Было так. Но кажется много кода?

```
import { DocumentInfo } from 'models/documents/api'
import { cc } from 'utils/combineClasses'
import EditMenu from './EditMenu'
import { default as FileMenu, Copy, Create, Delete, Share, Download } from
'./FileMenu'
import InsertMenu from './InsertMenu'
import styles from './styles.module.scss'
interface Props {
  documentInfo: DocumentInfo
  className?: string
}
const DocumentMenu = ({ documentInfo, className }: Props) => {
  return (
    <div className={cc(styles.documentMenuRoot, className)}>
      <FileMenu documentInfo={documentInfo}/>
      <EditMenu />
      <InsertMenu documentInfo={documentInfo} />
    </div>
  )
}
```

```
import { Icons } from 'constants/icons'
import { Permissions } from 'constants/permissions'
import { default as CustomButton } from 'components/ui/Button'
import CustomAriaButton from 'components/ui/CustomAriaButton'
import CustomSVG from 'components/ui/CustomSVG'
import { useRef, useState } from 'react'
import { useAbac } from 'react-abac'
import { Menu, MenuItem, MenuTrigger, Popover } from 'react-aria-components'
import { useTranslation } from 'react-i18next'
import { useClickAway } from 'react-use'
import { ModalType } from 'stores/modalStore'
import { DocumentInfo } from 'models/documents/api'
import { cc } from 'utils/combineClasses'
import styles from './styles.module.scss'
import { useDocumentMenu } from './useDocumentMenu'

interface Props {
  documentInfo: DocumentInfo
}
```

```
const FileMenu = ({ documentInfo }: Props) => {
  const [showModal, closeModal] = useModalStore(state => [state.showModal,
state.closeModal])
  const queryClient = useQueryClient()
  const navigate = useNavigate()

  const { mutate: createDocument } = useMutation({
    mutationFn: (data: CreateDocumentRequest) => postDocument(data),
    onSuccess: data => {
      queryClient.invalidateQueries({ queryKey: [QueryKeys.DOCUMENTS] })
      navigate({ to: Paths.DOCUMENT, params: { documentId: data.id } })
    }
  })

  const onCreateDocument = () => {
    createDocument({ title: 'New Document' })
  }

  const { mutate: deleteDocumentMutate } = useMutation({
    mutationFn: (id: string) => deleteDocument(id),
    onSuccess: () => {
      queryClient.invalidateQueries({ queryKey: [QueryKeys.DOCUMENTS] })
      queryClient.invalidateQueries({ queryKey: [QueryKeys.DOCUMENTS_SEARCH]
})
      navigate({ to: Paths.DOCUMENTS })
    }
  })

  const onDeleteDocument = () => {
    deleteDocumentMutate(documentInfo.id)
  }

  const { t } = useTranslation(['document', 'modals'])
  const { userHasPermissions } = useAbac()
  const { showModal, closeModal, onDeleteDocument, onCreateDocument } =
    useDocumentMenu(documentInfo)

  const [isOpen, setIsOpen] = useState(false)
  const triggerRef = useRef(null)

  useClickAway(triggerRef, () => {
    setIsOpen(false)
  })

  return (
    <MenuTrigger>
      <CustomAriaButton
        onPress={() => setIsOpen(state => !state)}
        ref={triggerRef}
        className={cc(styles.menuTrigger, isOpen && styles.active)}
      >
        {t('document:menu.file.title')}
      </CustomAriaButton>
      <Popover triggerRef={triggerRef} isNonModal isOpen={isOpen}>
        <Menu onAction={() => setIsOpen(false)} className={styles.menu}>
          <MenuItem onAction={onCreateDocument} className={styles.menuItem}>
            <CustomSVG className={cc(styles.menuOptionIcon)}>
              {Icons.document.createDocument}
            </CustomSVG>
            <p>{t('document:menu.file.createDocument')}</p>
          </MenuItem>
          <MenuItem
            onAction={() => showModal(ModalType.COPY_DOCUMENT, {
documentInfo })}
            className={styles.menuItem}
          >
            <CustomSVG className={cc(styles.menuOptionIcon)}>
              {Icons.document.copyDocument}
            </CustomSVG>
            <p>{t('document:menu.file.copyDocument')}</p>
          </MenuItem>
          <MenuItem
            onAction={() => showModal(ModalType.SHARE, { title:
documentInfo.title, documentInfo })}
            className={styles.menuItem}
          >
            <CustomSVG className={cc(styles.menuOptionIcon)}>
              {Icons.document.documentShare}
            </CustomSVG>
            <p>{t('document:menu.file.share')}</p>
          </MenuItem>
          {userHasPermissions(Permissions.DELETE_DOCUMENT,
documentInfo.userRole) && (
            <MenuItem
              className={styles.menuItem}
              onAction={() => {
                showModal(ModalType.CUSTOM, {
                  title: t('document:menu.file.deleteDocument'),
                  children: (
                    <>
                      <p className={styles.modalMessage}>
```

```
                        {t('modals:moveToTrash.description', { title:
document.title })}
                      </p>
                      <div className={styles.modalButtonsWrapper}>
                        <CustomButton variant="secondary" onPress=
{closeModal}>
                          {t('common:actions.cancel')}
                        </CustomButton>
                        <CustomButton
                          onPress={() => {
                            onDeleteDocument()
                            closeModal()
                          }}
                        >
                          {t('common:actions.delete')}
                        </CustomButton>
                      </div>
                    </>
                  )
                })
              }}
            >
              <CustomSVG className={styles.menuOptionIcon}>
{Icons.common.trash}</CustomSVG>
              <p>{t('document:menu.file.deleteDocument')}</p>
            </MenuItem>
          )}
        </Menu>
      </Popover>
    </MenuTrigger>
  )
}

export default FileMenu
```

Вот так стало, но проще ли?

```
import { Icons } from 'constants/icons'
import { Permissions } from 'constants/permissions'
import { default as CustomButton } from 'components/ui/Button'
import CustomAriaButton from 'components/ui/CustomAriaButton'
import CustomSVG from 'components/ui/CustomSVG'
import { useRef, useState } from 'react'
import { useAbac } from 'react-abac'
import { Menu, MenuItem, MenuTrigger, Popover } from 'react-aria-components'
import { useTranslation } from 'react-i18next'
import { useClickAway } from 'react-use'
import { ModalType } from 'stores/modalStore'
import { DocumentInfo } from 'models/documents/api'
import { cc } from 'utils/combineClasses'
import styles from './styles.module.scss'
import { useDocumentMenu } from './useDocumentMenu'

interface Props {
  documentInfo: DocumentInfo
}

const FileMenu = ({ documentInfo }: Props) => {
  const { t } = useTranslation(['document', 'modals'])
  const { userHasPermissions } = useAbac()
  const { showModal, closeModal, onDeleteDocument, onCreateDocument } =
    useDocumentMenu(documentInfo)

  const [isOpen, setIsOpen] = useState(false)
  const triggerRef = useRef(null)

  useClickAway(triggerRef, () => {
    setIsOpen(false)
  })

  return (
    <MenuTrigger>
      <CustomAriaButton
        onPress={() => setIsOpen(state => !state)}
        ref={triggerRef}
        className={cc(styles.menuTrigger, isOpen && styles.active)}
      >
        {t('document:menu.file.title')}
      </CustomAriaButton>
      <Popover triggerRef={triggerRef} isNonModal isOpen={isOpen}>
        <Menu onAction={() => setIsOpen(false)} className={styles.menu}>
          <MenuItem onAction={onCreateDocument} className={styles.menuItem}>
            <CustomSVG className={cc(styles.menuOptionIcon)}>
              {Icons.document.createDocument}
            </CustomSVG>
            <p>{t('document:menu.file.createDocument')}</p>
          </MenuItem>
          <MenuItem
            onAction={() => showModal(ModalType.COPY_DOCUMENT, {
documentInfo })}
```

```
            className={styles.menuItem}
          >
            <CustomSVG className={cc(styles.menuOptionIcon)}>
              {Icons.document.copyDocument}
            </CustomSVG>
            <p>{t('document:menu.file.copyDocument')}</p>
          </MenuItem>
          <MenuItem
            onAction={() => showModal(ModalType.SHARE, { title:
documentInfo.title, documentInfo })}
            className={styles.menuItem}
          >
            <CustomSVG className={cc(styles.menuOptionIcon)}>
              {Icons.document.documentShare}
            </CustomSVG>
            <p>{t('document:menu.file.share')}</p>
          </MenuItem>
          {userHasPermissions(Permissions.DELETE_DOCUMENT,
documentInfo.userRole) && (
            <MenuItem
              className={styles.menuItem}
              onAction={() => {
                showModal(ModalType.CUSTOM, {
                  title: t('document:menu.file.deleteDocument'),
                  children: (
                    <>
                      <p className={styles.modalMessage}>
                        {t('modals:moveToTrash.description', { title:
document.title })}
                      </p>
                      <div className={styles.modalButtonsWrapper}>
                        <CustomButton variant="secondary" onPress=
{closeModal}>

                          {t('common:actions.cancel')}
                        </CustomButton>
                        <CustomButton
                          onPress={() => {
                            onDeleteDocument()
                            closeModal()
                          }}
                        >
                          {t('common:actions.delete')}
                        </CustomButton>
                      </div>
                    </>
                  )
                })
              }}
            >
              <CustomSVG className={styles.menuOptionIcon}>
{Icons.common.trash}</CustomSVG>
              <p>{t('document:menu.file.deleteDocument')}</p>
            </MenuItem>
          )}
        </Menu>
      </Popover>
    </MenuTrigger>
  )
}

export default FileMenu
```

```
import { QueryKeys } from 'constants/queryKeys'
import { Paths } from 'constants/routes'
import { useMutation, useQueryClient } from '@tanstack/react-query'
import { useNavigate } from '@tanstack/react-router'
import { useModalStore } from 'stores/modalStore'
import { deleteDocument, postDocument } from 'api/documents'
import { CreateDocumentRequest, DocumentInfo } from 'models/documents/api'

export const useDocumentMenu = (documentInfo: DocumentInfo) => {
  const [showModal, closeModal] = useModalStore(state => [state.showModal,
state.closeModal])
  const queryClient = useQueryClient()
  const navigate = useNavigate()

  const { mutate: createDocument } = useMutation({
    mutationFn: (data: CreateDocumentRequest) => postDocument(data),
    onSuccess: data => {
      queryClient.invalidateQueries({ queryKey: [QueryKeys.DOCUMENTS] })
      navigate({ to: Paths.DOCUMENT, params: { documentId: data.id } })
    }
  })

  const onCreateDocument = () => {
    createDocument({ title: 'New Document' })
  }

  const { mutate: deleteDocumentMutate } = useMutation({
    mutationFn: (id: string) => deleteDocument(id),
    onSuccess: () => {
      queryClient.invalidateQueries({ queryKey: [QueryKeys.DOCUMENTS] })
      queryClient.invalidateQueries({ queryKey: [QueryKeys.DOCUMENTS_SEARCH]
})
      navigate({ to: Paths.DOCUMENTS })
    }
  })

  const onDeleteDocument = () => {
    deleteDocumentMutate(documentInfo.id)
  }

  return {
    showModal,
    closeModal,
    onDeleteDocument,
    onCreateDocument
  }
}
```

Игра в наперстки продолжается!

```
import { Icons } from 'constants/icons'
import { Permissions } from 'constants/permissions'
import { default as CustomButton } from 'components/ui/Button'
import CustomAriaButton from 'components/ui/CustomAriaButton'
import CustomSVG from 'components/ui/CustomSVG'
import { useRef, useState } from 'react'
import { useAbac } from 'react-abac'
import { Menu, MenuItem, MenuTrigger, Popover } from 'react-aria-components'
import { useTranslation } from 'react-i18next'
import { useClickAway } from 'react-use'
import { ModalType } from 'stores/modalStore'
import { DocumentInfo } from 'models/documents/api'
import { cc } from 'utils/combineClasses'
import styles from './styles.module.scss'
import { useDocumentMenu } from './useDocumentMenu'

interface Props {
  documentInfo: DocumentInfo
}

const FileMenu = ({ documentInfo }: Props) => {
  const { t } = useTranslation(['document', 'modals'])
  const { userHasPermissions } = useAbac()
  const { showModal, closeModal, showModalCloseDocument, onDeleteDocument,
onCreateDocument } =
    useDocumentMenu(documentInfo)

  const [isOpen, setIsOpen] = useState(false)
  const triggerRef = useRef(null)

  useClickAway(triggerRef, () => {
    setIsOpen(false)
  })

  return (
    <MenuTrigger>
      <CustomAriaButton
        onPress={() => setIsOpen(state => !state)}
        ref={triggerRef}
        className={cc(styles.menuTrigger, isOpen && styles.active)}
      >
        {t('document:menu.file.title')}
      </CustomAriaButton>
      <Popover triggerRef={triggerRef} isNonModal isOpen={isOpen}>
        <Menu onAction={() => setIsOpen(false)} className={styles.menu}>
          <MenuItem
            icon={Icons.common.createDocument}
            text={t('document:menu.file.createDocument')}
            onAction={onCreateDocument}
          />

          <MenuItem
            icon={Icons.common.copyDocument}
            text={t('document:menu.file.copyDocument')}
            onAction={() => showModal(ModalType.COPY_DOCUMENT, {
documentInfo })}
          />
          <MenuItem
            icon={Icons.common.documentShare}
            text={t('document:menu.file.share')}
            onAction={() => showModal(ModalType.SHARE, { title:
documentInfo.title, documentInfo })}
          />
          {userHasPermissions(Permissions.DELETE_DOCUMENT,
documentInfo.userRole) && (
          <MenuItem
            icon={Icons.common.trash}
            text={t('document:menu.file.deleteDocument')}
            onAction={showModalCloseDocument}
          />)}
        </Menu>
      </Popover>
    </MenuTrigger>
  )
}

export default FileMenu
```

```
import { QueryKeys } from 'constants/queryKeys'
import { Paths } from 'constants/routes'
import { useMutation, useQueryClient } from '@tanstack/react-query'
import { useNavigate } from '@tanstack/react-router'
import { useModalStore } from 'stores/modalStore'
import { deleteDocument, postDocument } from 'api/documents'
import { CreateDocumentRequest, DocumentInfo } from 'models/documents/api'

export const useDocumentMenu = (documentInfo: DocumentInfo) => {
  const [showModal, closeModal] = useModalStore(state => [state.showModal,
state.closeModal])
  const queryClient = useQueryClient()
  const navigate = useNavigate()

  const { mutate: createDocument } = useMutation({
    mutationFn: (data: CreateDocumentRequest) => postDocument(data),
    onSuccess: data => {
      queryClient.invalidateQueries({ queryKey: [QueryKeys.DOCUMENTS] })
      navigate({ to: Paths.DOCUMENT, params: { documentId: data.id } })
    }
  })

  const onCreateDocument = () => {
    createDocument({ title: 'New Document' })
  }

  const { mutate: deleteDocumentMutate } = useMutation({
    mutationFn: (id: string) => deleteDocument(id),
    onSuccess: () => {
      queryClient.invalidateQueries({ queryKey: [QueryKeys.DOCUMENTS] })
      queryClient.invalidateQueries({ queryKey: [QueryKeys.DOCUMENTS_SEARCH]
})
      navigate({ to: Paths.DOCUMENTS })
    }
  })
  const onDeleteDocument = () => {
    deleteDocumentMutate(documentInfo.id)
  }

  const showModalCloseDocument = () => showModal(ModalType.CUSTOM, {
                title: t('document:menu.file.deleteDocument'),
                children: (
                  <>
                    <p className={styles.modalMessage}>
                      {t('modals:moveToTrash.description', { title:
document.title })}
                    </p>
                    <div className={styles.modalButtonsWrapper}>
                      <CustomButton variant="secondary" onPress=
{closeModal}>
                        {t('common:actions.cancel')}
                      </CustomButton>
                      <CustomButton
                        onPress={() => {
                          onDeleteDocument()
                          closeModal()
                        }}
                      >
                        {t('common:actions.delete')}
                      </CustomButton>
                    </div>
                  </>
                )
              })
            }}

  return {
    showModal,
    closeModal,
    showModalCloseDocument,
    onDeleteDocument,
    onCreateDocument
  }
}
```

```
import { Icons } from 'constants/icons'
import CustomSVG from 'components/ui/CustomSVG'
import { forwardRef, type ReactElement } from 'react'
import { MenuItem as MenuItemAria } from 'react-aria-components'
import styles from '../styles.module.scss'
```

```
interface Props {
  text: string
  icon?: ReactElement
  onAction?: () => void
  arrayLeft?: boolean
}

const MenuItem = forwardRef<HTMLElement, Props>(({ onAction, icon, text,
arrayLeft }, ref) => {
  return (
    <MenuItemAria onAction={onAction} className={styles.menuItem} ref={ref}>
      {icon && <CustomSVG className={styles.menuOptionIcon}>{icon}
</CustomSVG>}
      <p>{text}</p>
      {arrayLeft && (
        <CustomSVG className={styles.chevron}>
{Icons.arrows.chevronArrowRight}</CustomSVG>
      )}
    </MenuItemAria>
  )
})

export default MenuItem
```

Название useDocumentMenu подходит этому хуку? Что будет если в useDocumentMenu появится стейт? а если несколько?
А действительно стало проще? Попробуем определить за что что отвечает? Это декларативный код? Что из этого будет легко переиспользовать?
Что еще может быть вынесено?

Моя версия

```
import CustomAriaButton from 'components/ui/CustomAriaButton'
import { useRef, useState, type ReactNode, type FC } from 'react'
import { Menu, MenuTrigger, Popover } from 'react-aria-components'
import { useTranslation } from 'react-i18next'
import { useClickAway } from 'react-use'
import { cc } from 'utils/combineClasses'
import styles from '../styles.module.scss'

interface Props {
  items: ReactNode
}
```

```
const FileMenu: FC<Props> = ({ items }) => {
  const { t } = useTranslation(['document', 'modals'])

  const [isOpen, setIsOpen] = useState(false)
  const triggerRef = useRef(null)

  useClickAway(triggerRef, () => {
    setIsOpen(false)
  })

  return (
    <MenuTrigger>
      <CustomAriaButton
        onPress={() => setIsOpen(state => !state)}
        ref={triggerRef}
        className={cc(styles.menuTrigger, isOpen && styles.active)}
      >
        {t('document:menu.file.title')}
      </CustomAriaButton>
      <Popover triggerRef={triggerRef} isNonModal isOpen={isOpen}>
        <Menu onAction={() => setIsOpen(false)} className={styles.menu}>
          {items}
        </Menu>
      </Popover>
    </MenuTrigger>
  )
}

export default FileMenu
```

```
import { DocumentInfo } from 'models/documents/api'
import { cc } from 'utils/combineClasses'
import EditMenu from './EditMenu'
import { default as FileMenu, Copy, Create, Delete, Share, Download } from
'./FileMenu'
import InsertMenu from './InsertMenu'
import styles from './styles.module.scss'
interface Props {
  documentInfo: DocumentInfo
  className?: string
}
const DocumentMenu = ({ documentInfo, className }: Props) => {
  return (
    <div className={cc(styles.documentMenuRoot, className)}>
```

```
      <FileMenu
        items={[
          <Create key="create" />,
          <Copy key="copy" documentInfo={documentInfo} />,
          <Share key="share" documentInfo={documentInfo} />,
          <Download key="download" />,
          <Delete key="delete" documentInfo={documentInfo} />
        ]}
      />
      <EditMenu />
      <InsertMenu documentInfo={documentInfo} />
    </div>
  )
}
```

```
import { Icons } from 'constants/icons'
import type { FC } from 'react'
import { useTranslation } from 'react-i18next'
import { ModalType, useModalStore } from 'stores/modalStore'
import { DocumentInfo } from 'models/documents/api'
import MenuItem from './MenuItem'

interface Props {
  documentInfo: DocumentInfo
}

const Copy: FC<Props> = ({ documentInfo }) => {
  const { t } = useTranslation(['document', 'modals'])
  const [showModal] = useModalStore(state => [state.showModal])

  return (
    <MenuItem
      onAction={() => showModal(ModalType.COPY_DOCUMENT, { documentInfo })}
      icon={Icons.document.copyDocument}
      text={t('document:menu.file.copyDocument')}
    />
  )
}

export default Copy
```

```
import { Icons } from 'constants/icons'
import { QueryKeys } from 'constants/queryKeys'
import { Paths } from 'constants/routes'
```

```tsx
import { useMutation, useQueryClient } from '@tanstack/react-query'
import { useNavigate } from '@tanstack/react-router'
import type { FC } from 'react'
import { useTranslation } from 'react-i18next'
import { postDocument } from 'api/documents'
import MenuItem from './MenuItem'

const Create: FC = () => {
  const { t } = useTranslation(['document', 'modals'])

  const queryClient = useQueryClient()
  const navigate = useNavigate()

  const { mutate } = useMutation({
    mutationFn: () => postDocument({ title: 'New Document' }),
    onSuccess: data => {
      queryClient.invalidateQueries({ queryKey: [QueryKeys.DOCUMENTS] })
      navigate({ to: Paths.DOCUMENT, params: { documentId: data.id } })
    }
  })

  return (
    <MenuItem
      onAction={mutate}
      icon={Icons.document.createDocument}
      text={t('document:menu.file.createDocument')}
    />
  )
}

export default Create
```

```tsx
import { Icons } from 'constants/icons'
import { Permissions } from 'constants/permissions'
import { QueryKeys } from 'constants/queryKeys'
import { Paths } from 'constants/routes'
import { useMutation, useQueryClient } from '@tanstack/react-query'
import { useNavigate } from '@tanstack/react-router'
import { default as CustomButton } from 'components/ui/Button'
import type { FC } from 'react'
import { useAbac } from 'react-abac'
import { useTranslation } from 'react-i18next'
import { ModalType, useModalStore } from 'stores/modalStore'
import { deleteDocument } from 'api/documents'
import { DocumentInfo } from 'models/documents/api'
import styles from '../styles.module.scss'
import MenuItem from './MenuItem'

interface Props {
  documentInfo: DocumentInfo
}

const Delete: FC<Props> = ({ documentInfo }) => {
  const navigate = useNavigate()
  const { t } = useTranslation(['document', 'modals'])
  const { userHasPermissions } = useAbac()
  const queryClient = useQueryClient()
  const { mutate } = useMutation({
    mutationFn: (id: string) => deleteDocument(id),
    onSuccess: () => {
      queryClient.invalidateQueries({ queryKey: [QueryKeys.DOCUMENTS] })
      queryClient.invalidateQueries({ queryKey: [QueryKeys.DOCUMENTS_SEARCH]
})
      navigate({ to: Paths.DOCUMENTS })
    }
  })

  const [showModal, closeModal] = useModalStore(state => [state.showModal,
state.closeModal])

  const onDeleteDocument = () => {
    showModal(ModalType.CUSTOM, {
      title: t('document:menu.file.deleteDocument'),
      children: (
        <>
          <p className={styles.modalMessage}>
            {documentInfo?.trashedAt
              ? t('modals:deleteFile.description', { title: document.title
})
              : t('modals:moveToTrash.description', { title: document.title
})}
          </p>
          <div className={styles.modalButtonsWrapper}>
            <CustomButton variant="secondary" onPress={closeModal}>
              {t('common:actions.cancel')}
            </CustomButton>
            <CustomButton
              onPress={() => {
                mutate(documentInfo.id)
                closeModal()
              }}
            >
              {t('common:actions.delete')}
            </CustomButton>
          </div>
        </>
      )
    })
  }
  if (!userHasPermissions(Permissions.DELETE_DOCUMENT,
documentInfo.userRole)) return null

  return (
    <MenuItem
      onAction={onDeleteDocument}
      icon={Icons.common.trash}
      text={t('document:menu.file.deleteDocument')}
    />
  )
}

export default Delete
```

Выводы

+ Декларативность на уровне FileMenu
+ Редактируя пункты меню не нужно думать о том как работает компонент FileMenu
+ Легко добавить новый компонент
+ Меньше ререндоров

- Написано больше кода

АУФ спорьте с волком

- Никогда не стоит выносить всю бизнес логику компонента в один хук если не получается четко определить конкретную ответственность хука(читай придумать ему не абстрактное название по которому можно понять что он возвращает)!
- Стоит по возможности изолировать бизнес логику на нижнем уровне иерархии компонентов

Композиция компонентов

Композиция компонентов в React — это принцип построения интерфейса, при котором сложные компоненты создаются путем объединения (вкладывания) более простых,

независимых и переиспользуемых компонентов, как из “кирпичиков”
Когда использовать

- Для изоляции логики
- Когда пропсов слишком много