Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
Факультет инфокоммуникационных технологий

ОТЧЕТ
О ЛАБОРАТОРНОЙ РАБОТЕ № 3

Специальность:
09.03.03 Мобильные и сетевые технологии
Интеллектуальные мобильные приложения

Выполнил:
Ковалев В.

Группа:
K33401

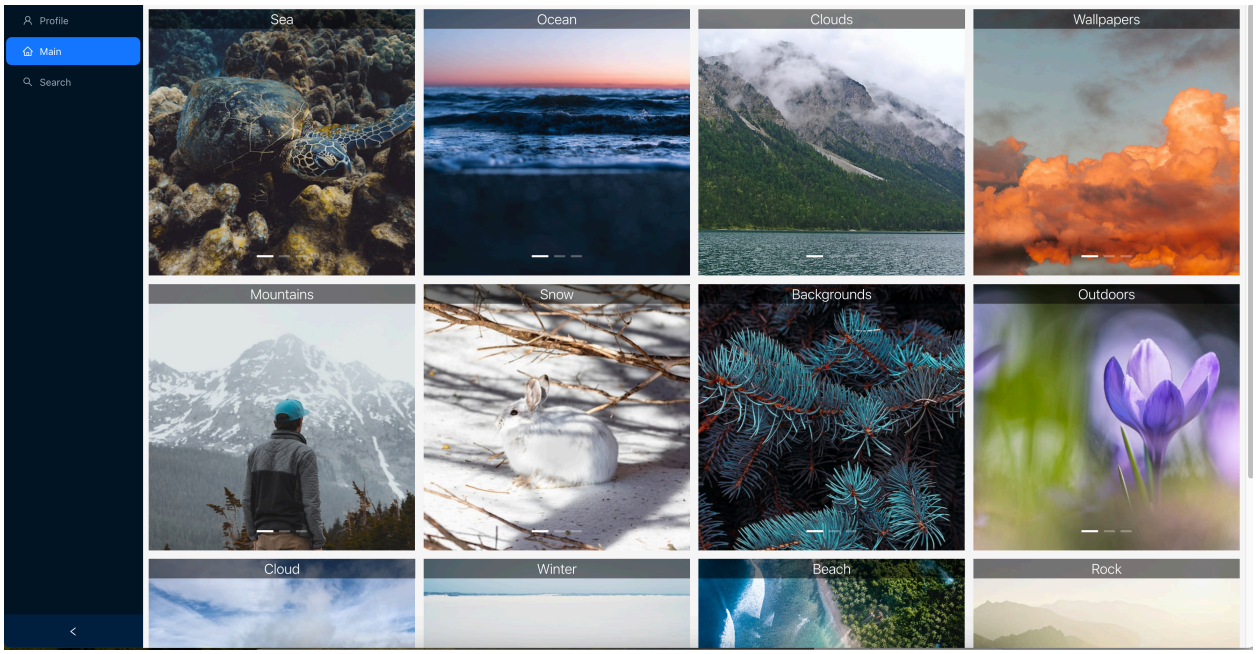
Проверил:
Добряков Д. И.

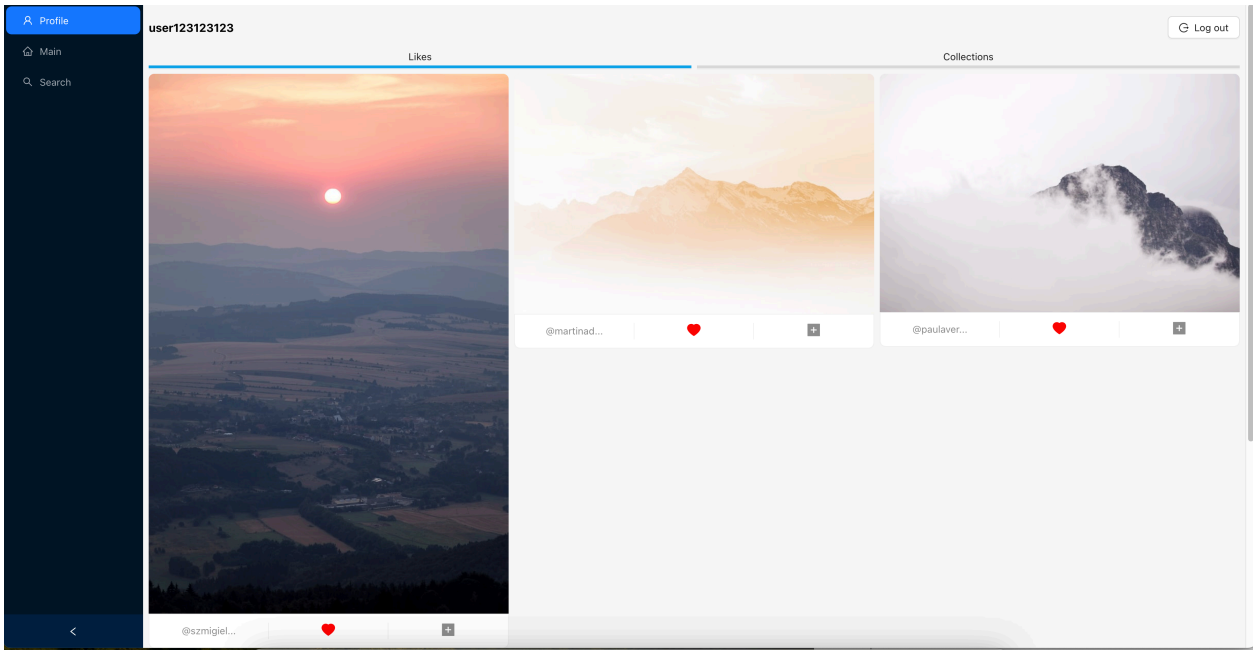
Санкт-Петербург 2022

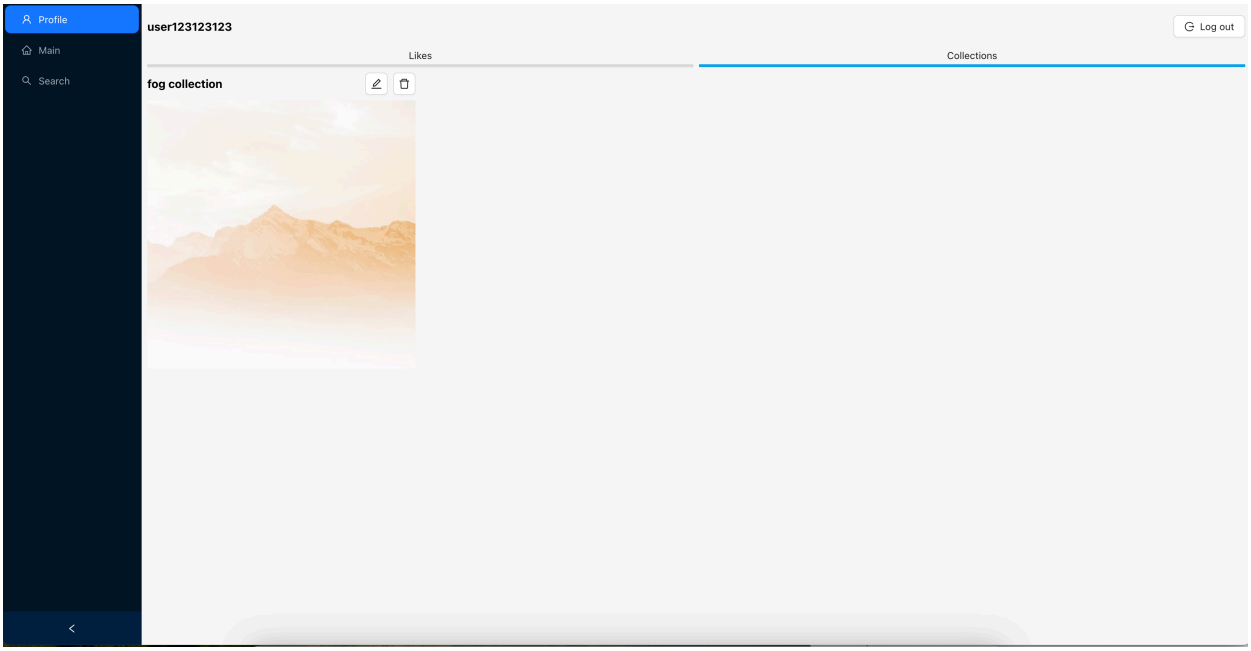
ЦЕЛЬ РАБОТЫ

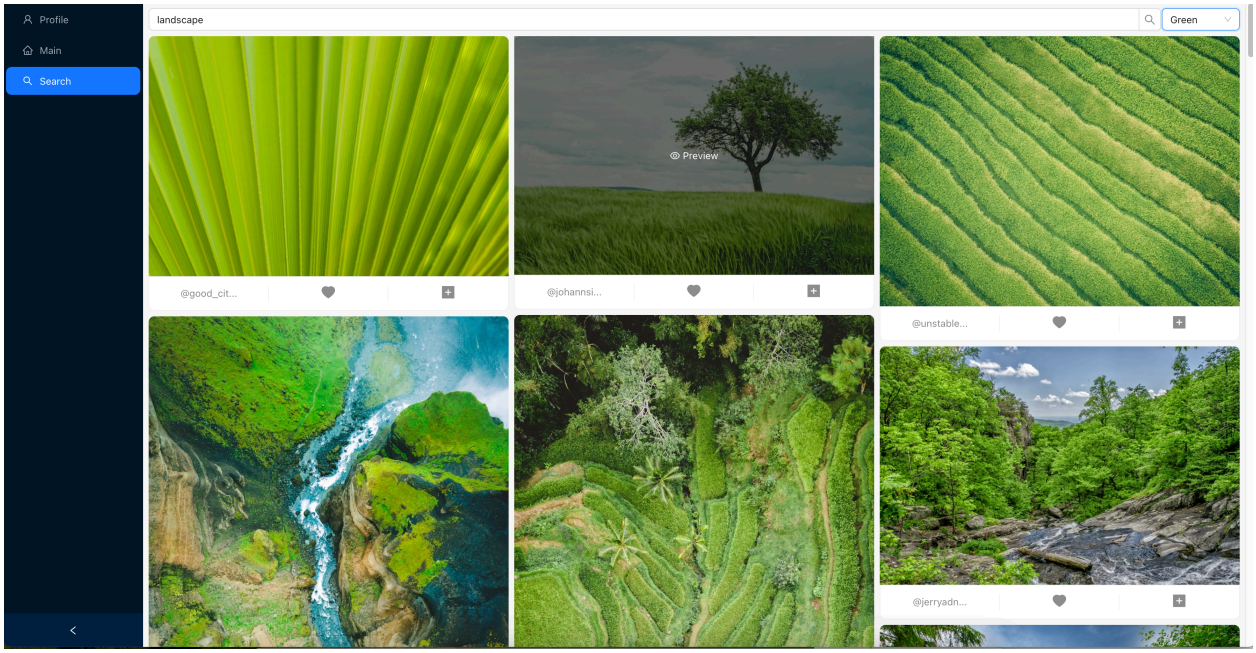
Мигрировать ранее написанный сайт на фреймворк React.

ДЕМОНСТРАЦИЯ






Profile
Main
Search
user123123123
Log out
Likes
Collections
fog collection


Profile
Main
Search
landscape
Green
Preview
@good_cit...
@johannsi...
@unstable...
@jerryadn...

ХОД РАБОТЫ

Взаимодействие с API:

```
export const getToken = async ({ username, password }) => {
  const json = JSON.stringify( value: { username, password });
  const response = await API.post( url: "auth/token/login/", json);
  return response.data;
};

4 usages  Kovalev Valery
export const getUser = async ({ token }) => {
  const response = await API.get( url: "auth/users/me/",  config: {
    headers: {
      Authorization: `Token ${token}`,
    },
  });
  return await response.data;
};

2 usages  Valery Kovalev
export const logOut = async ({token}) => {
    const response = await API.post( url: "auth/token/logout/",  data: {},  config: {
        headers: {
            Authorization: `Token ${token}`,
        },
    });
    return response.data;
}

2 usages  Valery Kovalev
export const register = async ({username,password}) => {
    const json = JSON.stringify( value: {username, password})
    const response = await API.post( url: "auth/users/", json)
    return response.data
}
```

2 usages Kovalev Valery

```
export const getCategories = async ({ limit : number  = 5, offset : number  = 0 } = {}) => {
    const response = await API.get( url: "keywords/",  config: {
        params: {
            limit,
            offset,
        },
    });
    return response.data;
};
```

4 usages Kovalev Valery

```
export const getPhotosByKeywords = async ({ keywords, limit : number  = 3, offset : number  = 0, tone :
    const response = await API.get( url: "search/",  config: {
        params: {
            limit,
            offset,
            "keywords[]": keywords,
            random: 5,
            tone,
        },
    });
    return response.data;
};
```

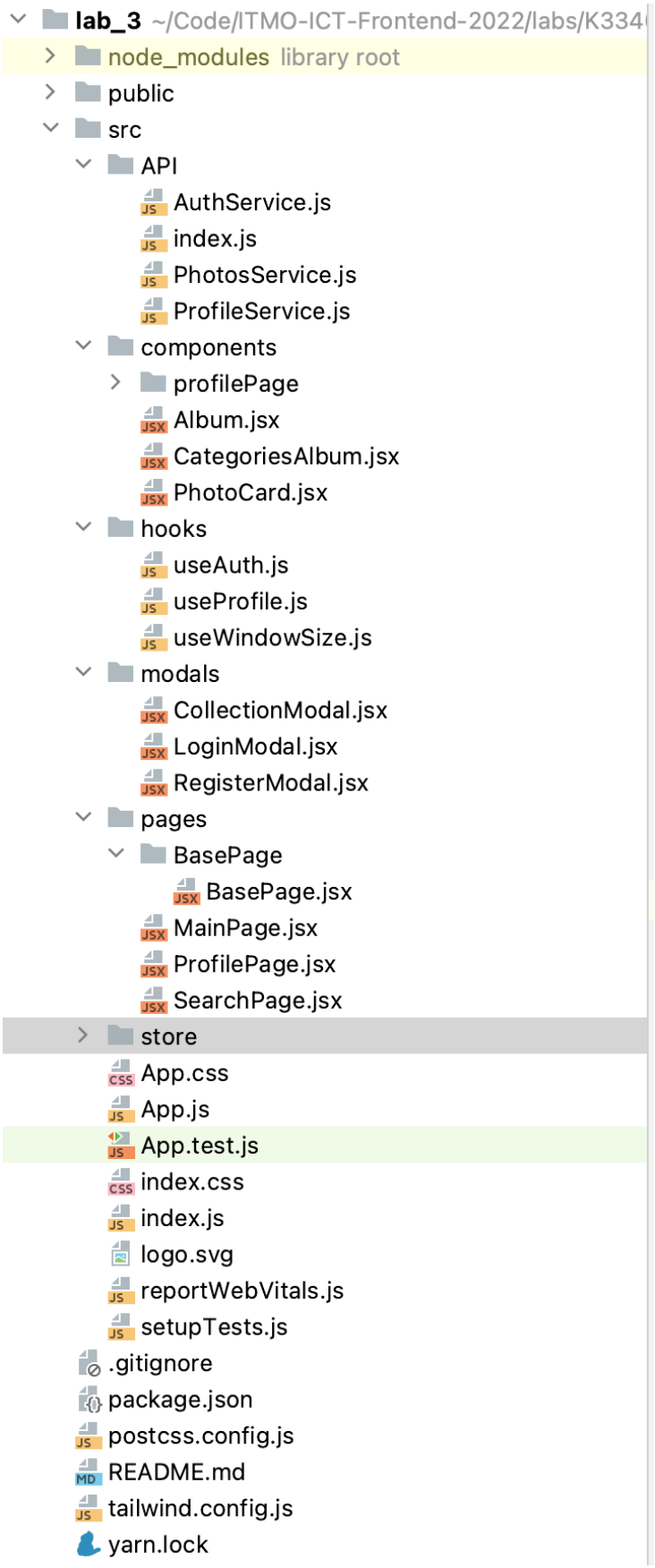
```
export const likePhoto = async (token, photo_id) => {
    await API.post(
        url: "user/likes/create",
        data: {
            photo: photo_id,
        },
        config: {
            headers: {
                Authorization: `Token ${token}`,
            },
        }
    );
};

4 usages  Kovalev Valery
export const getLikes = async (token) => {
    const response = await API.get( url: "user/likes/", config: {
        headers: {
            Authorization: `Token ${token}`,
        },
    });
    return response.data;
};

2 usages  Kovalev Valery
export const unlikePhoto = async (token, photo_id) => {
    await API.delete( url: `user/likes/${photo_id}`, config: {
        headers: {
            Authorization: `Token ${token}`,
        },
    });
};


5+ usages  Valery Kovalev
export const getCollections = async (token) => {
    const response = await API.get( url: 'user/collections/', config: {
        headers: {
            Authorization: `Token ${token}`,
        }
    })
    return response.data
}
```

```
export const getCollections = async (token) => {
    const response = await API.get( url: 'user/collections/', config: {
        headers: {
            Authorization: `Token ${token}`,
        }
    })
    return response.data
}

2 usages  Valery Kovalev
export const createCollection = async (token, title, photo_ids) => {
    await API.post( url: 'user/collections/create', data: {title, photos:photo_ids}, config: {
        headers: {
            Authorization: `Token ${token}`,
        }
    })
}

2 usages  Valery Kovalev
export const deleteCollection = async (token, collection_id) => {
    await API.delete( url: `user/collections/${collection_id}`, config: {
        headers: {
            Authorization: `Token ${token}`,
        }
    })
}

2 usages  Valery Kovalev
export const updateCollection = async (token, collection_id, title, photo_ids) => {
    await API.put( url: `user/collections/${collection_id}`, data: {title, photos:photo_ids}, config:
        headers: {
            Authorization: `Token ${token}`,
        }
    })
}
```

Редьюсеры:

```
const rootReducer = combineReducers({
    auth: authReducer,
    categories: categoriesReducer,
    search: searchReducer,
    profile: profileReducer,
    collectionModal: collectionModalReducer
});

2 usages  Kovalev Valery
export const setupStore = () => {
    return configureStore( options: {
        reducer: rootReducer,
    });
};
```

Структура файлов:



Хуки:

```
export const useAuth = () => {
  const dispatch = useDispatch();
  const { token, user } = useSelector( selector: (state) => state.auth);
  const isAuth = useMemo( factory: () => {
    return !!user?.username;
  }, deps: [user]);

  useEffect( effect: () => {
    if (!isAuth || !token) {
      dispatch(fetchUser());
    }
  }, deps: [isAuth, dispatch]);

  return { user, isAuth };
};
```

```
2 usages   Valery Kovalev
export const useProfile = () => {
    const dispatch = useDispatch();
    const {user} = useSelector( selector: (state) => state.auth);
    useEffect( effect: () => {
            if (user.username) {
                dispatch(fetchCollections())
                dispatch(fetchLikes())
            } else {
                dispatch(clearProfile())
            }
        },
        deps: [dispatch, user])
}
```

Базовая страница:

```
function getItem(label, key, icon, children) {
    return {
        key,
        icon,
        children,
        label,
    };
}

const NavigateItems = [getItem( label: "Main",  key: "main", <HomeOutlined />), getItem( label: "Search",  key: "search", <SearchOutlined />)];

const LoggedNavigateItems = [
    getItem( label: "Profile",  key: "profile", <UserOutlined />),
    getItem( label: "Main",  key: "main", <HomeOutlined />),
    getItem( label: "Search",  key: "search", <SearchOutlined />),
];

const AuthItems = [getItem( label: "Log In",  key: "login", <LoginOutlined />), getItem( label: "Register",  key: "register", <FormOutlined />)];

5+ usages   Kovalev Valery +1
const BasePage = ({ pageName, children, ref }) => {
    const { Sider, Content } = Layout;
    const [collapsed, setCollapsed] = useState( initialState: false);
    const [isLoginModalOpen, setIsLoginModalOpen] = useState( initialState: false);
    const [isRegisterModalOpen, setIsRegisterModalOpen] = useState( initialState: false);
    const { searchWord, tone } = useSelector( selector: (state) => state.search);
    const navigate = useNavigate();

    const searchLink = useMemo( factory: () => {
        if (searchWord && tone) {
            return `/search/${searchWord}/${tone}`;
        } else if (searchWord && !tone) {
            return `/search/${searchWord}`;
        } else {
            return "/search";
        }
    }, deps: [searchWord, tone]);

    3 usages   Kovalev Valery
    const menuOnClick = (e) => {
```

```
    3 usages   Kovalev Valery
    const menuOnClick = (e) => {
        switch (e.key) {
            case "login":
                setIsLoginModalOpen( value: true);
                break;
            case "register":
                setIsRegisterModalOpen( value: true);
                break;
            case "main":
                navigate("/");
                break;
            case "search":
                navigate(searchLink);
                break;
            case "profile":
                navigate("/profile");
                break;
            default:
                break;
        }
    };

    const { isAuth } = useAuth();
    useProfile()


    return (
        <Layout className="min-h-screen" ref={ref}>
            <LoginModal isLoginModalOpen={isLoginModalOpen} setIsLoginModalOpen={setIsLoginModalOpen} />
            <RegisterModal isRegisterModalOpen={isRegisterModalOpen} setIsRegisterModalOpen={setIsRegisterModalOpen} />
            <CollectionModal/>

            <Sider
                collapsible
                style={{ position: "sticky", overflow: "auto", height: "100vh", left: 0, top: 0 }}
                collapsed={collapsed}
                onCollapse={(value : boolean ) => setCollapsed(value)}
            >
                {isAuth ? (
                    <>
                        <Menu
```

```
                        theme="dark"
                        onClick={menuOnClick}
                        selectedKeys={[pageName]}
                        mode="inline"
                        items={LoggedNavigateItems}
                    />
                </>
            ) : (
                <>
                    <Menu selectable={false} onClick={menuOnClick} theme="dark" mode="inline" items={AuthItems} />
                    <Menu
                        theme="dark"
                        onClick={menuOnClick}
                        selectedKeys={[pageName]}
                        mode="inline"
                        items={NavigateItems}
                    />
                </>
            )}
        </Sider>
        <Layout className="site-layout p-2">
            <Content>{children}</Content>
        </Layout>
    </Layout>
  );
};

5+ usages  Kovalev Valery
export default BasePage;
```

Пример страницы:

```
function MainPage() {
    const dispatch = useDispatch();
    const { categories, isLoading, offset, count, page, limit } = useSelector( selector: (state) => state.categories);
    const [currentPage, setCurrentPage] = useState(page);

    useEffect( effect: () => {
        dispatch(fetchCategories());
    }, deps: [dispatch, offset]);

    1 usage  Kovalev Valery
    const onPaginate = (page) => {
        dispatch(changePage(page));
        setCurrentPage(page);
    };

    1 usage  Kovalev Valery
    const onShowSizeChange = (page, size) => {
        dispatch(changeLimit( state: { page, limit: size }));
    };

    return (
        <BasePage pageName="main">
            <CategoriesAlbum categories={categories} isLoading={isLoading} countOnPage={limit} />
            <div className="flex justify-center my-2">
                <Pagination
                    onShowSizeChange={onShowSizeChange}
                    hideOnSinglePage
                    current={currentPage}
                    onChange={onPaginate}
                    defaultCurrent={1}
                    total={count}
                    defaultPageSize={limit}
                    pageSizeOptions={[12, 24, 36]}
                />
            </div>
        </BasePage>
    );
}

2 usages  Kovalev Valery
export default MainPage;
```

Вывод:

Я перенес приложение на React, используя также Tailwind, Ant Design и Redux Toolkit. Этот вариант сильно отличается от того, что я делал на JS и Bootstrap, так как я хотел опробовать новые для меня технологии.