📘 Project Report – GitHub Organizer

💼 Project Title

GitHub Organizer – A Repository Cleanup Dashboard

📌 Objective

Build a dashboard that authenticates with GitHub using OAuth, fetches user repositories securely, and allows basic repo selection and organization features.

What I Learned

1. GitHub OAuth Authentication (Backend)

Code: `server/controllers/githubController.js`

```
const axios = require("axios");
const jwt = require("jsonwebtoken");
const User = require("../models/User");

const handleGitHubCallback = async (req, res) => {
  const code = req.query.code;
  const tokenResponse = await axios.post(
    `https://github.com/login/oauth/access_token`,
    {
      client_id: process.env.GITHUB_CLIENT_ID,
      client_secret: process.env.GITHUB_CLIENT_SECRET,
      code,
      redirect_uri: process.env.GITHUB_REDIRECT_URI,
    },
    { headers: { Accept: "application/json" } }
  );

  const accessToken = tokenResponse.data.access_token;
  const userResponse = await axios.get(`https://api.github.com/user`, {
    headers: { Authorization: `Bearer ${accessToken}` },
  });

  const { id, login, avatar_url } = userResponse.data;
```

```
  let user = await User.findOne({ githubId: id });
  if (!user) {
    user = new User({ githubId: id, username: login, avatar: avatar_url,
accessToken });
  } else {
    user.accessToken = accessToken;
  }

  await user.save();

  const jwtToken = jwt.sign({ userId: user._id }, process.env.JWT_SECRET, {
expiresIn: '7d' });
  res.redirect(`${process.env.FRONTEND_URL}/welcome?token=${jwtToken}`);
};
```

---

2. Decode JWT and Save User (Frontend)

Code: `Welcome.jsx`

```
import { useEffect } from 'react';
import { useNavigate } from 'react-router-dom';
import jwt_decode from 'jwt-decode';

const Welcome = () => {
  const navigate = useNavigate();

  useEffect(() => {
    const token = new URLSearchParams(window.location.search).get("token");

    if (token) {
      localStorage.setItem("token", token);
      const user = jwt_decode(token);
      localStorage.setItem("user", JSON.stringify(user));
      navigate("/dashboard");
    } else {
      navigate("/");
    }
  }, []);

  return <div>Redirecting...</div>;
};

export default Welcome;
```

---

3. Fetch GitHub Repos Securely

Code: `Dashboard.jsx`

```
const fetchRepositories = async () => {
  const token = localStorage.getItem("token");
  if (!token) return;

  try {
    const res = await fetch(`http://localhost:5000/api/github/repos`, {
      headers: { Authorization: `Bearer ${token}` },
    });
    const data = await res.json();
    setRepos(data.repos);
    setShowDropdown(true);
  } catch (err) {
    console.error("Repo fetch error", err);
  }
};
```

---

4. Logout Implementation

```
const handleLogout = () => {
  localStorage.removeItem("token");
  localStorage.removeItem("user");
  navigate("/");
};
```

---

5. Show Dropdown + Repo Selection

```
{showDropdown && (
  <div className="absolute mt-2 w-full bg-white border rounded shadow max-h-60
overflow-y-auto">
    {repos.map((repo) => (
      <div
        key={repo.id}
        onClick={() => {
          setSelectedRepo(repo);
          setShowDropdown(false);
        }}
        className="p-3 hover:bg-gray-100 cursor-pointer"
```

```
      >
        <div className="font-medium">{repo.name}</div>
        <div className="text-xs text-gray-500">{repo.description}</div>
      </div>
    ))}
  </div>
)}
```

💡 Key Takeaways

- Learned OAuth 2.0 flow with GitHub.
- Used `jwt-decode` to parse tokens in the frontend.
- Built a secure API gateway between frontend and GitHub.
- Improved skills in Tailwind CSS and conditional rendering in React.
- Learned how to maintain clean code, state management, and user sessions.

📅 Next Steps / Future Features

- Repo cleanup: Remove unused dependencies, cleanup scripts.
- Show repo analytics: star count, last updated, open issues.
- Delete/archive repos.
- Add GitHub Actions integration for automation.
- Deploy on Vercel + Render or Railway.

Would you like this exported to PDF or Markdown format?