## Final Project

For this assignment, you will submit the Github repo link for your final project. When you're ready, please download the template located below this window, fill it out, and submit it as your assignment.

To get a developer certification for the course, your project must:

1. Following this naming format: [**https://github.com/YOUR\_GITHUB\_USERNAME\_HERE/blockchain-developer-bootcamp-final-project**](https://github.com/YOUR_GITHUB_USERNAME_HERE/blockchain-developer-bootcamp-final-project)
2. **Contain a README.md file** which describes the project, describes the directory structure, and where the frontend project can be accessed (see #8). ***Please also include your public Ethereum account if you would like to receive your certification as an NFT (this is optional).***
3. **Contain smart contract(s)** which:
   * Are **commented** to [the specs described here](https://docs.soliditylang.org/en/latest/natspec-format.html)
   * Use at least **two design patterns** from the "Smart Contracts" section ([SEE A LIST OF DESIGN PATTERNS HERE](https://docs.google.com/document/d/1tthsXLlv5BDXEGUfoP6_MAsL_8_T0sRBNQs_1OnPxak/edit))
   * **Protect against two attack vectors** from the "Smart Contracts" section with its [SWC number](https://swcregistry.io/) ([SEE A LIST OF ATTACK VECTORS HERE](https://docs.google.com/document/d/1tthsXLlv5BDXEGUfoP6_MAsL_8_T0sRBNQs_1OnPxak/edit))
   * Inherits from at least **one library or interface**
   * Can be easily compiled, ~~migrated~~ and tested (see #5) (we just need to compile and test it locally.)
4. **Contain a Markdown file named design\_pattern\_decisions.md and** **avoiding\_common\_attacks.md** describing your design patterns and security measures. [You can see a list of the design patterns and security issues discussed in the course here.](https://docs.google.com/document/d/1tthsXLlv5BDXEGUfoP6_MAsL_8_T0sRBNQs_1OnPxak/edit?usp=sharing)
5. **Have at least five unit tests for your smart contract(s)** that pass. In the code, include a sentence or two explaining what the tests are covering their expected behavior. You are not required to build unit tests for your frontend, just your smart contracts.
6. **Contain a deployed\_address.txt file** which contains the testnet address and network where your contract(s) have been deployed
7. **Have a frontend interface** built with a framework like React or plain HTML/CSS/JS that:
   * **Detects** the presence of MetaMask
   * **Connects** to the current account
   * **Displays information** from your smart contract
   * Allows a user to **submit a transaction** to update smart contract state
   * **Updates the frontend** if the transaction is successful or not
8. **Hosted on** [**Github Pages,**](https://pages.github.com/)[**Netlify**](https://www.netlify.com/)**,** [**Fleek,**](https://fleek.co/) [**Surge,**](https://surge.sh/) [**Heroku**](https://www.heroku.com/) or some other free frontend service that gives users a public interface to your decentralized application. That address should be in your README.md document.
9. In your README.md, be sure to **have clear instructions** on:
   * **Installing dependencies** for your project
   * **Accessing or—**if your project needs a server (not required)—**running your project**
   * **Running your smart contract unit tests** and which port a local testnet should be running on.
   * *Note: This section used to require three bash scripts but has been revised.*
10. **A screencast of you walking through your project,** including submitting transactions and seeing the updated state. You can use a screenrecorder of your choosing or something like Loom, and you can share the link to the recording in your README.md

**Remember: DO NOT UPLOAD SENSITIVE INFORMATION TO GITHUB OR A PUBLIC SITE!** Your Infura account details, MetaMask mnemonics, any private keys, etc., should all be in a .env file which you add to your .gitignore in your project locally. In your README.md, you should instruct the user on how to populate the .env locally with their own information. [Read more about that here!](https://blog.infura.io/how-to-use-dotenv-to-enhance-basic-security-within-your-dapp/)

**A note on style:** In this course, we emphasize **functionality** and **security** over style. This is not a course in interface design, it is a course in Ethereum and decentralized application development. We simply want you to demonstrate what you have learned throughout the course, even if it's very straightforward.

**You will be able to commit updates to your repository until 11:59 PM** [**AoE time**](https://www.timeanddate.com/time/zones/aoe) **on November 30th**