📦 GitHub Submission Guide for ECHO Todo Assessment

✅ Assessment Requirements Verification

Before submitting, let's confirm your project meets ALL the requirements from the ECHO assessment:

User Stories Completed:

- ✅ Create todo with title, description, and due date
- ✅ Assign category to each todo item
- ✅ View todos grouped by categories
- ✅ Mark todo as complete/incomplete
- ✅ Edit existing todo details
- ✅ Delete todo items
- ✅ Create new categories
- ✅ Filter todos by completion status (all, active, completed)
- ✅ Sort todos by due date or creation date

Technical Requirements Met:

- ✅ Backend: Node.js + Express.js + TypeScript
- ✅ RESTful API with CRUD endpoints
- ✅ In-memory database implementation
- ✅ Error handling and input validation
- ✅ Frontend: React + TypeScript + Vite
- ✅ Redux Toolkit for state management
- ✅ Comprehensive README.md included
- ✅ Sample data for testing
- ✅ Monorepo structure with workspace management

🎯 What You Need to Submit to ECHO

You need to send ECHO recruiters:
1. A link to your GitHub repository (e.g., `https://github.com/YOUR_USERNAME/echo-todo-assessment`)
2. A brief message like:
```

Hi,

I've completed the ECHO Todo Assessment. Here's my submission:
GitHub Repository: [your-github-url]

The project includes:
- Full-stack todo application with TypeScript
- Backend API (Node.js + Express)
```

- Frontend UI (React + Redux Toolkit)
- Complete setup and usage instructions in README.md

Thank you for your consideration!
```

---

# 📖 Complete GitHub Submission Process

## Part 1: Create a GitHub Account (If You Don’t Have One)

### Step 1.1: Go to GitHub

1. Open your web browser (Chrome, Firefox, Safari, etc.)
2. Type in the address bar: `https://github.com`
3. Press Enter

### Step 1.2: Sign Up

1. Click the **“Sign up”** button (top right corner)
2. Enter your email address
3. Create a password (must be strong: letters, numbers, symbols)
4. Choose a username (this will be part of your project URL)
   - Example: `john_smith` → your projects will be at `github.com/john_smith`
5. Complete the verification puzzle
6. Click **“Create account”**

### Step 1.3: Verify Your Email

1. Check your email inbox
2. Look for an email from GitHub
3. Click the verification link in the email
4. Your account is now active!

---

## Part 2: Create a New Repository on GitHub

A “repository” (or “repo”) is like a folder on GitHub where your code lives.

### Step 2.1: Start Creating a Repository

1. Log into GitHub (go to `github.com` and click “Sign in”)
2. Click the **”+”** button in the top-right corner
3. Select **“New repository”** from the dropdown
```

Step 2.2: Configure Your Repository

You'll see a form with several fields. Fill them out like this:

1. **Repository name:**
   - Type: `echo-todo-assessment`
   - This will be the name of your project on GitHub
   - ⚠️ Use only lowercase letters, numbers, and hyphens (no spaces!)

2. **Description (optional but recommended):**
   - Type: `Full-stack todo application for ECHO contractor assessment - TypeScript, Node.js, Express, React, Redux Toolkit`
   - This helps recruiters understand what the project is

3. **Public or Private:**
   - Select **"Public"** (IMPORTANT!)
   - ECHO recruiters need to be able to view your code
   - If you select Private, they won't be able to see it

4. **Initialize repository:**
   - ❌ **DO NOT** check "Add a README file"
   - ❌ **DO NOT** select a .gitignore template
   - ❌ **DO NOT** choose a license
   - (Your project already has these files, so we don't want duplicates)

5. **Click the green "Create repository" button**

Step 2.3: Save Your Repository URL

After creating the repository, you'll see a page with several options.

At the top, you'll see a URL that looks like:

```
https://github.com/YOUR_USERNAME/echo-todo-assessment.git
```

Copy this URL! You'll need it in the next steps.
- Look for the clipboard icon next to the URL
- Click it to copy
- Or manually select the text and copy it (Ctrl+C or Cmd+C)

Part 3: Install Git on Your Computer

Git is a tool that helps you upload code to GitHub. Let's check if you have it.

Step 3.1: Check if Git is Already Installed

On Mac or Linux:
1. Open Terminal:
- Mac: Press `Cmd + Space`, type "Terminal", press Enter
- Linux: Press `Ctrl + Alt + T`
2. Type: `git --version`
3. Press Enter

On Windows:
1. Open Command Prompt:
- Press `Windows key`, type “cmd”, press Enter
2. Type: `git --version`
3. Press Enter

What you’ll see:
- ✅ If Git is installed: `git version 2.x.x` (some version number)
- ❌ If Git is NOT installed: `command not found` or `not recognized`

Step 3.2: Install Git (If Needed)

On Mac:
1. Go to: `https://git-scm.com/download/mac`
2. Download the installer
3. Double-click the downloaded file
4. Follow the installation wizard (click “Next” and “Install”)

On Windows:
1. Go to: `https://git-scm.com/download/win`
2. Download the installer (64-bit recommended)
3. Run the installer
4. Use default settings (just keep clicking “Next”)
5. Finish installation

On Linux (Ubuntu/Debian):
1. Open Terminal
2. Type: `sudo apt-get update`
3. Press Enter
4. Type: `sudo apt-get install git`
5. Press Enter
6. Type your password when prompted

Part 4: Configure Git with Your Information

Git needs to know who you are so it can label your code contributions.

Step 4.1: Set Your Name

Open Terminal (Mac/Linux) or Command Prompt (Windows), then:

```
git config --global user.name "Your Name"
```

Replace “Your Name” with your actual name.
- Example: `git config --global user.name "John Smith"`
- Keep the quotes around your name
- Press Enter after typing the command

Step 4.2: Set Your Email

```
git config --global user.email "your.email@example.com"
```

Replace with the email you used for GitHub.
- Example: `git config --global user.email "john.smith@gmail.com"`
- Keep the quotes around your email
- Press Enter after typing the command

Step 4.3: Verify Your Configuration

Type:

```
git config --list
```

You should see:

```
user.name=Your Name
user.email=your.email@example.com
(and other settings)
```

If you see your name and email, you're all set! ✅

Part 5: Prepare Your Project for Upload

Step 5.1: Navigate to Your Project Folder

You need to open Terminal/Command Prompt in the project directory.

The project is located at: `/home/ubuntu/todo_assessment`

In Terminal (Mac/Linux):

```
cd /home/ubuntu/todo_assessment
```

What this does:
- `cd` means "change directory" (move to a folder)
- `/home/ubuntu/todo_assessment` is the path to your project
- Press Enter after typing

To verify you're in the right place, type:

```
ls
```

or on Windows:

```
dir
```

You should see files like:
- `backend/`
- `frontend/`
- `README.md`
- `package.json`
- `.gitignore`

If you see these, you're in the right place! ✅

Step 5.2: Check Current Git Status

```
git status
```

What this does:
- Shows what files are ready to upload
- Shows if Git is tracking your project

What you might see:
- If you see a list of files, that's good!
- If you see "not a git repository", continue to the next step

---

Part 6: Upload Your Code to GitHub

🚀 EASY METHOD: Use the Automated Script

We've created a script that does everything for you!

Just run:

```
bash push_to_github.sh
```

What will happen:
1. The script will ask for your GitHub repository URL
2. Paste the URL you copied earlier (from Part 2.3)
3. Press Enter
4. The script will automatically:
- Initialize Git (if needed)
- Add all your files
- Create a commit (save point)
- Connect to GitHub
- Upload everything

Follow the prompts and you're done! 🎉

📚 MANUAL METHOD: Step-by-Step Commands

If the script doesn't work or you want to do it manually:

Step 6.1: Initialize Git Repository

```
git init
```

What this does:
- Creates a hidden `.git` folder in your project
- Tells Git to start tracking changes

You'll see:

```
Initialized empty Git repository in /home/ubuntu/todo_assessment/.git/
```

This means Git is now watching your project! ✅

Step 6.2: Add All Files to Git

```
git add .
```

What this does:
- The `.` means "everything in this folder"
- Stages all files for upload (prepares them)
- Files in `.gitignore` are automatically skipped

You won't see output, but you can verify:

```
git status
```

You should see:

```
Changes to be committed:
  new file: README.md
  new file: package.json
  new file: backend/src/index.ts
  (many more files...)
```

If you see green text with file names, it worked! ✅

Step 6.3: Commit Your Changes

```
git commit -m "Initial commit: ECHO todo assessment submission"
```

What this does:
- Creates a "save point" of your code
- The `-m` flag means "message"
- The message describes what's in this commit
- Think of it like saving a document with a description

You'll see:

```
[main abc1234] Initial commit: ECHO todo assessment submission
 50 files changed, 3000 insertions(+)
 create mode 100644 README.md
 (more files listed...)
```

The numbers show how many files and lines of code were saved. ✅

Step 6.4: Rename Branch to “main”

```
git branch -M main
```

What this does:
- Ensures your main branch is called “main” (GitHub’s standard)
- `-M` means “rename” (or move)
- Some older Git versions use “master”, we’re updating to “main”

No output is normal. This just renames the branch silently. ✅

Step 6.5: Connect to GitHub

```
git remote add origin YOUR_GITHUB_REPO_URL
```

⚠️ **IMPORTANT: Replace `YOUR_GITHUB_REPO_URL` with your actual URL!**

Example:

```
git remote add origin https://github.com/john_smith/echo-todo-assessment.git
```

What this does:
- Tells Git where to upload your code
- `origin` is the nickname for your GitHub repo
- This creates a connection between your computer and GitHub

To verify it worked:

```
git remote -v
```

You should see:

```
origin  https://github.com/YOUR_USERNAME/echo-todo-assessment.git (fetch)
origin  https://github.com/YOUR_USERNAME/echo-todo-assessment.git (push)
```

If you see your GitHub URL, it worked! ✅

Step 6.6: Push to GitHub

```
git push -u origin main
```

What this does:
- Uploads your code to GitHub

- `-u` sets up tracking (so future pushes are easier)
- `origin` is your GitHub repo (from step 6.5)
- `main` is the branch name

You might see:
1. **A prompt for username and password:**
- Username: Your GitHub username
- Password: ⚠️ Use a Personal Access Token (NOT your GitHub password)
- See “Part 7: Troubleshooting” below for how to create a token

1. **Upload progress:**
   ```
   Enumerating objects: 100, done.
   Counting objects: 100% (100/100), done.
   Delta compression using up to 8 threads
   Compressing objects: 100% (80/80), done.
   Writing objects: 100% (100/100), 50.00 KiB | 5.00 MiB/s, done.
   Total 100 (delta 20), reused 0 (delta 0)
   To https://github.com/YOUR_USERNAME/echo-todo-assessment.git
   ```
   - [new branch] main -> main
     Branch ‘main’ set up to track remote branch ‘main’ from ‘origin’.
     ```

**If you see this, YOUR CODE IS ON GITHUB!** 🎉🎉🎉

---

# Part 7: Verify Your Submission

## Step 7.1: View Your Repository on GitHub

1. Go to your browser
2. Navigate to: `https://github.com/YOUR_USERNAME/echo-todo-assessment`
3. Replace `YOUR_USERNAME` with your actual GitHub username

**You should see:**
- Your project files (backend/, frontend/, README.md, etc.)
- A file browser interface
- Your commit message
- Green “Code” button

## Step 7.2: Check the README

1. On your repository page, scroll down
2. You should see the README.md content displayed
3. It should show:
   - Project title: “Todo Application - ECHO Contractor Assessment”
   - Features list
   - Installation instructions
   - API documentation

**If you see all this, your submission is complete!** ✅✅✅

---
     ```

Part 8: Send to ECHO Recruiters

Step 8.1: Copy Your Repository URL

From your GitHub repository page, copy the URL from the browser address bar.

It should look like:

```
https://github.com/YOUR_USERNAME/echo-todo-assessment
```

Note: Remove the `.git` at the end if it's there.

Step 8.2: Compose Your Email

Send an email to the ECHO recruiter with:

Subject:

```
ECHO Todo Assessment Submission - [Your Name]
```

Body:

```
Hi [Recruiter Name],

I've completed the ECHO contractor technical assessment. Please find my submission be-
low:

📦 GitHub Repository: https://github.com/YOUR_USERNAME/echo-todo-assessment

📋 Project Overview:
- Full-stack todo application built with TypeScript
- Backend: Node.js + Express.js with RESTful API
- Frontend: React + Redux Toolkit with Vite
- Features: CRUD operations, category management, filtering, sorting
- Complete documentation in README.md with setup instructions

✅ All user stories and technical requirements have been implemented:
- Todo CRUD operations with title, description, due date
- Category management and assignment
- Grouped view by categories
- Complete/incomplete toggling
- Filtering by status (all, active, completed)
- Sorting by due date and creation date
- Error handling and input validation
- Redux Toolkit for state management
- Sample data for immediate testing

The project includes detailed setup instructions in the README. You can clone and run
it locally with:
```

git clone https://github.com/YOUR_USERNAME/echo-todo-assessment.git
cd echo-todo-assessment
npm install
npm run dev

```
Thank you for considering my application. I'm excited about the opportunity to work
on the Echo Platform at Amazon Robotics.

Best regards,
[Your Name]
[Your Email]
[Your Phone Number]
```

Replace:
- `[Recruiter Name]` with their actual name
- `YOUR_USERNAME` with your GitHub username
- `[Your Name]` with your full name
- `[Your Email]` and `[Your Phone Number]` with your contact info

🆘 Troubleshooting Common Issues

Issue 1: "Permission denied (publickey)" or Authentication Failed

Problem: GitHub is rejecting your credentials.

Solution: Create a Personal Access Token

1. **Go to GitHub Settings:**
   - Log into GitHub
   - Click your profile picture (top right)
   - Click "Settings"
   - Scroll down to "Developer settings" (bottom of left sidebar)
   - Click "Personal access tokens" → "Tokens (classic)"
   - Click "Generate new token" → "Generate new token (classic)"

2. **Configure the Token:**
   - Note: `ECHO Assessment Upload`
   - Expiration: `30 days` (or longer)
   - Select scopes:
     - ✅ Check `repo` (this checks all sub-boxes)
     - Scroll down and click "Generate token"

3. **Copy the Token:**
   - You'll see a long string like: `ghp_xxxxxxxxxxxxxxxxxxxx`
   - ⚠️ **COPY IT NOW!** You can't see it again!
   - Save it somewhere safe (password manager, notes app)

4. **Use Token Instead of Password:**
   - When `git push` asks for a password, paste the token
   - Username: Your GitHub username
   - Password: Paste the token (ghp_xxxxx...)

Issue 2: “fatal: remote origin already exists”

Problem: You tried to add a remote that’s already added.

Solution:

```
git remote remove origin
git remote add origin YOUR_GITHUB_URL
```

Issue 3: “fatal: not a git repository”

Problem: You’re not in the right folder or Git isn’t initialized.

Solution:
1. Check you’re in the right folder:
`bash`
`pwd`
Should show: `/home/ubuntu/todo_assessment`

1. If not, navigate there:
   `bash`
   `cd /home/ubuntu/todo_assessment`
2. Initialize Git:
   `bash`
   `git init`

Issue 4: “Updates were rejected” or “non-fast-forward”

Problem: GitHub has different code than your local version.

Solution:

Option A: Force push (USE CAREFULLY):

```
git push -u origin main --force
```

Option B: Pull first, then push:

```
git pull origin main --allow-unrelated-histories
git push -u origin main
```

Issue 5: “The repository already exists”

Problem: You created the GitHub repo WITH a README, and it conflicts.

Solution:
1. Delete the repository on GitHub:
- Go to your repo on GitHub
- Click “Settings” (tab)
- Scroll to bottom → “Danger Zone”
- Click “Delete this repository”
- Type the repository name to confirm

1. Create a new repository:
   - Follow Part 2 again
   - ❌ **DO NOT** check “Initialize with README”

Issue 6: “npm install” Fails When Testing

Problem: Dependencies won’t install.

Solution:
1. Check Node.js version:
`bash`
`node --version`
Should be v18 or higher.

1. Clear npm cache:
   `bash`
   `npm cache clean --force`

2. Delete node_modules and reinstall:
   `bash`
   `rm -rf node_modules package-lock.json`
   `rm -rf backend/node_modules backend/package-lock.json`
   `rm -rf frontend/node_modules frontend/package-lock.json`
   `npm install`

Issue 7: “Port 3000 already in use”

Problem: Another application is using port 3000.

Solution:

On Mac/Linux:

```
lsof -ti:3000 | xargs kill -9
```

On Windows:

```
netstat -ano | findstr :3000
taskkill /PID [PID_NUMBER] /F
```

Issue 8: Git Commands Don’t Work

Problem: Terminal says “command not found” for git.

Solution:
1. Verify Git installation:
```
bash
  git --version
```

1. If not installed, go back to Part 3 and install Git

2. If installed but not recognized, restart Terminal:
   - Close Terminal completely
   - Open it again
   - Try the command again

📚 Git Commands Reference

Basic Commands You’ve Used:

Command	What It Does	When to Use
`git init`	Start tracking a project with Git	First time setting up
`git add .`	Stage all files for commit	Before committing
`git commit -m "message"`	Save changes with a description	After staging files
`git remote add origin URL`	Connect to GitHub	First time linking to GitHub
`git push -u origin main`	Upload to GitHub	To share your code
`git status`	Check what’s changed	Anytime to see status
`git log`	View commit history	To see past saves

Useful Future Commands:

Command	What It Does
`git pull origin main`	Download latest changes from GitHub
`git clone URL`	Download a repository from GitHub
`git branch`	List branches
`git checkout -b new-branch`	Create and switch to new branch
`git diff`	See what changed

🎉 Congratulations!

If you've followed this guide, your ECHO todo assessment is now on GitHub and ready for submission!

Final Checklist:

- ✅ GitHub account created
- ✅ Repository created (public, named `echo-todo-assessment`)
- ✅ Git installed and configured
- ✅ Code pushed to GitHub
- ✅ README visible on GitHub
- ✅ Repository URL copied
- ✅ Email sent to ECHO recruiters

What Happens Next?

1. ECHO recruiters will review your code on GitHub
2. They'll look at:
   - Code organization and quality
   - TypeScript usage
   - Redux implementation
   - API design
   - UI/UX
   - Documentation
3. They may clone and run your project locally
4. You might be invited for an interview

Good Luck! 🍀

You've completed a full-stack TypeScript application with:
- RESTful API backend

- React + Redux frontend
- Proper state management
- Category organization
- Filtering and sorting
- Error handling
- Complete documentation

This demonstrates strong full-stack development skills. Great work!

Need More Help?

- **Git Documentation:** https://git-scm.com/doc
- **GitHub Guides:** https://guides.github.com/
- **Git Command Cheatsheet:** https://education.github.com/git-cheat-sheet-education.pdf

---

This guide was created to make GitHub submission accessible to everyone, regardless of technical background. If you found it helpful, you're exactly who it was made for! ❤️