

Complete Step-by-Step Guide: From Empty GitHub Repo to Final Submission

1. Fix Your Local Project Folder

- Ensure your project folder contains the required files: index.html, styles.css, script.js, and optionally README.md.
- Create them manually if they do not exist.
- Open the folder in VS Code. This becomes your real project root.

2. Connect Your Local Folder to GitHub Correctly

- Open terminal inside your project folder.
- Initialize a local git repository: `git init`
- Link the folder to your GitHub repo: `git remote add origin <your GitHub URL>`
- Synchronize your local folder with the remote: `git pull origin main`
- Your local and remote are now aligned.

3. Create the Real Project Structure

- Build the HTML skeleton layout.
- Add placeholders for weather info (temperature, humidity, wind, city, icon).
- Add search bar area.
- Add containers for loading and error messages.
- Keep CSS and JS empty for now.
- Commit and push the skeleton to GitHub.

4. Create the 5 Branches for Group Members

- Create the branches in your terminal:
- `git branch member1-html`
- `git branch member2-css`
- `git branch member3-api`
- `git branch member4-ui`
- `git branch member5-logic`

- Push each branch to GitHub to allow parallel teamwork.

5. Establish the Contracts (Leader's Most Important Duty)

- HTML contract: list element IDs, classes, and containers for JS interactions.
- CSS contract: define background classes (.sunny, .rainy, .cloudy, .night) and utility classes.
- JS contract: define structure of weather data (temperature, humidity, wind, condition).
- Create a contracts.md file and push it.

6. Create Mock Data for Members 4 and 5

- Define mock weather data structure:
- city, temperature, humidity, wind, condition.
- This allows DOM updates and search logic development before the real API is ready.
- Commit and push the mock data.

7. Team Begins Parallel Development

- Member 1: improve and finalize HTML.
- Member 2: start CSS styling.
- Member 3: integrate real API.
- Member 4: connect DOM updates using skeleton + mock data.
- Member 5: implement search logic using mock data.
- Team works with no waiting thanks to your scaffolding and contracts.

8. Start Integration Routine (Leader's Weekly Job)

- Week 1 integration: merge HTML + CSS + mock API and fix naming mismatches.
- Week 2 integration: unify real API → DOM updates → search logic.
- Week 3 integration: final polishing, responsiveness fixing, unused code cleanup.

9. Deploy to Netlify

- Log in to Netlify.
- Connect your GitHub repository.
- Deploy the main branch.
- Test the deployed version on multiple devices.

10. Final Documentation & Submission

- Update README with project description, installation instructions, roles, screenshots.
- Export Word/PDF documents.
- Submit GitHub link, Netlify link, and all documentation.
- Your entire workflow is complete.

Short Version Summary

- Build folder → sync with GitHub
- Create skeleton → push
- Create branches → push
- Define structure & naming → push
- Create mock data → push
- Everyone starts work
- Weekly integration
- Deploy
- Document
- Submit