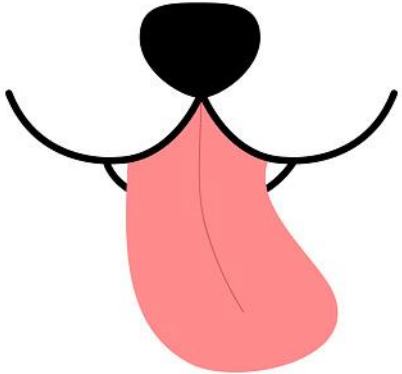
Dawgly

A Dog Services Application

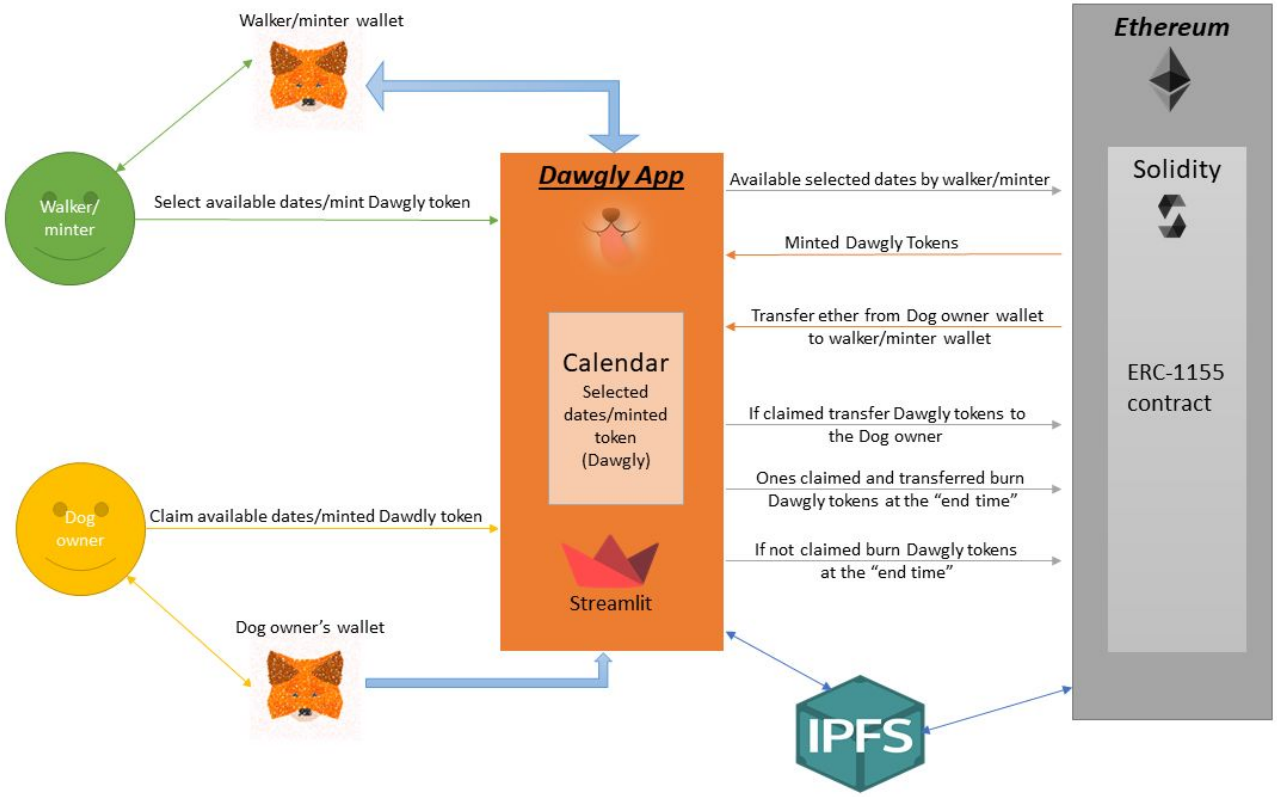
Bailey Richterman, Alex Park, Jose Sampedro, Kian Momeni, Melaku Melaku

Executive Summary

Dawgly is a dog services app that will connect doggy parents to our team members. We also created a Dawgly token that represents our team members' availability (1 token = 1 hour).



Executive Summary

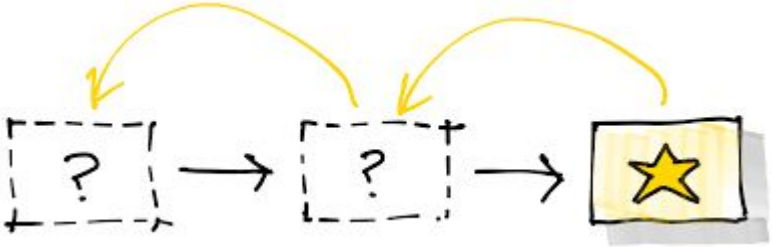


Data Collection/Preparation

- No outside data is included in our current application
- Reviewed ERC contracts (20, 721, 1155)
  - Decided to use ERC-1155 because it allows “for each token ID to represent a new configurable token type, which may have it own metadata, supply, and other attributes.”
  - ERC 1155 is a smart contract that represents multiple tokens at once.
  - Allows for more efficient trades and bundling of transactions - thus saving costs.
  - A single ERC 1155 token contract can hold the entire systystem state.

Our Approach

1. Front end application
   a. Helped us gather our thoughts and create a concise plan as we started to develop the more difficult parts of the app
2. Dawgly Token/Smart Contract
   a. The token represents the team member's time availability
3. Integrate the two



Sample Case

The Clients

Customer -1	1 Dog	Hires -> Alex
Customer -2	2 Dogs	Hires ->Bailey

Walkers

Alex - Availability

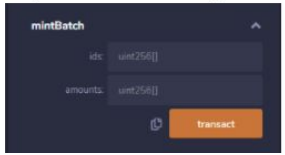
Hrs Available	Days Available	Pets	Total Hours
11	7	1	77

Bailey - Availability

Hrs Available	Days Available	Pets	Total Hours
11	1	2	22

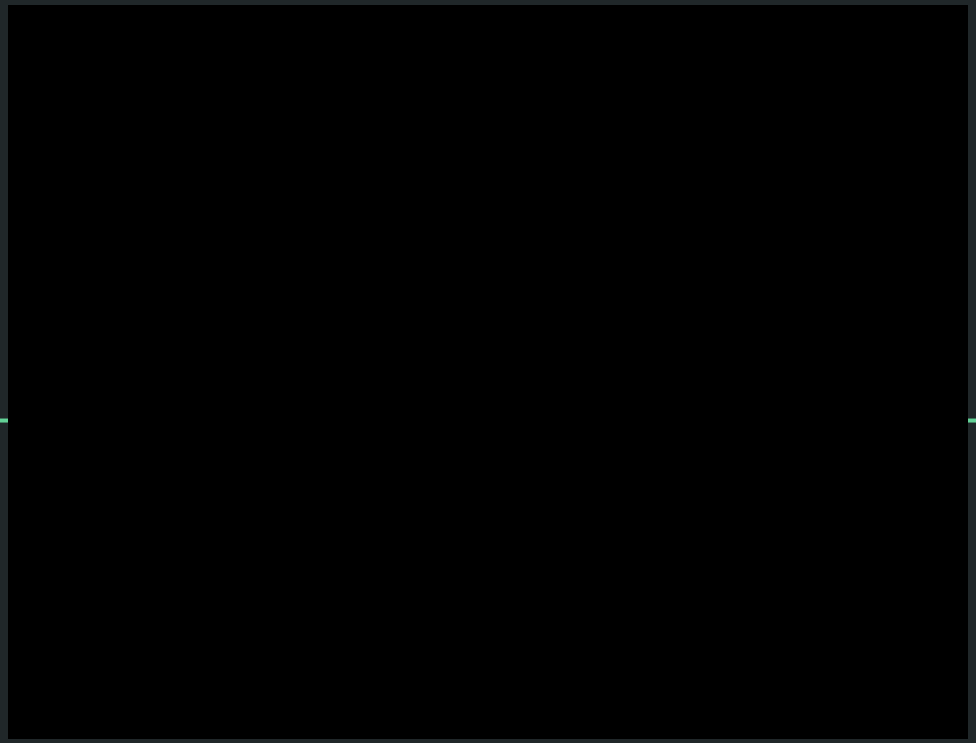
	Hours	1	2	3	4	5	6	7	8	9	10	11
Walker Name	**Time**	**9:00**	**10:00**	**11:00**	**12:00**	**13:00**	**14:00**	**15:00**	**16:00**	**17:00**	**18:00**	**19:00**
Alex	**Token ID**	9	10	11	12	13	14	15	16	17	18	19
	Token Value	1	1	1	1	1	1	1	1	1	1	1
Bailey	**Token ID**	9	10	11	12	13	14	15	16	17	18	19
	Token Value	2	2	2	2	2	2	2	2	2	2	2

Baley would enter the following values

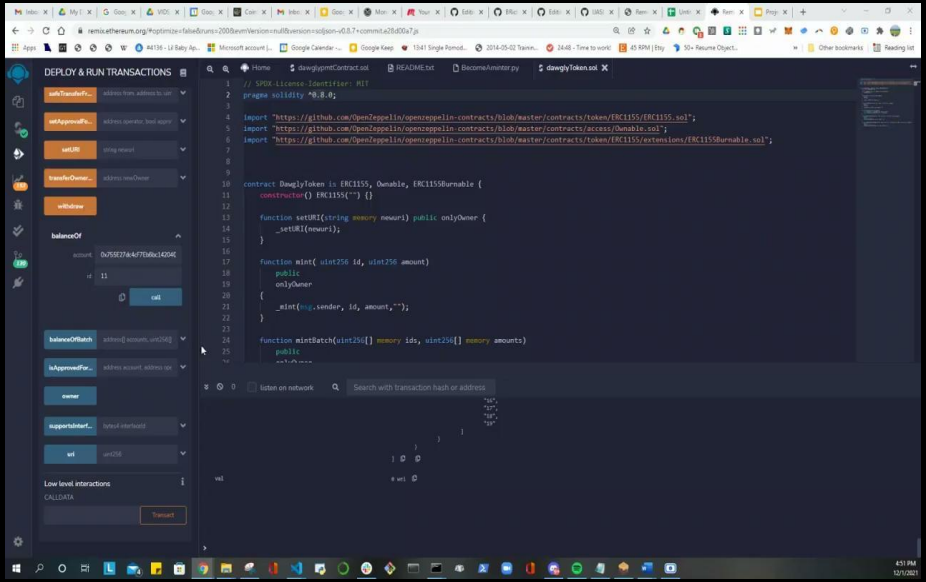


ids:
[9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
amounts:
[2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]

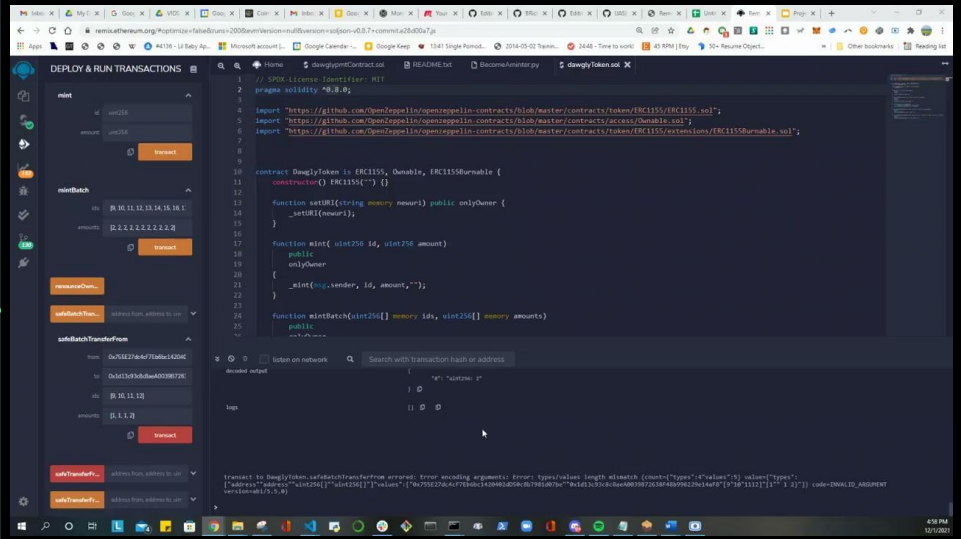
Demo - Mint



Demo - Walker Balance

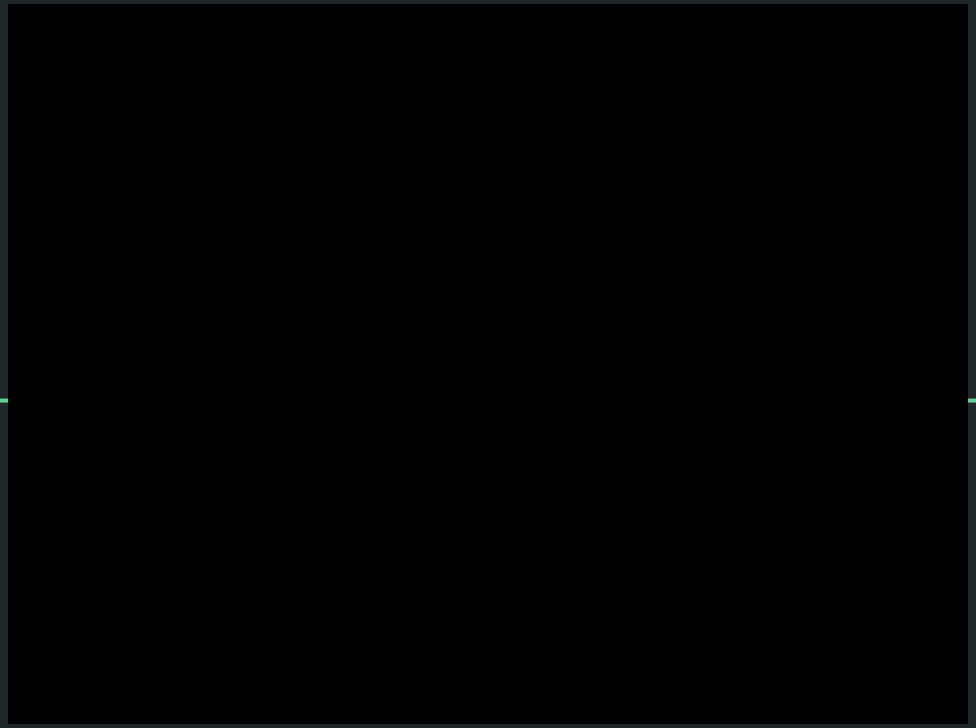


Demo - Token Claim/Transfers



Demo - Client Balance

Demo - Front End



Conclusion

- Our app successfully asks users about the size of their dog, the type of service they would like, date, time, and length of time
- Our app successfully allows for users to mint our team members' tokens (time) in exchange for the desired dog service
  - At the end of the selected time frame the token will be burned
- We fully demonstrated our knowledge of Python, Solidity, Streamlit, Blockchain, and Smart Contracts

Next Steps

- Make the user interface more user friendly
  - Make the integration of the front end app and the smart contract more seamless (ie. make time and date selection more user friendly...connect selected time commitment to the # of tokens)
  - Improve the app aesthetic
- Build out login dropdown to allow various users to join
  - Right now we have our “accounts” pre-set in our code
- Have the app automatically contact our team members when their time has been reserved

Questions?

Vote for Team 6!

Appendix

https://github.com/BRichterman/Team6_Project3

https://docs.openzeppelin.com/contracts/3.x/erc1155

https://ethereum.org/en/developers/docs/standards/tokens/

https://docs.openzeppelin.com/contracts/3.x/erc1155