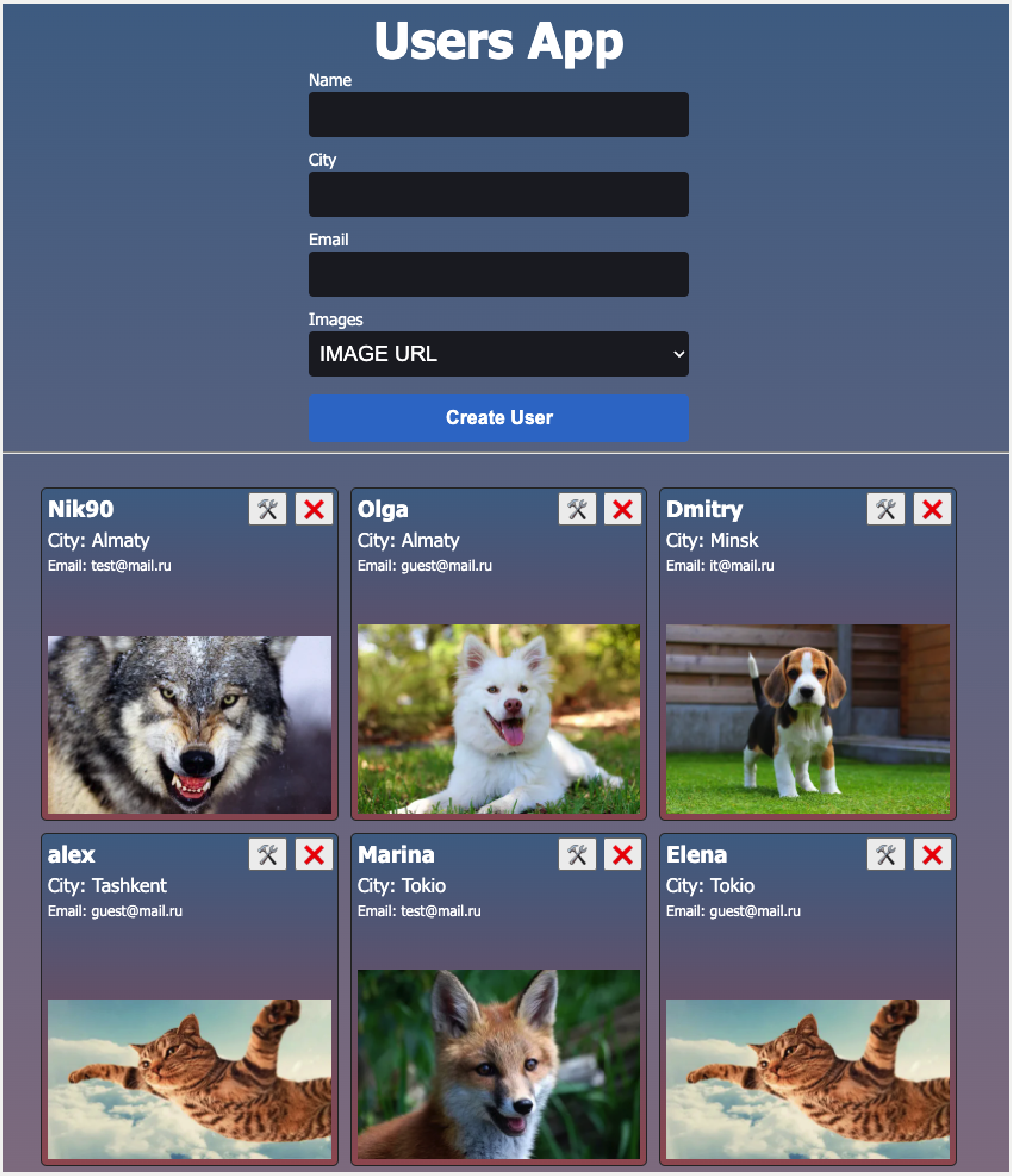
Frontender[1.0] JavaScript - Form, CRUD, mock api. USERS Management APP. Приложение

YouTube	https://youtu.be/lVDnSz9QqtU
Telegram	https://t.me/Dmitry_Kolotilshikov
Github	https://github.com/DmitryKolotilshikov/
Boosty	https://boosty.to/dmitry_ko
# Номер урока	51

USERS APP



Users App

Name

City

Email

Images

IMAGE URL

Create User

Nik90
City: Almaty
Email: test@mail.ru

Olga
City: Almaty
Email: guest@mail.ru

Dmitry
City: Minsk
Email: it@mail.ru

alex
City: Tashkent
Email: guest@mail.ru

Marina
City: Tokio
Email: test@mail.ru

Elena
City: Tokio
Email: guest@mail.ru

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script src="script.js" defer></script>
    <link rel="stylesheet" href="styles.css">
    <title>Form project</title>
```

```html
</head>

<body>
    <div class="container">
        <section class="login-section">
            <h1 class="title">Users App</h1>

            <div class="form-block">
                <form class="form" data-create-user-form>
                    <div class="control-field">
                        <label for="nameId" class="form-label">Name</label>
                        <input type="text" class="form-control" id="nameId" name="userName" required minlength="2"
                            maxlength="23">
                    </div>

                    <div class="control-field">
                        <label for="cityId" class="form-label">City</label>
                        <input type="text" class="form-control" id="cityId" name="userCity" required minlength="2"
                            maxlength="20">
                    </div>

                    <div class="control-field">
                        <label for="emailId" class="form-label">Email</label>
                        <input type="email" class="form-control form-control--email" id="cityemailIdId" name="userEmail"
                            required>
                    </div>

                    <div class="control-field">
                        <label for="imagesUrlId" class="form-label">Images</label>

                        <select name="userImageUrl" id="imagesUrlId" class="form-control form-control--images" required>
                            <option value="">Image URL</option>
                            <hr>
                            <option
                                value="https://avatars.mds.yandex.net/i?id=88cc30ba21222ee61db2d32974a5b380259ee41f-3380069-images-thumbs&n=13">
                                Cat 1</option>
                            <option
                                value="https://avatars.mds.yandex.net/i?id=6444bd82bce43803b8ad0601c12a80e7-5230955-images-thumbs&n=13">
                                Cat 2</option>
                            <option
                                value="https://avatars.mds.yandex.net/i?id=60f5028735fd33706fd8e50bb1d7f636062b21a4-8210619-images-thumbs&n=13">
                                Cat 3</option>
                            <option
                                value="https://avatars.mds.yandex.net/i?id=ae0521f7a56e37beaa15c3469ab4c338e350c501-4453150-images-thumbs&n=13">
                                Dog 1</option>
                            <option
                                value="https://avatars.mds.yandex.net/i?id=fec854b859968252cfa2ac789041838662475e7e-4667938-images-thumbs&n=13">
                                Dog 2</option>
                            <option
```

```
                                value="https://avatars.mds.yandex.net/i?id=26253ff7e734e6fd04
31b2fbc2b4a1a669ed2685be8a39d1-9148232-images-thumbs&n=13">
                                Dog 3</option>
                            <option
                                value="https://avatars.mds.yandex.net/i?id=eaed52ea5bd298c60f
f850710e5d05ddd9d26b49-8082760-images-thumbs&n=13">
                                Wolf 1</option>
                            <option
                                value="https://avatars.mds.yandex.net/i?id=730e0bcc75f17fff29
6adf3dcdaae2036067665ec12d546e-12645552-images-thumbs&n=13">
                                Fox 1</option>
                        </select>
                    </div>

                    <button type="submit" class="btn submit-btn">Create User</button>
                </form>
            </div>
        </section>
    </div>
    <hr>

    <div class="users-container" data-users-container></div>

    <dialog class="edit-form-dialog" data-edit-user-form-dialog></dialog>
</body>

</html>
```

```
* {
    box-sizing: border-box;
}

body {
    margin: 0;
    font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;
    min-height: 100vh;
    background: linear-gradient(180deg, #4d6f93 0%, #a84545a6 100%);
}

.container {
    padding: 10px 0 0 0;
    display: flex;
    justify-content: center;
    align-items: center;
}

.login-section {
    width: 100%;
    max-width: 1000px;
    padding: 0 15px;
    display: flex;
    flex-direction: column;
    justify-content: flex-start;
    align-items: center;
    flex-wrap: nowrap;
}
```

```css
.title {
    margin: 0;
    color: #fff;
    font-size: 42px;
}

.form-block {
    width: 100%;
    max-width: 320px;
}

.form {
    display: flex;
    flex-direction: column;
    justify-content: flex-start;
    align-items: stretch;
    gap: 10px;
}

.control-field {
    width: 100%;
    display: flex;
    flex-direction: column;
    justify-content: flex-start;
    align-items: stretch;
    gap: 2px;
}

.form-label {
    color: #fff;
    font-size: 14px;
}

.form-control {
    height: 38px;
    padding: 0 5px;
    font-size: 18px;
    color: #fff;
    text-transform: uppercase;
    background-color: #20222b;
    border: none;
    border-radius: 4px;
}

.form-control:focus-visible {
    outline: 1px solid #fff;
}

.form-control--email {
    text-transform: initial;
}

.btn {
    height: 40px;
    border: none;
    border-radius: 4px;
    color: #fff;
    font-weight: 700;
    text-transform: capitalize;
    font-size: 16px;
    cursor: pointer;
}

.btn:hover {
    opacity: 0.9;
}

.btn:focus-visible {
    outline: 1px solid #fff;
}

.submit-btn {
    margin: 5px 0 0 0;
    background-color: #377bcf;
}

.users-container {
    margin: 0 auto;
    padding: 20px 0 50px 0;
    width: 100%;
    max-width: 1200px;
    display: flex;
    justify-content: center;
    align-items: center;
    gap: 10px;
    flex-wrap: wrap;
}

.user-card {
    max-width: 250px;
    height: 280px;
    flex-shrink: 0;
    border: 1px solid #2f2f2f;
    border-radius: 5px;
    padding: 5px;
    display: flex;
    flex-direction: column;
    justify-content: flex-start;
    align-items: flex-start;
    background: linear-gradient(180deg, #4d6f93 0%, #a84545a6 100%);
    color: #fff;
    position: relative;

    h3 {
        margin: 0;
    }

    img {
        width: 100%;
        object-fit: cover;
        margin-top: auto;
    }

    p {
        margin: 5px 0;
    }
    span {
        font-size: 12px;
    }

    .user-edit-btn {
        position: absolute;
        top: 3px;
        right: 42px;
        cursor: pointer;
    }

    .user-remove-btn {
        position: absolute;
        top: 3px;
        right: 3px;
        cursor: pointer;
    }
}

.user-card:hover {
    box-shadow: 0 0 15px 8px #377bcf;
}

.edit-form-dialog {
    margin: 0;
    position: absolute;
    min-width: 300px;
    top: 50%;
    left: 50%;
    translate: -50% -50%;

    .form-label {
        color: #000;
    }

    .close-edit-form-btn {
        position: absolute;
        top: 3px;
        right: 3px;
        cursor: pointer;
    }
}

.edit-form-dialog::backdrop {
    background: rgba(0, 0, 0, 0.5);
}
```

```
const createUserForm = document.querySelector("[data-create-user-form]");
const editUserFormDialog = document.querySelector("[data-edit-user-form-dialog]");
const usersContainer = document.querySelector("[data-users-container]");

const MOCK_API_URL = "https://675c54bafe09df667f63812c.mockapi.io/users";

let users = [];

// ------- Клик по всему контейнеру (делегирование событий) -------
```

```
usersContainer.addEventListener("click", (e) => {
    if (e.target.hasAttribute("data-user-remove-btn")) {
        // console.log("userRemoveBtn" in e.target.dataset)
        const isRemoveUser = confirm("Вы точно хотите удалить этого красавчика?");
        isRemoveUser && removeExistingUserAsync(e.target.dataset.userId);
        return;
    }

    if (e.target.hasAttribute("data-user-edit-btn")) {
        populateDialog(e.target.dataset.userId);

        editUserFormDialog.showModal();
    }
})

// ------- Событие отравки формы создания пользователя -------
createUserForm.addEventListener("submit", (e) => {
    e.preventDefault();
    const formData = new FormData(createUserForm);
    const formUserData = Object.fromEntries(formData);

    const newUserData = {
        name: formUserData.userName,
        city: formUserData.userCity,
        email: formUserData.userEmail,
        avatar: formUserData.userImageUrl,
    }

    createNewUserAsync(newUserData);
})

// ------- Редактирование существующего пользователя -------
const editExistingUserAsync = async (newUserData) => {
    try {
        const response = await fetch(`${MOCK_API_URL}/${newUserData.id}`, {
            method: "PUT",
            body: JSON.stringify(newUserData),
            headers: {
                "Content-type": "application/json"
            }
        });

        if (response.status === 400) {
            throw new Error(`клиентская ошибка`)
        }

        const editedUser = await response.json();

        users = users.map((user) => {
            if (user.id === editedUser.id) {
                return editedUser;
            }
            return user;
        })

        editUserFormDialog.close();
        renderUsers();
        alert("ПОЛЬЗОВАТЕЛЬ УСПЕШНО ОТРЕДАКТИРОВАН")
    } catch (error) {
        console.error("ОШИБКА при редактировании пользователя пользователя: ", error.message)
    }
}

// ------- Удаление существующего пользователя -------
const removeExistingUserAsync = async (userId) => {
    try {
        const response = await fetch(`${MOCK_API_URL}/${userId}`, {
            method: "DELETE"
        });

        if (response.status === 404) {
            throw new Error(`${userId} не найден`)
        }

        const removedUser = await response.json();

        users = users.filter(user => user.id !== removedUser.id);

        renderUsers();

        alert("ПОЛЬЗОВАТЕЛЬ УСПЕШНО УДАЛЕН");
    } catch (error) {
        console.error("ОШИБКА при удалении пользователя: ", error.message)
    }
}

// ------- Создание нового пользователя -------
const createNewUserAsync = async (newUserData) => {
    try {
        const response = await fetch(MOCK_API_URL, {
            method: "POST",
            body: JSON.stringify(newUserData),
            headers: {
                "Content-type": "application/json"
            }
        });
        const newCreatedUser = await response.json();

        users.unshift(newCreatedUser);
        renderUsers();

        createUserForm.reset();

        alert("НОВЫЙ ПОЛЬЗОВАТЕЛЬ УСПЕШНО СОЗДАН")
    } catch (error) {
        console.error("ОШИБКА создания нового пользователя: ", error.message)
    }
}

// ------- Получение всех пользователей -------
const getUsersAsync = async () => {
    try {
        const response = await fetch(MOCK_API_URL);
        users = await response.json();
        renderUsers();
    } catch (error) {
        console.error("ПОЙМАННАЯ ОШИБКА: ", error.message)
    }
}

// ------- Отрисовка пользователей -------
const renderUsers = () => {
    usersContainer.innerHTML = "";

    users.forEach((user) => {
        usersContainer.insertAdjacentHTML("beforeend", `
            <div class="user-card">
                <h3>${user.name}</h3>
                <p>City: ${user.city}</p>
                <span>Email: ${user.email}</span>
                <img src="${user.avatar}"/>
                <button class="user-edit-btn" data-user-id="${user.id}" data-user-edit-btn>🛠️</button>
                <button class="user-remove-btn" data-user-id="${user.id}" data-user-remove-btn>❌</button>
            </div>
        `)
    })
}

// ------- Заполнение модального окна разметкой формы -------
const populateDialog = (userId) => {
    editUserFormDialog.innerHTML = "";

    const editForm = document.createElement("form");
    const closeFormBtn = document.createElement("button");

    closeFormBtn.classList.add("close-edit-form-btn");
    closeFormBtn.textContent = "❌";
    closeFormBtn.addEventListener("click", () => editUserFormDialog.close());

    editForm.addEventListener("submit", (e) => {
        e.preventDefault();
        const formData = new FormData(editForm);
        const formUserData = Object.fromEntries(formData);

        const newUserData = {
            id: formUserData.userId,
            name: formUserData.userName,
            city: formUserData.userCity,
            email: formUserData.userEmail,
            avatar: formUserData.userImageUrl,
        }

        editExistingUserAsync(newUserData);
    })

    editForm.classList.add("form");
    editForm.innerHTML = `
        <input type="text" name="userId" value="${userId}" hidden/>

        <div class="control-field">
```

```
            <label for="nameId" class="form-label">Name</label>
            <input type="text" class="form-control" id="nameId" name="userName" required minl
ength="2"
                maxlength="23">
        </div>

        <div class="control-field">
            <label for="cityId" class="form-label">City</label>
            <input type="text" class="form-control" id="cityId" name="userCity" required minl
ength="2"
                maxlength="20">
        </div>

        <div class="control-field">
            <label for="emailId" class="form-label">Email</label>
            <input type="email" class="form-control form-control--email" id="cityemailIdId" n
ame="userEmail"
                required>
        </div>

        <div class="control-field">
            <label for="imagesUrlId" class="form-label">Images</label>

            <select name="userImageUrl" id="imagesUrlId" class="form-control form-control--im
ages" required>
                <option value="">Image URL</option>
                <hr>
                <option
                    value="https://avatars.mds.yandex.net/i?id=88cc30ba21222ee61db2d32974a5b3
80259ee41f-3380069-images-thumbs&n=13">
                    Cat 1</option>
                <option
                    value="https://avatars.mds.yandex.net/i?id=6444bd82bce43803b8ad0601c12a80
e7-5230955-images-thumbs&n=13">
                    Cat 2</option>
                <option
                    value="https://avatars.mds.yandex.net/i?id=60f5028735fd33706fd8e50bb1d7f6
36062b21a4-8210619-images-thumbs&n=13">
                    Cat 3</option>
                <option
                    value="https://avatars.mds.yandex.net/i?id=ae0521f7a56e37beaa15c3469ab4c3
38e350c501-4453150-images-thumbs&n=13">
                    Dog 1</option>
                <option
                    value="https://avatars.mds.yandex.net/i?id=fec854b859968252cfa2ac78904183
8662475e7e-4667938-images-thumbs&n=13">
                    Dog 2</option>
                <option
                    value="https://avatars.mds.yandex.net/i?id=26253ff7e734e6fd0431b2fbc2b4a1
a669ed2685be8a39d1-9148232-images-thumbs&n=13">
                    Dog 3</option>
                <option
                    value="https://avatars.mds.yandex.net/i?id=eaed52ea5bd298c60ff850710e5d05
ddd9d26b49-8082760-images-thumbs&n=13">
                    Wolf 1</option>
                <option
                    value="https://avatars.mds.yandex.net/i?id=730e0bcc75f17fff296adf3dcdaae2
036067665ec12d546e-12645552-images-thumbs&n=13">
```

```
                    Fox 1</option>
            </select>
        </div>

        <button type="submit" class="btn submit-btn">Edit User</button>
    `

    editUserFormDialog.append(editForm, closeFormBtn);
}

getUsersAsync();
```