean => {
34    if (milestones.length <= 1) {
35      return true;
36    }
37    const milestonesDates = milestones.map((item) => moment(item.invoiceDate));
38    const minDate = moment.min(milestonesDates);
39
40    return milestonesDates[0] <= minDate;
41  };
42
43

```

- Lets remove unnecessary code and make it simliar
- (done)

```

frontend > libs > shared > utils > src > lib > payments-functions > calculatePayments.ts > calculateSummaryMilestones
1  import { MilestonePaymentType, RetainerPaymentType } from '@breef/shared/types';
2
3  export const calculateSummaryMilestones = (
4    milestones: MilestonePaymentType[],
5  ) => {
6    if (milestones.length !== 0) {           original
7      return milestones
8        .map(item =>
9          !isNaN(Number(item.amount)) ? Number(item.amount) : 0,
10        )
11        .reduce((acc, curr) => acc + curr);
12    }
13    return 0;
14  };
15
16  export const calculateSummaryMilestones = (milestones: MilestonePaymentType[]): number => {
17    return milestones.reduce((acc, curr) => acc + (Number(curr.amount) || 0), 0);
18  };
19

```

- We need to make this code simple
- (done)

```

14 |     };
15 |
16 |     export const calculateSummaryRetainer = (
17 |       retainer: RetainerPaymentType | null,           Original
18 |     ) => {
19 |       const amount = Number(retainer?.amount);
20 |       const numberOFPayments = Number(retainer?.numberOfPayments);
21 |       if (retainer && !isNaN(amount)) {
22 |         if (retainer.numberOfPayments !== 0 && !isNaN(numberOFPayments)) {
23 |           return amount * numberOFPayments;
24 |         }
25 |         return amount;
26 |       }
27 |       return 0;
28 |     };
29 |
30 |     export const calculateSummaryRetainer = (retainer: RetainerPaymentType | null): number => {
31 |       if (!retainer) {                                Proposition
32 |         return 0;
33 |       }
34 |       const amount = Number(retainer.amount);
35 |       const numberOFPayments = Number(retainer.numberOfPayments);
36 |
37 |       if (isNaN(amount)) {
38 |         return 0;
39 |       }
40 |
41 |       if (numberOfPayments !== 0 && isNaN(numberOfPayments)) {
42 |         return amount;
43 |       }
44 |
45 |       return amount * numberOFPayments;
46 |     };

```

- We can use this changes in useEffect

```

44 |
45 |   const styleError = {
46 |     width: 'max-content',
47 |     backgroundColor: `${colors.mainError}`,
48 |     border: 'none',
49 |     color: `${colors.mainWhite}`,
50 |   };
51 |
52 |   useEffect(() => {                         Original
53 |     if (isError) return setOpen(true);
54 |     return setOpen(false);
55 |   }, [isError]);
56 |
57 |   useEffect(() => {                         Proposition
58 |     setOpen(isError || false);
59 |   }, [isError]);
60 |
61 |   const { x, y, reference, floating, context } = useFloating({
62 |     placement,
63 |     open,
64 |     onOpenChange: !isError ? setOpen : undefined,

```

- We don't need to use !! operator and also we need to check for noteIcon is added to prevent rendering the span element when noteIcon is falsy

```

17 |   className,
18 |   noteIcon,
19 |   subtitle,
20 | }: TitleStepProps) => {
21 |   return (
22 |     <StyledTitleStep className={className}>
23 |       {!step && !numberSteps && (
24 |         <div className="step-number">
25 |           {step}/{numberSteps}
26 |         </div>
27 |       )
28 |       {step && numberSteps && (
29 |         <div className="step-number">
30 |           {step}/{numberSteps}
31 |         </div>
32 |       )}
33 |       <div className="step-title-wrapper">
34 |         <h2 className="step-title">
35 |           {title}
36 |           {noteIcon && <span className="note-icon">{noteIcon}</span>}
37 |           {(noteIcon && <span className="note-icon">{noteIcon}</span>)}
38 |         </h2>
39 |         {subtitle && <h4 className="step-subtitle">{subtitle}</h4>}
40 |       </div>
41 |     </StyledTitleStep>
42 |   );
43 | }
44 |
45 | export default TitleStep;

```

- We can use solution without switch operator

```

useGetTip.tsx
...
19 export const useGetTip = (step: number, type: TipTypeKeys) => {
20   const [tip, setTip] = useState<TipType | TipType[]>({
21     title: '',
22     description: ''
23   });
24
25   useEffect(() => {
26     switch (type) {
27       case TipTypeKeys.projectOverview:
28         setTip(getTipProjectOverview(step));
29         break;
30       case TipTypeKeys.projectPhase:
31         setTip(getTipProjectPhaseStep());
32         break;
33       case TipTypeKeys.projectInfoNotes:
34         setTip(getTipProjectInfoNotes(step));
35         break;
36       case TipTypeKeys.pitchProfile:
37         setTip(getTipPitchProfile(step));
38         break;
39       case TipTypeKeys.pitchBudget:
40         setTip(getTipPitchBudget(step));
41         break;
42       case TipTypeKeys.pitchProject:
43         setTip(getTipPitchProject(step));
44         break;
45       case TipTypeKeys.agencySelection:
46         setTip(getTipAgency(step));
47         break;
48       default:
49         break;
50     }
51   }, [step, type]);
52
53   return tip;
54 };

```

```

tipType.tsx
...
15 type TipHandlers = {
16   projectOverview: () => TipType[];
17   projectPhase: () => TipType;
18   projectInfoNotes: () => TipType;
19   pitchProfile: () => TipType;
20   pitchBudget: () => TipType;
21   pitchProject: () => TipType;
22   agencySelection: () => TipType;
23 };
24
25 export const useGetTip = (step: number, type: TipTypeKeys) => {
26   const [tip, setTip] = useState<TipType | TipType[]>({
27     title: '',
28     description: ''
29   });
30
31   useEffect(() => {
32     const tipHandler = tipHandlers[type];
33     if (tipHandler) {
34       setTip(tipHandler(step));
35     }
36   }, [step, type]);
37
38   return tip;
39 };

```

- We need to refactor TipCard.tsx component. Possible solution

```

TipCard.tsx
...
11
12 export const TipCard = ({ tip, className, logoUrl = BreefAvatar.src, leadFirstName = 'Breef', leadLastName = 'Team' }: TipCardProps) => {
13   const tipArray = Array.isArray(tip) ? tip : [tip];
14   const fixed = tipArray.length === 1;
15
16   return (
17     <StyledTipCard
18       className={className ? `${className} tip-card` : 'tip-card'}
19     >
20       <div className="text-group">
21         {tipArray.map((tip, index) => (
22           <Fragment key={index}>
23             <div className="title-card">{tip.title}</div>
24             <div className="description">{tip.description}</div>
25           </Fragment>
26         ))}
27       {fixed && logoUrl.length >= 1 ?
28         <div className="avtor">
29           {logoUrl && <img src={logoUrl} alt="Avatar" />}
30           <div className="lead-name">{`${leadFirstName} ${leadLastName}`}</div>
31         </div>
32       }
33     </div>
34     <div>
35       {fixed && !fixed && logoUrl.length >= 1 ?
36         <div className="avtor-wrapp">
37           <div className="avtor">
38             {logoUrl && <img src={logoUrl} alt="Avatar" />}
39             <div className="lead-name">{`${leadFirstName} ${leadLastName}`}</div>
40           </div>
41         </div>
42       }
43     </div>
44   );
45 };

```

```

tipType.tsx
...
15 const TipCard = ({ tip, className, logoUrl = BreefAvatar.src, leadFirstName = 'Breef', leadLastName = 'Team' }: TipCardProps) => {
16   const tips = tipArray.length === 1 ? tip : [tip];
17   const isSingleTip = tips.length === 1;
18
19   const renderAuthor = () => (
20     <div className="avtor">{isSingleTip ? 'fixed' : ''}</div>
21     {logoUrl && <img src={logoUrl} alt="Avatar" />}
22     <div className="lead-name">{`${leadFirstName} ${leadLastName}`}</div>
23   );
24
25   const renderTipContent = () => (
26     <div className="text-group">
27       {tips.map((tip) => (
28         <Fragment key={tip.title}>
29           <div className="title-card">{tip.title}</div>
30           <div className="description">{tip.description}</div>
31         </Fragment>
32       ))
33     </div>
34   );
35
36   return (
37     <StyledTipCard
38       className={className ? `${className} tip-card` : 'tip-card'}
39     >
40       {renderTipContent()}
41       {isSingleTip && renderAuthor()}
42     </StyledTipCard>
43   );
44 };

```

- We need to avoid using on unnecessary else and make code simpler (done)

```

TipCard.tsx
...
48   <div>
49     {textAreaProps}
50   </div>
51   const [text, setText] = useState('');
52   const areaWrapperRef = useRef<HTMLDivElement | null>(null);
53   const textAreaRef = useRef<HTMLTextAreaElement | null>(null);
54
55   useEffect(() => {
56     if (value) {
57       setText(value);
58     } else {
59       setText('');
60     }
61   }, [value]);
62
63   useEffect(() => {
64     setText(value || '');
65   }, [value]);

```

```

TextArea.tsx
...
48   <div>
49     {textAreaProps}
50   </div>
51   const [text, setText] = useState('');
52   const textAreaRef = useRef<HTMLTextAreaElement | null>(null);
53
54   useEffect(() => {
55     setText(value || '');
56   }, [value]);

```

- TeamMember.tsx (done)

The screenshot shows three tabs in a code editor: `TeamMembers.tsx`, `profiletypes.ts`, and another `TeamMembers.tsx`. The first tab contains code with several red annotations:

- A red arrow points from the text "it'll be great to use another name for example member" to the word `member` in the line `const ownerUser = teamMembersInfo?.teamMembers.find(f => f.position.match(/owner/i),);`.
- A red arrow points from the text "We need to use id instead of index" to the line `key={key}` in the `teamMembers.map((member, key) => (` block.
- A red arrow points from the text "teamMembers.map((member, key) => (" to the line `teamMembers.map((member, key) => (` in the bottom `TeamMembers.tsx` file.

The `profiletypes.ts` file contains export type definitions:

```

export type TeamMembersMergedResponseType = {
    invites: [
        {
            email: string;
            date: string;
            id: number;
            phoneNumber: string;
            status: string;
        }
    ];
    teamMembers: [
        {
            id: number;
            email: string;
            firstname: string;
            lastName: string;
            phoneNumber?: string;
            position: string;
        }[]
    ];
};

export type SkillsRequestType = {
    email: string;
    resend?: boolean;
};

export type ListRolesResponse = {
    id: string;
    title: string;
}[];

export type BillingDataRequestType = {
    billing_details: [
        {
            legal_name: string;
        }
    ];
};

```

```

TeamMember.tsx
...
test.txt
...

```

Original

```

49     setMemberRole(position || '');
50   }, [position]);
51 
52   return (
53     <StyledTeamMember disabledEvents={disabledEvents}>
54       <div className="left-section">
55         {firstName && lastName && position ? (
56           <>
57             <span className="text">
58               {firstName + '' + lastName}
59             </span>
60             <span className="subtext">{email}</span>
61           </>
62         ) : (
63           <>
64             <span className="text">{email}</span>
65             <span className="subtext">
66               Invited on {dateLastAction}
67             </span>
68           </>
69         )
70       </div>
71       <div className="right-section">
72         {status && status === 'pending' && !position ? (
73           <CustomizableRole
74             value={isMobile ? 'Pending invite'
75             dropdownList={actionsListPendingInvite}
76             customChange={onSelectActionForInvite}
77           />
78         ) : (
79           <DropdownRole
80             value={memberRole || ''}
81             onChange={onChangeMember as ChangeHandler}
82             actionsList={actionsList}
83             isAction={isAction}
84             onlyRemoveMember={onlyRemoveMember}
85           />
86         )
87       </div>
88     </StyledTeamMember>
89   );
90 }

```

Proposition

```

50   const renderLeftSection = () => {
51     if (firstName && lastName && position) {
52       return (
53         <>
54           <span className="text">
55             {firstName + '' + lastName}
56           </span>
57           <span className="subtext">{email}</span>
58         </>
59       );
60     }
61     return (
62       <span className="text">{email}</span>
63       <span className="subtext">Invited on {dateLastAction}</span>
64     );
65   };
66   const renderRightSection = () => {
67     if (status && status === 'pending' && !position) {
68       return (
69         <CustomizableRole
70           value={isMobile ? 'Pending invite'
71             dropdownList={actionsListPendingInvite}
72             customChange={onSelectActionForInvite}
73           />
74         );
75       return (
76         <DropdownRole
77           value={memberRole || ''}
78           onChange={onChangeMember as ChangeHandler}
79           actionsList={actionsList}
80           isAction={isAction}
81           onlyRemoveMember={onlyRemoveMember}
82         />
83       );
84     }
85     return (
86       <StyledTeamMember disabledEvents={disabledEvents}>
87         <div className="left-section">{renderLeftSection()}</div>
88         <div className="right-section">{renderRightSection()}</div>
89       </StyledTeamMember>
90     );
91   };
92 }

```

### • TabsNavigation.tsx

1 this will make our code simpler and we will not use an extra else (done)

```

TabsNavigation.tsx
...

```

```

12   const router = useRouter();
13   const query: { tab?: string; projectId?: string } = router.query;
14   const routerRef = useRef(router);
15 
16   const getQueryParams = (newTab: string) => {
17     if (query.projectId) {
18       return {
19         query: {
20           projectId: query.projectId,
21           tab: newTab,
22         },
23       };
24     } else {
25       return {
26         query: {
27           tab: newTab,
28         },
29       };
30     }
31   };
32 
33   const getQueryParams = (newTab: string) => {
34     return {
35       query: {
36         ...(!query.projectId && { projectId: query.projectId }),
37         tab: newTab,
38       },
39     };
40   };

```

2 this proposition make our solution more readable and we don't use inner if (done)

```

TabsNavigation.tsx
...

```

```

46   // eslint-disable-next-line react/no-hooks/exhaustive-deps
47   [router.pathname],
48 );
49 
50   useEffect(() => {
51     if (!router.isReady) {
52       if (
53         !query.tab ||
54         !config.some(item => item.tab === query.tab) ||
55         config.find(item => item.tab === query.tab)?.disabled
56       ) {
57         onChangeTab(config[0].tab);
58       }
59     }
60   }, [router.isReady, config, onChangeTab, query.tab]);
61 
62   useEffect(() => {
63     if (!router.isReady) {
64       return;
65     }
66 
67     const canSelect =
68       config.find(item => item.tab === query.tab)?.disabled !== true;
69     if (!query.tab || !canSelect) {
70       onChangeTab(config[0].tab);
71     }
72   }, [router.isReady, config, onChangeTab, query.tab]);

```

3 Why we use config here and we get it as props of component (done)

```

TabsNavigation.tsx ●
libs > shared > ui-components > src > lib > tabsNavigation > TabsNavigation.tsx > TabsNavigation
  40
  41   t,
  42   undefined,
  43   { shallow: true },
  44 )
  45 .then(() => null);
  46 // eslint-disable-next-line react-hooks/exhaustive-deps
  47 [router.pathname],
  48 );
  49
  50 useEffect(() => {
  51   if (router.isReady) {
  52     if (
  53       !query.tab ||
  54       !config.some(item => item.tab === query.tab) ||
  55       config.find(item => item.tab === query.tab)?.disabled
  56     ) {
  57       onChangeTab(config[0].tab);
  58     }
  59   }
  60 }, [router.isReady, config, onChangeTab, query.tab]);
  61

```

4 in this case won't render unnecessary code and also we need to change (**done**)

```

TabsNavigation.tsx ●
libs > shared > ui-components > src > lib > tabsNavigation > TabsNavigation.tsx > TabsNavigation
  59 ), [router.isReady, config, onChangeTab, query.tab];
  60
  61 return (
  62   <StyledTabsNavigation className="tabs-navigation">
  63     <div className="tabs-container">
  64       {config.map((tab, key) => (
  65         return (
  66           <TabButton
  67             key={key}
  68             title={tab.title}
  69             onClick={() => onChangeTab(tab.tab)}
  70             isActive={query.tab === tab.tab}
  71             disabled={tab.disabled} || false
  72           />
  73         );
  74       ))
  75     )
  76   </div>
  77   <div className="row" />
  78 </StyledTabsNavigation>;
  79 )
  80
  81 if (!config) {
  82   return null;
  83 }
  84
  85 return (
  86   <StyledTabsNavigation className="tabs-navigation">
  87     <div className="tabs-container">
  88       {config.map(tab => (
  89         <TabButton
  90           key={tab.title}
  91           title={tab.title}
  92           onClick={() => onChangeTab(tab.tab)}
  93           isActive={query.tab === tab.tab}
  94           disabled={tab.disabled}
  95         />
  96       ))
  97     )
  98   </div>
  99   <div className="row" />
100 </StyledTabsNavigation>;
101

```

• in this case we'll get better clarity and readability (**done**)

```

AccentNumber.tsx 2 ●
> shared > ui-components > src > lib > accentNumber > AccentNumber.tsx > AccentNumber > [transformation]
  1 import { StyledAccentNumber } from './AccentNumber.styled';
  2
  3 export function AccentNumber({
  4   number,
  5   isOptional,
  6 }: {
  7   number: number;
  8   isOptional?: boolean;
  9 }) {
 10   const transformToAccentFormat = (number: number) => {
 11     const stringValue = number + '';
 12     return '0'.repeat(3 - stringValue.length) + stringValue;
 13   };
 14
 15   const transformToAccentFormat = (number: number) => {
 16     const value = String(number);
 17     return '0'.repeat(3 - value.length) + value;
 18   };
 19

```

- I think in this case router is not optional 2 we don't need to pass empty string (**done**)

```

1  // AcceptTerms.tsx
2  import { FieldCheckbox } from '@breef/shared/ui-components';
3
4  type AcceptTermsType = {
5    onChange: (value: React.ChangeEvent) => void;
6    value: boolean;
7  };
8
9  const AcceptTerms: React.FC<AcceptTermsType> = ({ onChange, value }) => {
10  const router = useRouter();
11  return (
12    <StyledAcceptPrivacy>
13      <FieldCheckbox
14        className="accept-privacy"
15        data-testid="accept-privacy"
16        checked={value}
17        onChange={onChange}
18        value={value}
19        label="I agree to Breef's terms of use"
20        isDisabled={false}
21      >
22        <p>I agree to Breef's<input checked="" type="checkbox"/></p>
23        <a href={router basePath + TERMS_OF_USE_ROUTE} target="_blank" rel="noopener noreferrer">terms of use</a>
24        and<input checked="" type="checkbox"/>
25        <a href={router basePath + PRIVACY_POLICY_ROUTE} target="_blank" rel="noopener noreferrer">Privacy Policy</a>
26      </FieldCheckbox>
27    </StyledAcceptPrivacy>
28  );
29  export default AcceptTerms;
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55

```

- We need to change names in interface. Then our code will be descriptive (**done**)

```

1  // Accordion.tsx
2  import { ChevronIcon, SmallRocket } from '@breef/shared/assets';
3  import { AnimatePresence, motion } from 'framer-motion';
4
5  interface AccordionProps {
6    title: string | ReactNode;
7    children?: ReactNode;
8    isShowImage?: boolean;
9    defaultIsOpen?: boolean;
10   isAccent?: boolean;
11   closeAccordion?: boolean;
12   idx?: string;
13   isTriangle?: boolean;
14   setAccordionIsOpen?: (isOpen: boolean) => void;
15 }
16
17
18  export function Accordion({
19    title,
20    isShowImage,
21    defaultIsOpen,
22    isAccent,
23    closeAccordion,
24    children,
25    idx,
26    isTriangle = false,
27    setAccordionIsOpen,
28  }: AccordionProps) {
29  const [isOpen, setIsOpen] = useState(false);
30  const accordionInnerRef = useRef<HTMLDivElement>(null);
31  const transitionTime = 500;
32
33  useEffect(() => {
34    if (closeAccordion) {
35      setIsOpen(false);
36      setAccordionIsOpen && setAccordionIsOpen(false);
37    }
38  }, [closeAccordion]);
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55

```

- it's not great to use index as key (**done**)

```

1  // Answers.tsx
2  import { styledAnswers } from '@breef/shared/ui-components';
3
4  <StyledAnswers
5    className="answers-block"
6    imagePosition={image.position}
7  >
8    <div className="answers-sidebar">
9      <h2>{isMobile && titleMobile ? titleMobile : title}</h2>
10     <Link href={FAQ_ROUTE}>{'Check out \nmore FAQs'}/>
11     <img src={image.src} alt="FAQ" />
12   </div>
13   <div
14     data-testid="answers-wrapper"
15     className="answers-list-wrapper"
16   >
17     {answersData.map((item, key) => (
18       <AnswerItem
19         key={key}
20         title={item.title}
21         description={item.description}
22       >
23       </AnswerItem>
24     )));
25   </div>
26 </StyledAnswers>
27 <StyledAnswersWrapper>
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55

```

- I think in current case it'll be better to use one of next solutions (screen 2, 3). In this case we'll get stronger typings (**done**)

```
const clientActionButtonCtaStatuses: ActionButtonConfigType = {
  activeProjects: {
    text: 'View Projects',
    description: `YOU HAVE ${projectsCount} ACTIVE PROJECTS`,
    onClick: () => scrollToProjectSettingsBar(),
    imageConfig: {
      imageUrl: ctаГlobeImage.src,
      position: {
        top: 0,
        right: 20,
      },
    },
  },
  projectPlaning: {
    text: 'Grab Time',
    description: 'BOOK A PLANNING CALL',
    onClick: () => toggleBookACallPopup(),
    imageConfig: {
    },
  },
};

export type ActionButtonNameType =
  | 'activeProjects'
  | 'projectPlaning'
  | 'startProject';

export type ActionButtonConfigType1 = {
  text: string | ReactNode;
  description?: string;
  descriptionMore?: string;
  onClick?: (meta: MetadataType) => void;
  imageConfig?: {
    imageUrl: string;
    position: {
      top?: number;
      bottom?: number;
      right?: number;
    };
  };
};

export type test = {
  [k in ActionButtonNameType]?: ActionButtonConfigType1;
};

// or
export type clientActionButtonCtaStatuses =
  | 'activeProjects'
  | 'projectPlaning'
  | 'startProject';

export type agencyActionButtonCtaStatuses = 'activeProjects' | 'signUp' | 'incompleteProfile';

export type ActionButtonConfigType2 = {
  text: string | ReactNode;
  description?: string;
  descriptionMore?: string;
  onClick?: (meta: MetadataType) => void;
  imageConfig: {
    imageUrl: string;
    position: {
      top?: number;
      bottom?: number;
      right?: number;
    };
  };
};

type PartialRecord<K extends keyof any, T> = {
  [P in K]?: T
}

type clientActionButtonCtaStatusesType = PartialRecord<clientActionButtonCtaStatuses, ActionButtonConfigType2>;
type agencyActionButtonCtaStatusesType = PartialRecord<agencyActionButtonCtaStatuses, ActionButtonConfigType2>;
type client = Record<clientActionButtonCtaStatuses, ActionButtonConfigType1>;
```

- The same problem (done)

```

    useActionButtonProjectConfig.tsx
    ...
    41     const { changePage } = useRouteControl();
    42     const clientProjectActionButtonStatuses: ActionButtonConfigType = {
    43         continuePost: {
    44             text: 'Review + Post',
    45             description: 'POST YOUR PROJECT',
    46             onClick: meta => {
    47                 resetProjectEdit();
    48                 changePage(
    49                     PROJECT_EDIT_ROUTE.reverse({
    50                         projectId: meta.projectId || 0,
    51                     }) || '',
    52                 );
    53             },
    54             imageConfig: {
    55                 imageUrl: ctaManOrangeImage.src,
    56                 position: {
    57                     right: 0,
    58                     bottom: 12,
    59                 },
    60             },
    61         },
    ...
  
```

- I propose to simplify the current solution and also change the variable names to improve the readability of the code. (done)

```

    useCheckIsContainsUrlPath.tsx
    ...
    1 import { useRouter } from 'next/router';
    2 import React, { useState } from 'react';
    3
    4 export const useCheckIsContainsUrlPath = ({(
    5     pathNames,
    6 ): {
    7     pathNames: string[];
    8 }) => {
    9     const router = useRouter();
    10     const [isContainsPathName, setIsContainsPathName] = useState(false);
    11
    12     const checkIsContainsUrlPath = () => {
    13         const arrLength = pathNames.length - 1;
    14         for (let i = 0; i < arrLength; i++) {
    15             if (router.pathname.includes(pathNames[i])) {
    16                 setIsContainsPathName(true);
    17                 return;
    18             }
    19             setIsContainsPathName(false);
    20         }
    21     };
    22
    23     useEffect(() => {
    24         if (pathNames.length) {
    25             checkIsContainsUrlPath();
    26         }
    27     }, [pathNames]);
    28
    29     return { isContainsPathName };
    30 };
  
```

```

    testtsx 1.0
    ...
    1 import { useRouter } from 'next/router';
    2 import { useState } from 'react';
    3
    4 export const useUrlPathContains = ((pathNames): (pathNames: string[]) => {
    5     const [hasMatchedPath, setHasMatchedPath] = useState(false);
    6
    7     const checkIsContainsUrlPath = () => {
    8         const containsPath = pathNames.some(pathName =>
    9             router.pathname.includes(pathName),
    10         );
    11         setHasMatchedPath(containsPath);
    12     };
    13
    14     useEffect(() => {
    15         if (pathNames.length) {
    16             checkIsContainsUrlPath();
    17         }
    18     }, [pathNames]);
    19
    20     return { hasMatchedPath };
    21 };
    22
    23
  
```

- NEW

- Добавил интерфейс, изменил название переменной

```

    AccentNumber.tsx
    ...
    1 import { StyledAccentNumber } from './AccentNumber.styled';
    2
    3 export function AccentNumber({(
    4     number,
    5     isOptional,
    6 ): {
    7     number: number;
    8     isOptional: boolean;
    9 }) => {
    ...
    14     const transformToAccentFormat = (number: number) => {
    15         const stringValue = number + '';
    16         return `0.${repeat(3 - stringValue.length) + stringValue}`;
    17     };
    18     return (
    19         <StyledAccentNumber className="accent-number">
    20             <span className="round" />
    21             <span className="number">
    22                 {transformToAccentFormat(number)}
    23                 {isOptional && '(optional)'}
    24             </span>
    25         </StyledAccentNumber>
    26     );
    27 };
  
```

```

    testtsx U
    ...
    1 import { StyledAccentNumber } from './AccentNumber.styled';
    2
    3 function transformToAccentFormat(number: number): string {
    4     const formattedNumber = number.toString();
    5     return `0.${repeat(3 - formattedNumber.length) + formattedNumber}`;
    6 }
    7
    8 interface AccentNumberProps {
    9     number: number;
    10     isOptional?: boolean;
    11 }
    12
    13 export function AccentNumber({ number, isOptional }: AccentNumberProps) {
    14     return (
    15         <StyledAccentNumber className="accent-number">
    16             <span className="round" />
    17             <span className="number" />
    18                 {transformToAccentFormat(number)}
    19                 {isOptional && '(optional)'}
    20             </span>
    21         </StyledAccentNumber>
    22     );
    23 }
    24
  
```

- Возможно pageTitle будет лучше именем

```
Head.tsx 1 ●
libs > shared > ui-components > src > lib > animateLayoutPage > head > Head.tsx > HeadApp
1 import Head from 'next/head';
2
3 export default function HeadApp({ title }: { title: string }) {
4   const titleContent = `breef. | ${title}`;
5
6   const pageTitle = `breef. | ${title}`; ↑
7   return (
8     <Head>
9       <title>{titleContent}</title>
10      <meta
11        key="viewport"
12        name="viewport"
13        content="width=device-width, initial-scale=1, shrink-to-fit=no"
14      />
15      <link rel="icon" type="image/x-icon" href="/favicon.ico" />
16      <link rel="preload" as="font" />
17    </Head>
18  );
19}
20
```

- Поменять имя интерфейса (соответствует рекомендациям React) добавить функцию обработки события

AnswerItem.tsx 2

```
libs/shared/ui-components/src/lib/answers/answeritem/AnswerItem.tsx
```

```
6 interface AnswersProps {  
7   title: string;  
8   description: string;  
9 }  
10  
11 interface AnswerItemProps { 1  
12   title: string;  
13   description: string;  
14 }  
15  
16 export function AnswerItem({ title, description }: AnswersProps) {  
17   const [isOpen, setIsOpen] = useState(false);  
18  
19   const toggleOpen = () => { 2  
20     setIsOpen(!isOpen);  
21   };  
22  
23   return (  
24     <StyledAnswerItem  
25       data-testid="answer-item"  
26       onClick={() => setIsOpen(!isOpen)}  
27       isOpen={isOpen}  
28     >  
29       <div className="answer-header">  
30         <h2>{title}</h2>  
31         <ChevronSmallStraight />  
32       </div>  
33       <div>  
34         <p>{description}</p>  
35       </div>  
36     </StyledAnswerItem>  
37   );  
38 }  
39  
40 // This is a styled component.  
41 // It's generated by the styled-components library.  
42 // You can learn more about it here:  
43 // https://www.styled-components.com/docs/usage#using-a-component
```

- вынес обработку заголовка в отдельную переменную, не использовать `index(key)` в качестве `key` property

```
  ● AnswerItem.tsx    ● Answers.tsx 1    ● ...    ● teststsx

libs > shared > ui-components > src > lib > answers > ● Answers.tsx > ● Answers
27  : AnswersProps) {
28    const { isMobile } = useContext(MediaContext);
29
30    const renderTitle = isMobile && titleMobile ? titleMobile : title;
31
32    return (
33      <StyledAnswersWrapper>
34        <StyledAnswers
35          className="answers-block"
36          imagePosition={image.position}
37        >
38          <div className="answers-sidebar">
39            <h2>{isMobile && titleMobile ? titleMobile : title}</h2>
40            <a href={FAQ_ROUTE}>{"Check out \nmore FAQs!"}</a>
41            <img src={image.src} alt="FAQ" />
42          </div>
43          <div
44            data-testid="answers-wrapper"
45            className="answers-list-wrapper"
46          >
47            {answersData.map((item, key) => (
48              <AnswerItem
49                key={key}
50                title={item.title}
51                description={item.description}
52              >
53            )));
54          
```

- МОЖНО ДОБАВИТЬ НОВЫЙ ИНТЕРФЕЙС, СТОИТ ВЫНЕСТИ В ОТДЕЛЬНЫЙ обработчик клика

```
AnswerItem.tsx    Answers.tsx    MultipleAutocomplete.tsx 1
libs > shared > ui-components > src > lib > autocomplete > multipleAutocomplete > MultipleAutocomplete.tsx 1
1  import { TransformAddressType } from '../transformGoogleResult';
2  import PlacesAutocomplete from './Autocomplete';
3  import LinkButton from '../../../../../button/linkButton/LinkButton';
4  import { AuthGoogleType } from '@breef/shared/types';
5  import { StyledMultipleAutocomplete } from './MultipleAutocomplete.styled';
6  import { toast } from 'react-toastify';
7
8  interface Place {
9    location: string;
10 }
11
12 interface MultipleAutocompleteProps {
13   places: { location: string }[];
14   onChange: (places: unknown) => void;
15   onClick: (
16     key: string,
17     data: AuthGoogleType | React.SyntheticEvent | { query: string },
18   ) => void;
19   error?: string;
20   isDisableNextBtn?: boolean;
21 }
22
```

```
AnswerItem.tsx    Answers.tsx    MultipleAutocomplete.tsx 1
libs > shared > ui-components > src > lib > autocomplete > multipleAutocomplete > MultipleAutocomplete.tsx 1
52
53   return (
54     <StyledMultipleAutocomplete
55       data-testid="multiple-places-autocomplete"
56       className="multiple-autocomplete"
57     >
58       {places.map((item, key) => (
59         <PlacesAutocomplete
60           values={places}
61           data-testid={`places-autocomplete-${key}`}
62           key={key}
63           value={item.location}
64           onClick={(id, name, address, error) =>
65             error
66             ? showError(error)
67             : onSelectPlace(key, { id, name, address })
68         }
69       ))
70     )
71   );
72
```

```
AnswerItem.tsx    Answers.tsx    MultipleAutocomplete.tsx 1
libs > shared > ui-components > src > lib > autocomplete > multipleAutocomplete > MultipleAutocomplete.tsx 1
56   className="multiple-autocomplete"
57   >
58     {places.map((item, key) => (
59       <PlacesAutocomplete
60         values={places}
61         data-testid={`places-autocomplete-${key}`}
62         key={key}
63         value={item.location}
64         onClick={(id, name, address, error) =>
65           error
66             ? showError(error)
67             : onSelectPlace(key, { id, name, address })
68         }
69       )
70     ))
71   );
72   isArrowNext={!key}
73   isDisableNext={isDisableNextBtn}
74   onNextButton={e => onClick('nav-event', e)}
75   onDeleteSelf={key ? () => onDeleteLocation(key) : undefined}
76   />
77   {places.length < 3 && (
78     <LinkButton
79       data-testid="add-office-btn"
80       name="Office"
81       icon="plus"
82       onClick={addNew}
83       size="big"
84       className="link-button"
85     >
```

- вынести remove и add в отдельный тип

- можно вынести рендер иконок в отдельную функцию

```
export const ChipButton = ({  
  id,  
  name,  
  action,  
  icon = true,  
  onClick,  
  disabled = false,  
}: ChipButtonProps) => {  
  const handleClick = (event: React.SyntheticEvent) => {  
    onClick(event, action, id);  
  };  
  return (  
    <StyledChipButton  
      isDisabled={disabled}  
      actions={action}  
      icon={icon}  
      onClick={handleClick}  
      data-testid="chip-button"  
    >  
      {name}  
      {icon && action === 'remove' && (  
        <CloseIcon className="close-icon" />  
      )}  
      {icon && action === 'add' && <PlusIcon className="plus-icon" />}  
    </StyledChipButton>  
  );  
};  
  
export default ChipButton;
```

```
19   id,  
20   name,  
21   action,  
22   icon = true,  
23   onClick,  
24   disabled = false,  
25 }: ChipButtonProps) => {  
26   const handleClick = (event: SyntheticEvent) => {  
27     onClick(event, action, id);  
28   };  
29  
30   const renderIcon = () => {  
31     if (!icon) return null;  
32  
33     if (action === 'remove') {  
34       return <CloseIcon className="close-icon" />;  
35     }  
36  
37     return <PlusIcon className="plus-icon" />;  
38   };  
39  
40   return (  
41     <StyledChipButton  
42       isDisabled={disabled}  
43       action={action}  
44       icon={icon}  
45       onClick={handleClick}  
46       data-testid="chip-button"  
47     >  
48       {name}  
49       {renderIcon()}  
50     </StyledChipButton>  
51   );  
52 };
```

- можно не использовать анонимную функцию

```
GoogleAuth.tsx
libs > shared > ui-components > src > lib > button > googleAuth > GoogleAuth.tsx > GoogleAuth

13  export function GoogleAuth({
14    name = 'Continue with Google',
15    onClick,
16  }: GoogleAuthProps) {
17    const { loginImplicit, userCredential } = useGoogleLoginMethods();
18
19    useEffect(() => {
20      if (userCredential) {
21        onClick(userCredential);
22      }
23      // eslint-disable-next-line react-hooks/exhaustive-deps
24    }, [userCredential]);
25
26    return (
27      <StyledGoogleForm className="google-identity" data-testid="google-auth">
28        <CustomGoogleButton onClick={() => loginImplicit()} title={name} />
29      </StyledGoogleForm>
30    );
31  }
32
33  export default GoogleAuth;
34
```

- МОЖНО ВЫНЕСТИ В ОТДЕЛЬНЫЙ ТИП

SEARCH

Replace

13 results in 13 files · Open in editor

- formConfiguration.tsx
- Button.tsx
- LinkButton.tsx
- SaveButton.tsx
- SocialButton.tsx
- FieldInput.spect.tsx
- FieldInput.tsx
- FieldSelect.tsx
- FieldSelectDefinedList.tsx
- FieldSelectList.tsx
- DefaultPopup.tsx
- FieldController.tsx
- FieldDetection.tsx

type TypeButton = 'Submit' | 'button'

export type TypeButton = 'Submit' | 'button'

type? 'Submit' | 'button'

- НУЖНО ПОЧИСТИТЬ timer

GoogleAuth.tsx

SaveButton.tsx 1

```

    33
    34     useEffect(() => {
    35         setIsSuccessState(isSuccess);
    36     }, [isSuccess]);
    37
    38     useEffect(() => {
    39         if (isSuccessState) {
    40             setTimeout(() => {
    41                 setIsSuccessState(false);
    42             }, 3000);
    43         }
    44     }, [isSuccessState]);
    45
    46     useEffect(() => {
    47         if (isSuccessState) {
    48             const timer = setTimeout(() => {
    49                 setIsSuccessState(false);
    50             }, 3000);
    51             return () => clearTimeout(timer);
    52         }
    53     }, [isSuccessState]);

```

proposition

BookACall.tsx

```

    45
    46     useEffect(() => [
    47         getMail && selfData && getEmail(selfData.email),
    48     ], [selfData]);
    49     const isLoading = isLoadingSelf || isLoadingProfile;
    50
    51     return (
    52         <StyledBookACall>
    53             data-testid="book-a-call"
    54             className="calendly"
    55             calendlyHeight={height}
    56             calendlyBorder={isCalendlyBorder}
    57             >
    58                 {(!isLoading && selfData && companyData) ? (
    59                     <CalendlyWidget nowko искониаат ?? вместо первоначального неопределено
    60                     calendlyLabel={
    61                         calendlyUrl
    62                             ? calendlyUrl
    63                             : companyData.brandLead.brandLead.calendlyLink
    64                     }
    65                     calendlyEvents={calendlyEvents}
    66                     onChange={handleChangeCalendly}
    67                     email={selfData.email}
    68                     height={height}
    69                     companyName={companyData.companyName}
    70                     phone={selfData.phone}&selfData.phoneNumber}
    71                     firstname={selfData.firstname}
    72                     lastname={selfData.lastname}
    73                     projectTypes={companyData.projectType}
    74                     .map(item => item.name)
    75                     .join(',')
    76                     isBookedCall={isBookedCall}
    77                 )
    78             ) : (
    79                 <Spinner class="

```

You have Windows Subsystem for Linux (WSL) installed on

- странное решение



```

    ...
    customAnswers: {
      a1: companyName,
      a2: phoneNumber,
      a3: projectTypes,
    },
  });

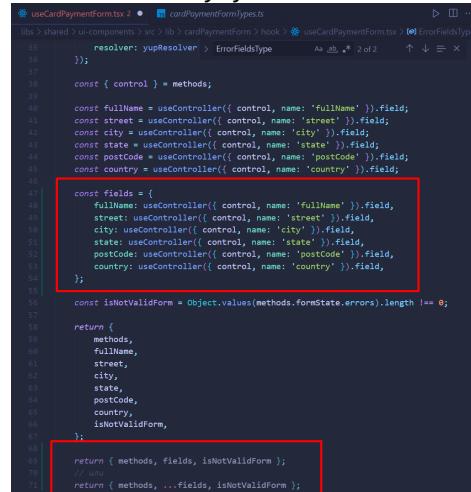
  ...

  ? function getHeightToEnsureNoCrop() {
    if (!isTimeSelected) return { height: height };
    if (isDoneScheduling) return { height: height };
    return { height: height };
  }

  ...

```

- МОЖНО НЕМНОГО УЛУЧШИТЬ ЧИТАЕМОСТЬ КОДА,



```

    ...
    const fields = {
      fullName: useController({ control, name: 'fullName' }).field,
      street: useController({ control, name: 'street' }).field,
      city: useController({ control, name: 'city' }).field,
      state: useController({ control, name: 'state' }).field,
      postCode: useController({ control, name: 'postCode' }).field,
      country: useController({ control, name: 'country' }).field,
    };

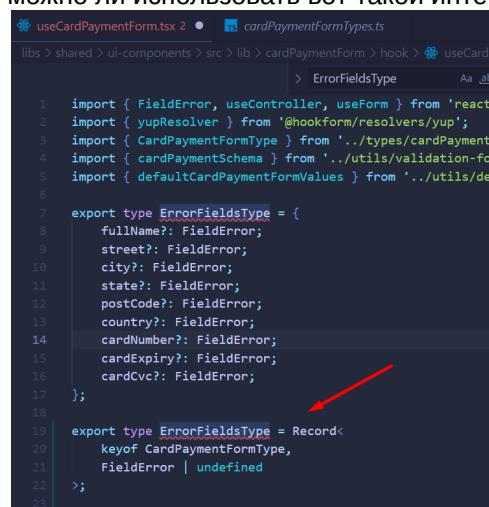
    const isNotValidForm = Object.values(methods.formState.errors).length === 8;
  ...

  return {
    methods,
    fullname,
    street,
    city,
    state,
    postCode,
    country,
    isNotValidForm,
  };
}

return { methods, fields, isNotValidForm };
}

```

- МОЖНО ЛИ ИСПОЛЬЗОВАТЬ ВОТ ТАКОЙ ИНТЕРФЕЙС ИЛИ НЕТ



```

    ...
    import { FieldError, useForm } from 'react-hook-form';
    import { yupResolver } from '@hookform/resolvers/yup';
    import { CardPaymentFormType } from '../types/cardPaymentForm';
    import { cardPaymentSchema } from '../utils/validation-form';
    import { defaultCardPaymentFormValues } from '../utils/default';

    export type ErrorFieldsType = {
      fullName?: FieldError;
      street?: FieldError;
      city?: FieldError;
      state?: FieldError;
      postCode?: FieldError;
      country?: FieldError;
      cardNumber?: FieldError;
      cardExpiry?: FieldError;
      cardCvc?: FieldError;
    };

    export type ErrorFieldsType = Record<
      keyof CardPaymentFormType,
      FieldError | undefined
    >;
  ...

```

Clean (code-level)	OK	
Security (code-level)	OK	
Tests	FAIL	
Documentation (code-level)	HAS ISSUE S	
Author	<sup>1</sup> Vakula Pavel	

---

<sup>1</sup><https://confluence.light-it.tools/display/~vakulapavel0>