Product Evaluation Document

Aaron, Gabe G., Gabe L., Nick

Part 1: Testing

In this section, we will describe the various forms of vigorous testing the Solien Gallery website has gone through in order to answer the question: “To what extent does Solien Gallery behave as we intended?” The first method of testing we implemented was internal testing. The initial element of our internal tests is catching function failures. We accomplished this through a robust automated error handling system that is designed to maximize the details logged on the server side, while being as vague as possible with the error returned to the front end to hide our backend implementation from the user. At its core, this system catches a Python exception, logs it, determines what the appropriate vague error code is to send to the client and then sends it to them. The implementation for this can be found in server.py. In addition to our error handling, we also utilize assertions throughout our Python modules to practice defensive programming techniques and validate function parameters.

To check for invariants, we focused heavily on the frontend. Through manual testing, we observed what happens to the HTML in the gallery page after 50 additions to the gallery view and subsequent deletions of Solien NFTs. In between these additions and deletions, randomized shuffling of the order of Soliens within the gallery view takes place as well as random background changes. Through the comparison of the website source code, these tests revealed no change in the HTML between the tests. On the backend, we directed our invariant testing towards the server’s handling of HTTP requests. To do this, we repeatedly attempted to query the blockchain and download a random gallery over a period of several minutes. These tests did not reveal any bugs.

In terms of external testing, we utilized both white and black box tests.
Within white box tests, we implemented path testing. To do this, we wanted to proceed through every possible logical flow that could be produced from user input as well as potentially impossible backend flows. Through a manual process, we proceeded through every possible front end gallery creation for a given NFT wallet. To do this, we worked through every possible background and Solien combination with the exception of the color picker background (which we used a random color for). To prevent a factorial explosion, this test did not go through a permutation of Soliens in the gallery as that would increase

the number of tests by a factor of n! where n is the number of Soliens in the wallet. We also repeated this test with the same wallet after manually uncaching it, causing us to query the blockchain for its data repeatedly. (It took a painstaking amount of time to do the uncached tests by hand.) These tests did not reveal any bugs. Further included in our path testing was an attempt to generate all error codes that could be triggered through the backend error handling system (400, 404, 500, 505). Error 400 was received after submitting a wallet that contains no NFTs. Error 404 was received after entering an invalid wallet address. Error 500 and 505 could not be triggered through the frontend, and required proactive fault injection (intentionally modifying and breaking our code).

Boundary testing was conducted with manual tests. In the boundary tests, we sought to test our corner cases. On the index pages this included: entering an empty string into the search bar, wild cards characters, various special characters and attempts to inject malicious code. We then tested creating galleries with the first and last wallet address listed in our cache. These tests did not reveal any additional bugs. However, through use of the coverage tool we conducted statement tests and did not achieve 100% coverage. However, this is not a major concern as it does not affect our intended functionality.

Andrew Simon ‘23, the maintainer of the official Solien NFT website (https://soliens.io/) agreed to consult as a third party to conduct “Black-Box” testing in exchange for a box of Sriracha Steak Hot Pockets from the Princeton U-store. Andrew conducted his tests manually. To stress test the system, he opened the index page in 10 windows and quickly attempted to proceed to the gallery creation page on all 10 windows with different wallet addresses. All 10 pages reacted appropriately. To handle use case testing, he went through the variety of use cases in an early version of our User Guide, but was unable to find any bugs.

To further test our program, an automated testing client was created called “testclient.py.” This testing client contains additional stress tests that attempt to overwhelm our website functioning as a weak D.O.S. attack. This script also contains a handful of unit tests for img_util.py. Lastly, automated tests attempt to alter the parameters passed through the URLS. This test is consistent through our automated tests (returns the same HTTP response code). However, through further manual testing various bugs arise. Thus, our only known bugs are entirely contingent upon a user modifying URL values.

Upon submitting an HTTP GET request with corrupted URL parameter values for the Gallery Builder page, the client will not throw an error and instead display the usual Gallery Builder page without NFTs. Submitting an HTTP GET request with corrupted URL parameter values will break the Download, enqueue gallery processing and loading pages as well. This suggests that the only fragile aspect of the Solien Gallery website is its use of URL parameters. Besides this, the website behaved as expected suggesting the conclusion that the rest of the site is robust and that it behaves almost exactly as we intended it to behave.

Part 2: Evaluation by Users

In this section, we will describe some of the feedback we have received from real users, most principally out of concern for answering the question, *How well does your product meet the needs of its users*. In the appendix of this document, we have included the tasks we asked our users to complete in addition to some questions we had them answer as a sort of survey. Throughout this process, we observed our users as they performed these tasks while listening to them narrate their actions. From this, we learned a few things in particular in the form of overwhelmingly positive feedback. Broadly speaking, the positive feedback we received seemed to have its origins in a common idea: that our website quite directly and visibly meets the needs of our users. Our idea for this project was conceived directly for the purpose of meeting the desire of Solien NFT users to be able to create a Twitter banner out of a gallery of the NFTs that they owned. We are proud to say that our website does exactly that, as are our users, evidently.

More concretely, some of our positive feedback consisted of the following: comments on the minimalist aesthetics and theme of our website, comments on the ease and intuitive nature of use, and affirmation of their enthusiasm to use this to create a Twitter banner in practice. For example, one user commented, “Oh that’s clean,” while another commented, “I like the minimalism,” both referring to the prevailing aesthetic of our website throughout its pages. Regarding the ease of use, one user remarked that “It’s all very straightforward and intuitive,” referring to our website in general. Our observations (and ensuing notes) of the users completing steps 1-5 (which are listed in our appendix) reflect this sentiment.

Additionally, we observed that our users seem to be quite entertained by the different options they are presented with for gallery customization, as demonstrated through the amount of time spent trying out galleries with different combinations of features and the apparent enthusiasm exhibited. One thing we were also interested in observing among our users was whether they could tell the difference in time necessary to perform the search function between wallets containing NFTs that are cached compared to wallets where one NFTs is not cached. We had mixed reactions in this regard, with some users stating that they observed a notable difference, while some said that they didn’t. Regarding the speed of our website in general, we were met with both positive and ambivalent responses. One user remarked that they were “Very impressed by the speed at which the account with 75 NFTs had its images loaded,” referring to the results of searching for wallet address “Dp32k9fx6mDo7FnvGaqaPDHX9fPEQXyYKUevjrypGkF8” on our homepage. On the other hand, other users remarked that the amount of time spent on the loading pages went by not entirely without notice (although they did mention that they were entertained by the phrases presented on the loading screen!). We acknowledge that the amount of time spent on the loading pages can at times be non-trivial, although we would like to appeal to the fact that this is largely out of our control, due to limitations imposed by blockchain traversal in addition to Cloudinary’s servers.

In addition to this mostly positive feedback, we also received feedback that was more constructive than merely positive. Namely, we were told that it would be useful if the gallery previewed on the gallery creator page (entitled “Gallery Builder”) would be displayed again on the gallery creator page at the end of the following sequence of actions: 1) the user composes a gallery on the gallery creator page; 2) the user proceeds to the download page; 3) the user returns to the gallery creator page. In response to this constructive feedback, we implemented a feature which accomplishes this. More concretely, we made changes to the gallery creator page code such that clicking “Create” opens a new tab with the download page (rather than remaining within the existing tab). With this setup, the user can switch to the gallery creator window, where they will still find the gallery that they have composed. Additionally, while acknowledging that we succeeded in terms of the scope of this course by stating, “Great job, guys. I’m really pleased with what you’ve done,” one user also noted, “I think there is potential to build upon this platform to include image-creating features with a purpose other than generating a Twitter banner. I look forward to seeing how Solien Gallery progresses!”

This experience was tremendously valuable to us. Throughout the semester, we provided and received feedback, but almost exclusively from the course staff as well as among ourselves. This experience showed us the importance of getting feedback directly from our users themselves. We look forward to receiving more feedback in the future from our users using our prepopulating contact-us email feature!

Part 3: Evaluation by Experts

In this section, we will describe where our product stands relative to formal evaluation techniques, specifically relative to the *heuristic evaluation* technique. Here we will consider whether our system fulfills the following 10 heuristics, where doing so is a measure of the quality of the product:

1. **Visibility of system status**

Our product consistently upholds this standard, namely through keeping “users informed about what is going on, through appropriate feedback within reasonable time” (Software Engineering Lecture Part 3, slide 49). Our system demonstrates our commitment to this throughout our website, especially once a user has searched for their wallet address on our homepage and we display a loading screen prior to the gallery creator screen, once a user has clicked to create a gallery they’ve composed on the gallery creator page and a loading page is displayed prior to reaching the download page, and on our gallery creator page itself, where our adding/removing/rearranging features are sufficiently elegant as to make it immediately and abundantly clear to the user what is going on.

2. **Match between system and the real world**

   Our product consistently upholds this standard, namely through speaking “the user's language, with words, phrases and concepts familiar to the user, rather than system-oriented terms” (Software Engineering Lecture Part 3, slide 50). This is best exemplified on our index page, where we use terms such as “wallet address,” which the user is likely very familiar with as a Solien owner, and on our loading pages, where we include clever space-related phrases the user may have been exposed to and might find funny, such as “Sending Request to Mars.”

3. **User control and freedom**

   Our product consistently upholds this standard, namely through addressing the fact that “Users often choose system functions by mistake and will need a clearly marked ‘emergency exit’ to leave the unwanted state without having to go through an extended dialogue” by supporting “undo and redo” (Software Engineering Lecture Part 3, slide 51). This is evidently upheld in the case of our search bar on our home page, where the user can easily replace the wallet address entered, and through nearly all of the features on our gallery creator page, where the user can undo and redo the additions or removals of NFTs and can change the background image in various different ways at any point while on the gallery creator page. It is perhaps this principle that our product must robustly embodies.

4. **Consistency and standards**

   Our product consistently upholds this standard, namely through recognizing that “Users should not have to wonder whether different words, situations, or actions mean the same thing” and thus we “Follow platform conventions” (Software

Engineering Lecture Part 3, slide 52). Throughout our website, consistent terminology is used for things such as “wallet address” and “gallery.”

5. **Error prevention**

   Our product consistently upholds this standard, namely through designing our system to “Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action” (Software Engineering Lecture Part 3, slide 53). For example, it should not be possible to create an empty gallery, meaning one without any NFTs added onto it. As such, rather than directing the user to a loading page upon attempting to create an empty gallery, our product does not change the page it’s on upon receiving a request to create an empty gallery. Thus, it is clear to the user that this error-filled request they are making requires that they create a gallery with some NFTs on it.

6. **Recognition rather than recall**

   Our product consistently upholds this standard, namely through minimizing “the user's memory load by making objects, actions, and options visible” (Software Engineering Lecture Part 3, slide 54). This is clear on our download page, which previews an image of the gallery the user will likely soon download. As a result, what the user is required to remember about the gallery is minimized at every stage of our system.

7. **Flexibility and efficiency of use**

   Our product upholds this standard, namely allowing “users to tailor frequent actions,” to a certain degree (Software Engineering Lecture Part 3, slide 55). The reason that this is only to a certain degree is because our product has generally been optimized to the best of our knowledge for all users. For example, the data on the NFTs of nearly all Soliens has already been cached in our database optimizing system. And it is in this sense that our system doesn’t speed things up for expert users, since processes are already sped up for everyone!

8. **Aesthetic and minimalist design**

   Our product consistently upholds this standard, namely through recognizing the fact that “Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility” (Software Engineering

Lecture Part 3, slide 56). Throughout our entire website, we have carefully crafted an aesthetic that is both elegant and minimalist. As such, we only present the most essential information to the user. This is the case in our searching function on our homepage, on the gallery creator page, and on the downloading page. This positive sentiment has been reflected consistently in the feedback we have received from users.

9. **Help users recognize, diagnose, and recover from errors**

   Our product consistently upholds this standard, namely through recognizing the fact that "Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution" (Software Engineering Lecture Part 3, slide 57). This sentiment is implemented throughout our website, whether the error is encountered on the index page resulting from the search functionality or elsewhere.

10. **Help and documentation**

    Our product consistently upholds this standard, namely through recognizing the fact that "Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large" (Software Engineering Lecture Part 3, slide 58). More concretely, our website on the homepage clearly indicates what action to take to get started in addition to including an About The Project section to indicate to the user the purpose of the webpage. Additionally, the gallery creator page demonstrates to the user how they can proceed to the downloading page by presenting the "Create" button. Additionally, our contact email functionality in addition to our User's Guide allows the user to contact us if necessary and to investigate questions they might have about use cases, respectively.

Appendix

Note: For the task list, we presented the text presented below to the user in addition to describing the steps to provide some supplementary information about our website, including, but not limited to, a discussion of Solien Gallery's purpose.

User task list:

Repeat the following 5 steps with the following 3 wallet addresses:
Ca9ugqnNQisHaJzzGSYT6orWtbdYwK4qnKs5D2PhmXYC,
6KQDNrJoPJRa1UHX7C4Wf5FHgjvnswLMTePyUTySFKeQ,
Dp32k9fx6mDo7FnvGaqaPDHX9fPEQXyYKUevjrypGkF8
(If you have your own wallet with Soliens feel free to use it, as well)

1) Navigate to our website, https://sol333.herokuapp.com/
2) Enter the given wallet address where you see "Wallet Address"
3) Click "Search."
4) Once the gallery page (https://sol333.herokuapp.com/gallery) has loaded, add NFTs to the gallery background in the middle of the page by clicking on them. Feel free to rearrange the order via the drag and drop functionality. Feel free to customize the gallery background by clicking on the 5 options. The fifth option is unique in that it allows you to choose any color as your background. Click on images in the gallery in the middle of the page to remove them from the gallery if you want. When the gallery is to your liking, click "Create."
5) When you've arrived at the download page (https://sol333.herokuapp.com/download) click on "Click Here to Download Image." When you've arrived at the Cloudinary page, feel free to download your gallery by right clicking on the image and then saving it.

User Questions/Survey:

1) Do you have any comments on the homepage? Is the user interface intuitive? Is the contact information at the bottom of the page helpful?
2) Do you have any comments about the gallery creator page? Is the user interface intuitive? Do you have any comments on the options for the backgrounds? Do you have any other comments on making your gallery more customizable?
3) Do you have any comments about the download page?
4) Do you have any comments about the loading pages? Did it ever take too long to load?
5) Did you observe any differences in user experience when using the three different wallet addresses?
6) If you own a Solien NFT, try it out with your own wallet. How was the user experience?
7) Do you have any comments on the aesthetics of our website? Is there something about the design that you liked? Is there something that you think could be improved?

8) Would you use this website to create a banner for your Twitter (if you owned a Solien NFT)? Would you recommend the website to your friends? If you responded “No” to either of these questions, why not?
9) Do you have any other comments or suggestions? Is there something that was especially impressive? Was there something that stood out in needing to be improved?