Steganography Project Master Test Plan

Linden Crandall, Jonathan Mainhart, Zhihua Zheng

University of Maryland Global Campus

CMIS 495: Current Trends and Projects in Computer Science

Prof. Majid Shaalan

April 29, 2022

Introduction

This document contains test cases used to measure functionality of the application. The tests were run on four different systems listed in table 1. Automated unit test cases for the backend (application logic) and frontend (user interface) are contained in table 2 and table 3 respectively. Manual integration tests are listed in table 4.

Descriptions and screen shots of the results of each test begin on page 11.

Test Systems

The application was tested on four separate systems listed in table 1. These systems represent the expected typical application user systems. Each system has Python version 3.9 installed. A Linux system was unavailable at the time of testing.

Table 1
Test Systems

System ID	OS (Version)	Processor	RAM
S1	macOS (12.3.1)	3.8 GHz 8-Core Intel i7	8 GB 2667 MHz DDR4
S2	macOS (12.1)	1.6 GHz Dual-Core Intel i5	8 GB 2133 MHz LPDDR3
S3	Windows 11 Home	2.9 GHz Intel Core i7	16GB 2933 MHz DDR4
S4	Windows 11	3.0 GHz Intel Core i7	16GB 3200 MHz DDR4

Test Cases

Table 2
Automated Tests (Back End)

#	Test Input	Expected	Pass/Fail				Fig. #
			S1	S2	S3	S4	
1	check max chars of 300x300 pix img	30,000	Pass	Pass	Pass	Pass	1
2	check max chars of 5x5 pix img	8	Pass	Pass	Pass	Pass	1
3	encode 5x5 pix img to max chars without truncation	‘01234567’	Pass	Pass	Pass	Pass	1
4	encode 5x5 pix image to max chars with overflow input ‘01234567ABCDEF’	‘01234567’	Pass	Pass	Pass	Pass	1
5	image and backup image attributes are exact copies	Attributes match	Pass	Pass	Pass	Pass	1
6	image encode changes pixels to not match backup image	Attributes do not match	Pass	Pass	Pass	Pass	1

7	check decoded message matches encoded message	Messages match	Pass	Pass	Pass	Pass	1
8	check the image can save to disk as a new file	new file saved	Pass	Pass	Pass	Pass	1
9	check the saved image decoded message matches the encoded message	Messages match	Pass	Pass	Pass	Pass	1
10	check that encoded image pixel values match original values after reset	Images reset	Pass	Pass	Pass	Pass	1
11	check convert message utility return value matches binary representation of test message	Values match	Pass	Pass	Pass	Pass	1

Table 3

Automated Unit Tests (Front End)

#	Test Input	Expected	Pass/Fail				Fig. #
			S1	S2	S3	S4	
1	on_open_button_click	ImageObject matches.	Pass	Pass	Pass	Pass	1
2	on_encode_button_click . class variable: enable_bool = True.	• Object display_image's method encode_image is called once with default text from TextField. • update_widgets_status is called with the arguments: (False, True, False).	Pass	Pass	Pass	Pass	1
3	on_encode_button_click . class variable: enable_bool = False.	• Method popip_user_notification is called with the arguments: ('Failed to execute encode function! \nPlease modify the text field input.',	Pass	Pass	Pass	Pass	1

		MainWidget.MESSAGE_TYPE.ERROR) •					
4	on_reset_button_click	• Variable warning_type maches MainWidget.WARNING_TYPE.RESET • Method popup_user_notification is called with the arguments: (‘Are you sure you want to reset the image?', MainWidget.MESSAGE_TYPE.WARNING)	Pass	Pass	Pass	Pass	1
5	execute_reset	• Object display_image’s method reset_image is called once.	Pass	Pass	Pass	Pass	1
6	on_save_button_clic	• Variable new_filename maches the String “expected”. • ImageChooserPopup Class method show_filechooser is called once.	Pass	Pass	Pass	Pass	1
7	on_save setup: new_filename is valid. overwrite with new_filename.	• Method popup_user_notification is called with the arguments: (‘Image name already exists. \nAre you sure you want to overwrite the image?', MainWidget.ME	Pass	Pass	Pass	Pass	1

		SSAGE_TYPE. WARNING)					
8	execute_save setup: decode_image method return string “decoded_msg” new_filename = 'test_image_3.jpeg' new_filepath = 'path'	• Object display_image’s method save_image is called once with arguments: ('path', 'test_image_3.jpeg') • Variable title maches the String “Steganosaurus – test_image_3.jpeg”. • main_image.source maches the String “path/ test_image_3.jpeg”. • Variable textfield_str matches “decoded_msg”.	Pass	Pass	Pass	Pass	1
9	validate_image_name setup: image_name = ‘test_image_3.jpeg'	• Method validate_image_name returns True. •	Pass	Pass	Pass	Pass	1
10	validate_image_name setup: image_name = ‘_test_image_3.jpeg'	• Method validate_image_name returns False. • Method popup_user_notification is called with the arguments: (‘Invalid file name! \n Only alphabet characters, numbers, dot, underscore and hyphens are allowed. (e.g.	Pass	Pass	Pass	Pass	1

		image_1)', MainWidget.MESSAGE_TYPE.ERROR)					
11	validate_image_name setup: image_name = '_test_image_3.jpeg'	• Method validate_image_name returns False. • Method popup_user_notification is called with the arguments: ('Invalid file name! \n Only alphabet characters, numbers, dot, underscore and hyphens are allowed. (e.g. image_1)', MainWidget.MESSAGE_TYPE.ERROR)	Pass	Pass	Pass	Pass	1
12	update_warning_btn_yes setup: warning_btn_yes = True self.warning_type == self.WARNING_TYPE.WARNINGSAVE	• Method update_textfield_input is called once. • Method execute_reset is called once. • Method update_textfield_input • is called once with arguments: (True, False, False)	Pass	Pass	Pass	Pass	1
13	update_warning_btn_yes setup: warning_btn_yes = True self.warning_type = self.WARNING_TYPE.RESET	• Method execute_reset is called once. • Method update_textfield_input	Pass	Pass	Pass	Pass	1

		• is called once with arguments: (True, False, True)					
14	update_warning_btn_yes setup: warning_btn_yes = False self.warning_type = self.WARNING_TYPE.RESET	• Method update_textfield_input is called once with arguments: (False, True, False) •	Pass	Pass	Pass	Pass	1
15	update_widgets_status setup: reset_btn_disabled = False textfield_disabled = True image_saver_dismiss = True	• Variable value reset_btn_disabled matches • Variable value textfield_disabled matches • Variable value image_saver_dismissmatches •	Pass	Pass	Pass	Pass	1
16	update_main_widgets setup: main_text_field.text = " display_image.max_available_chars = 100	• Variable value main_image.source matches • Variable value textfield_str matches • Variable value maximum_char_count matches • Variable value encodable_bool matches	Pass	Pass	Pass	Pass	1
17	update_main_widgets setup: display_image.decode_image returns “decoded_msg” display_image.max_available_chars = 11	• Variable value main_image.source matches • Variable value textfield_str matches • Variable value maximum_char_count matches	Pass	Pass	Pass	Pass	1

		• Variable value user_notification_msg matches • Variable value encodable_bool matches					
18	update_main_widgets setup: display_image.decode_image returns “decoded_msg” display_image.max_available_chars = 10	• Variable value main_image.source matches • Variable value textfield_str matches • Variable value maximum_char_count matches • Variable value user_notification_msg matches • Variable value encodable_bool matches	Pass	Pass	Pass	Pass	1
19	update_textfield_input setup: textfield_str ="somestring"	• Variable value main_text_field.text matches •	Pass	Pass	Pass	Pass	1

Table 4
Manual Tests

#	Test Input	Expected	Pass/Fail				Fig. #
			S1	S2	S3	S4	
1	Randomly selected image loads on launch	Image displays	Pass	Pass	Pass	Pass	2
2	Image is decoded when loaded	Decoded message displays	Pass	Pass	Pass	Pass	2
3	Open image button opens file chooser	File chooser displays	Pass	Pass	Pass	Pass	3
4	File chooser has read access to user file system	Able to navigate file system	Pass	Pass	Pass	Pass	3
5	Attempt to open non-image file	File chooser rejects non-image files	Pass	Pass	Pass	Pass	4
6	Cancel button in the File chooser view is pressed	Return to the main GUI	Pass	Pass	Pass	Pass	NA
7	Attempt to open image file	Image opens and is displayed	Pass	Pass	Pass	Pass	5
8	Attempt to enter test phrase into text input area	Text is displayed. Allowed character count decreases	Pass	Pass	Pass	Pass	6
9	Attempt to enter text greater than maximum allowed for the image	Warning message displays in the Main GUI.	Pass	Pass	Pass	Pass	7
10	Attempt to enter text greater than maximum allowed for the image, then encode image button is pressed.	Error dialog popups. Not encodable.	Pass	Pass	Pass	Pass	8
11	Enter valid characters in the textfield, then encode image button pressed	Text input is disabled Reset button is enabled	Pass	Pass	Pass	Pass	9
12	Save image button pressed	Save file dialog opens	Pass	Pass	Pass	Pass	10
13	Attempt to save file with invalid file name “123 ” which end with a space.	Error dialog popups.	Pass	Pass	Pass	Pass	11
14	Attempt to save file with incorrect extension “123.JNG”	New image is saved with the file name “123.JNG.png”. Successfully saved image dialog popups.	Pass	Pass	Pass	Pass	12

15	Attempt to save file as JPEG “123.jpeg”	New image is saved with filename “123.jpeg”	Pass	Pass	Pass	Pass	13
16	Save image button pressed. Select Cancel button in save dialog	Returned to main screen	Pass	Pass	Pass	Pass	NA
17	Overwrite existing filename	Warning pop up. Allows overwrite	Pass	Pass	Pass	Pass	14 15
18	Reset image after encoding. Then Yes button in the warning popup is pressed.	Warning pop ups. Image resets to original value	Pass	Pass	Pass	Pass	16 17
19	Reset image after encoding. No button in the warning popup is pressed.	Returned to main GUI.	Pass	Pass	Pass	Pass	NA
20	Open image containing message to decode	Decoded message displays in text area. File name is displayed in the title bar. Image is displayed in the main window.	Pass	Pass	Pass	Pass	18

Results

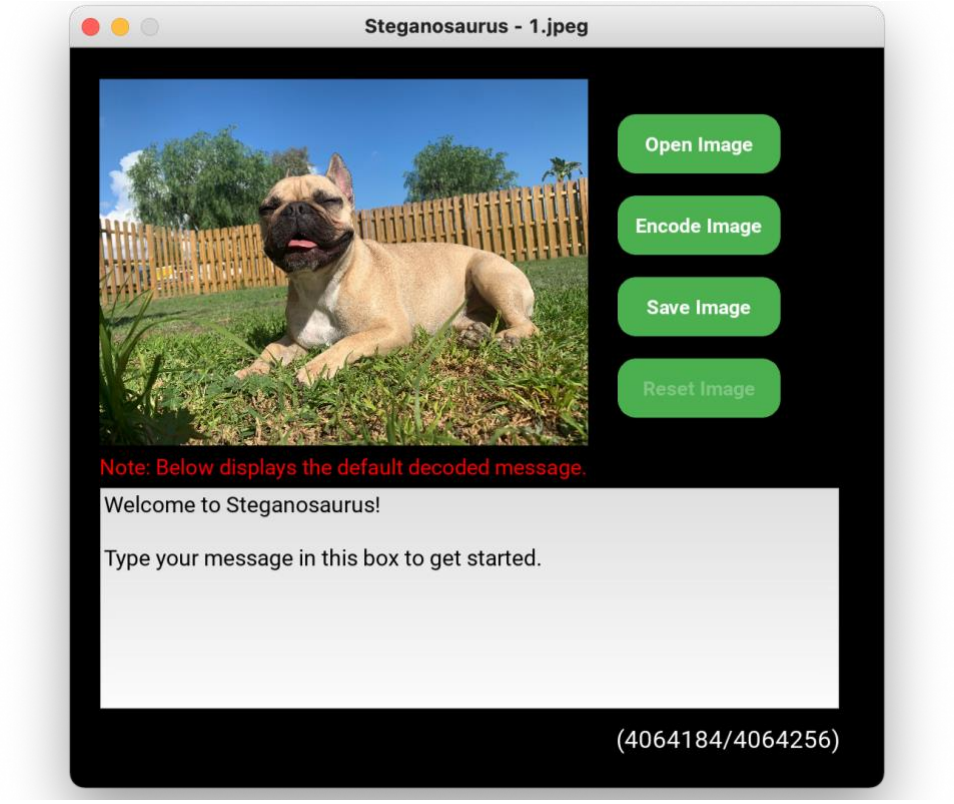
The automated tests were run on systems S1 – S4. All tests passed on every system. Figure 1 shows the results of the automated tests run on S2.

```
jonmainhart@CleverClog Steganosaurus % pytest steganosaurus/tests/test.py steganosaurus/tests/stegoTest.py -v
============================================= test session starts ==============================================
platform darwin -- Python 3.9.0, pytest-7.1.1, pluggy-1.0.0 -- /Library/Frameworks/Python.framework/Versions/3.9/bin/python3
cachedir: .pytest_cache
rootdir: /Users/jonmainhart/Documents/College/CMSC495/cmsc495_final/Steganosaurus
plugins: Faker-7.0.1
collected 30 items

steganosaurus/tests/test.py::TestImageObjectModel::test_max_char_90000_pix PASSED                    [  3%]
steganosaurus/tests/test.py::TestImageObjectModel::test_max_char_25_pix PASSED                       [  6%]
steganosaurus/tests/test.py::TestImageObjectModel::test_max_char_limit PASSED                        [ 10%]
steganosaurus/tests/test.py::TestImageObjectModel::test_max_char_overflow PASSED                     [ 13%]
steganosaurus/tests/test.py::TestImageObjectModel::test_image_attribute_match PASSED                 [ 16%]
steganosaurus/tests/test.py::TestImageObjectModel::test_encode_image PASSED                          [ 20%]
steganosaurus/tests/test.py::TestImageObjectModel::test_decode_image PASSED                          [ 23%]
steganosaurus/tests/test.py::TestImageObjectModel::test_save_image PASSED                            [ 26%]
steganosaurus/tests/test.py::TestImageObjectModel::test_saved_image_decode PASSED                    [ 30%]
steganosaurus/tests/test.py::TestImageObjectModel::test_reset_image PASSED                           [ 33%]
steganosaurus/tests/test.py::TestUtilities::test_convert_message PASSED                              [ 36%]
steganosaurus/tests/stegoTest.py::TestStego::test_execute_reset PASSED                               [ 40%]
steganosaurus/tests/stegoTest.py::TestStego::test_execute_save PASSED                                [ 43%]
steganosaurus/tests/stegoTest.py::TestStego::test_on_encode_button_click_false PASSED                [ 46%]
steganosaurus/tests/stegoTest.py::TestStego::test_on_encode_button_click_true PASSED                 [ 50%]
steganosaurus/tests/stegoTest.py::TestStego::test_on_open_button_click PASSED                        [ 53%]
steganosaurus/tests/stegoTest.py::TestStego::test_on_reset_button_click PASSED                       [ 56%]
steganosaurus/tests/stegoTest.py::TestStego::test_on_save_button_click PASSED                        [ 60%]
steganosaurus/tests/stegoTest.py::TestStego::test_save PASSED                                        [ 63%]
steganosaurus/tests/stegoTest.py::TestStego::test_save_2 PASSED                                      [ 66%]
steganosaurus/tests/stegoTest.py::TestStego::test_update_main_widgets_1 PASSED                       [ 70%]
steganosaurus/tests/stegoTest.py::TestStego::test_update_main_widgets_2 PASSED                       [ 73%]
steganosaurus/tests/stegoTest.py::TestStego::test_update_main_widgets_3 PASSED                       [ 76%]
steganosaurus/tests/stegoTest.py::TestStego::test_update_textfield_input PASSED                      [ 80%]
steganosaurus/tests/stegoTest.py::TestStego::test_update_warning_btn_yes_1 PASSED                    [ 83%]
steganosaurus/tests/stegoTest.py::TestStego::test_update_warning_btn_yes_2 PASSED                    [ 86%]
steganosaurus/tests/stegoTest.py::TestStego::test_update_warning_btn_yes_3 PASSED                    [ 90%]
steganosaurus/tests/stegoTest.py::TestStego::test_update_widgets_status PASSED                       [ 93%]
steganosaurus/tests/stegoTest.py::TestStego::test_validate_image_name_false PASSED                   [ 96%]
steganosaurus/tests/stegoTest.py::TestStego::test_validate_image_name_true PASSED                    [100%]

============================================== 30 passed in 40.33s ==============================================
```

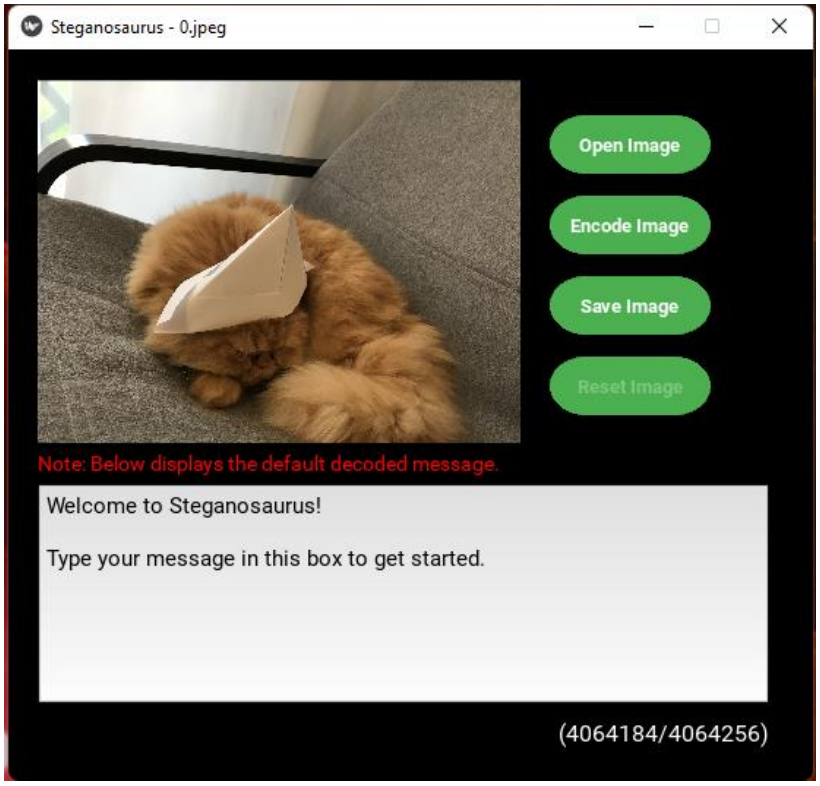
Figure 1. System 2 automated unit test results.

The 30 automated tests cover ImageObject methods used to encode, decode, calculate maximum message size, save, and reset images to their original state; binary encoding of messages; and front-end logic.

The manual testing of the application was successfully performed on all test systems. These tests included verifying that all buttons and fields worked as expected, that any expected user interaction worked as required, and that any unexpected user input was handled properly to prevent the application from crashing or operating outside of design parameters. Figures 2 through 18 show the output of the test cases contained in table 4.

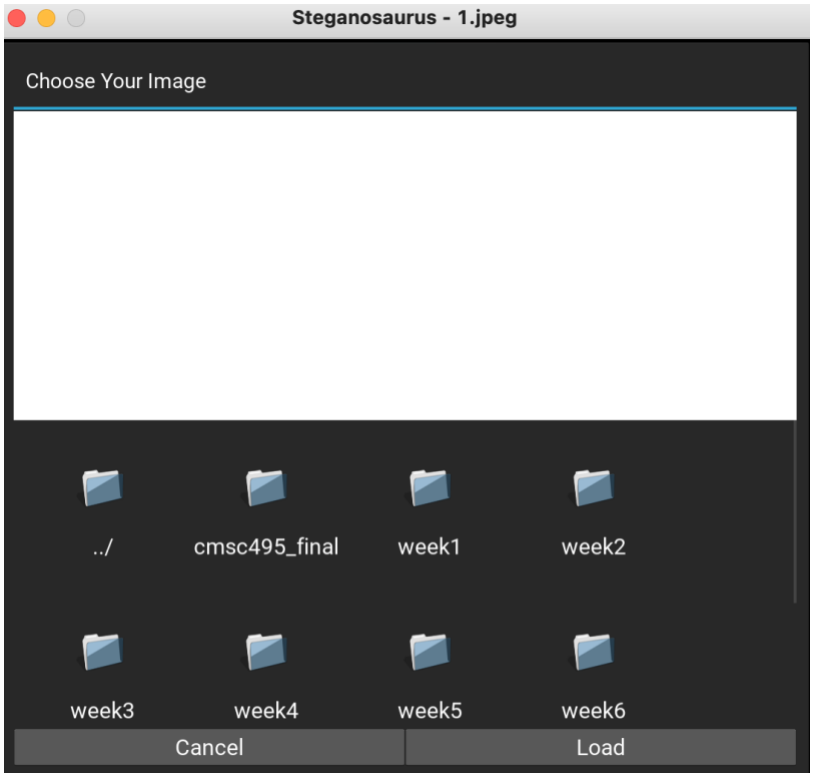


macOS

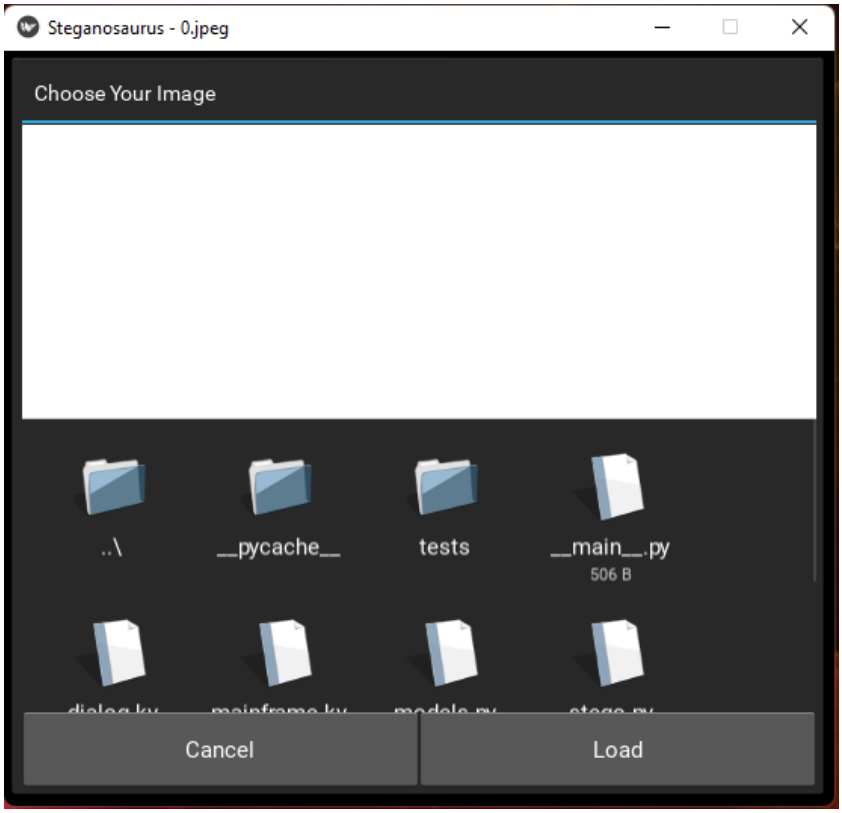


Windows

Figure 2. The application loads a randomized image when launched and decodes the secret message (manual test cases 4-1 and 4-2).

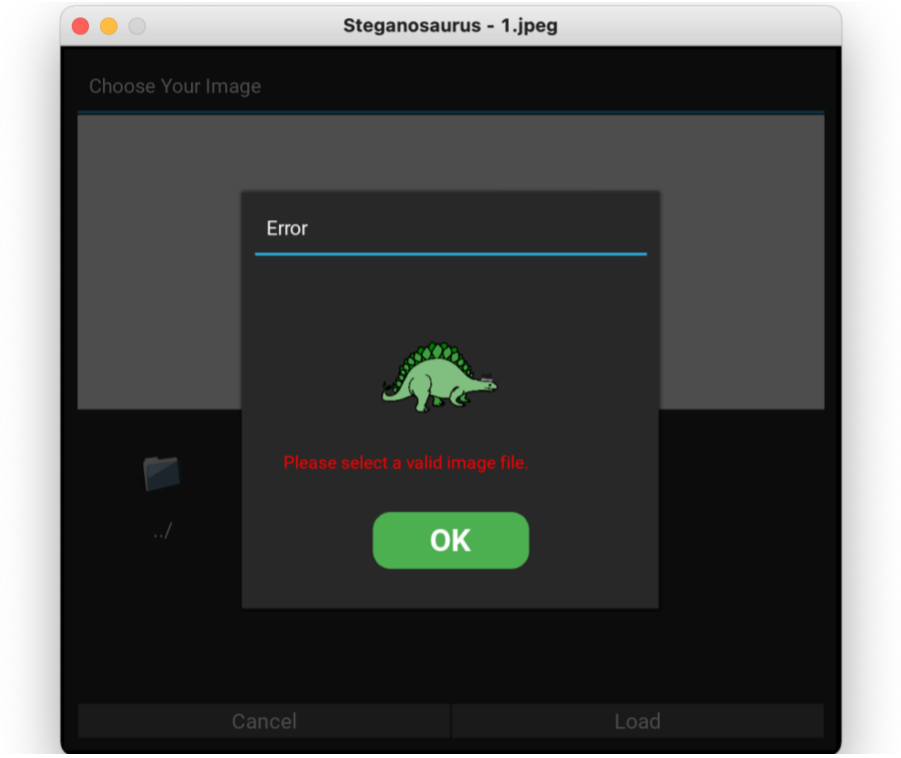


macOS

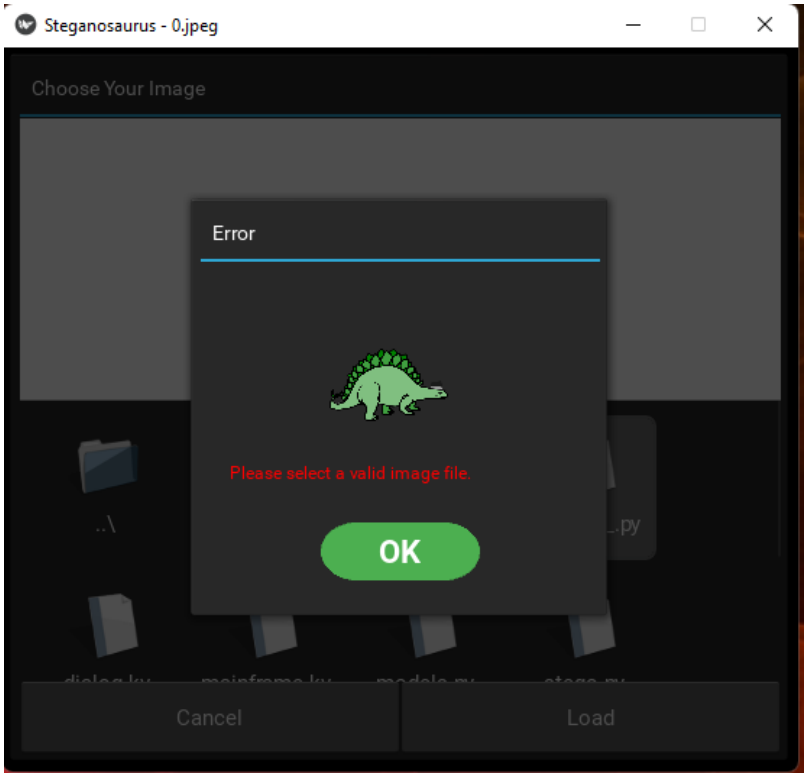


Windows

Figure 3. Results of clicking the "Open Image" button. The application has access to the file system (manual test cases 4-3 and 4-4).



macOS



Windows

Figure 4. Results of attempting to open a ineligible file (Test case 4-5).

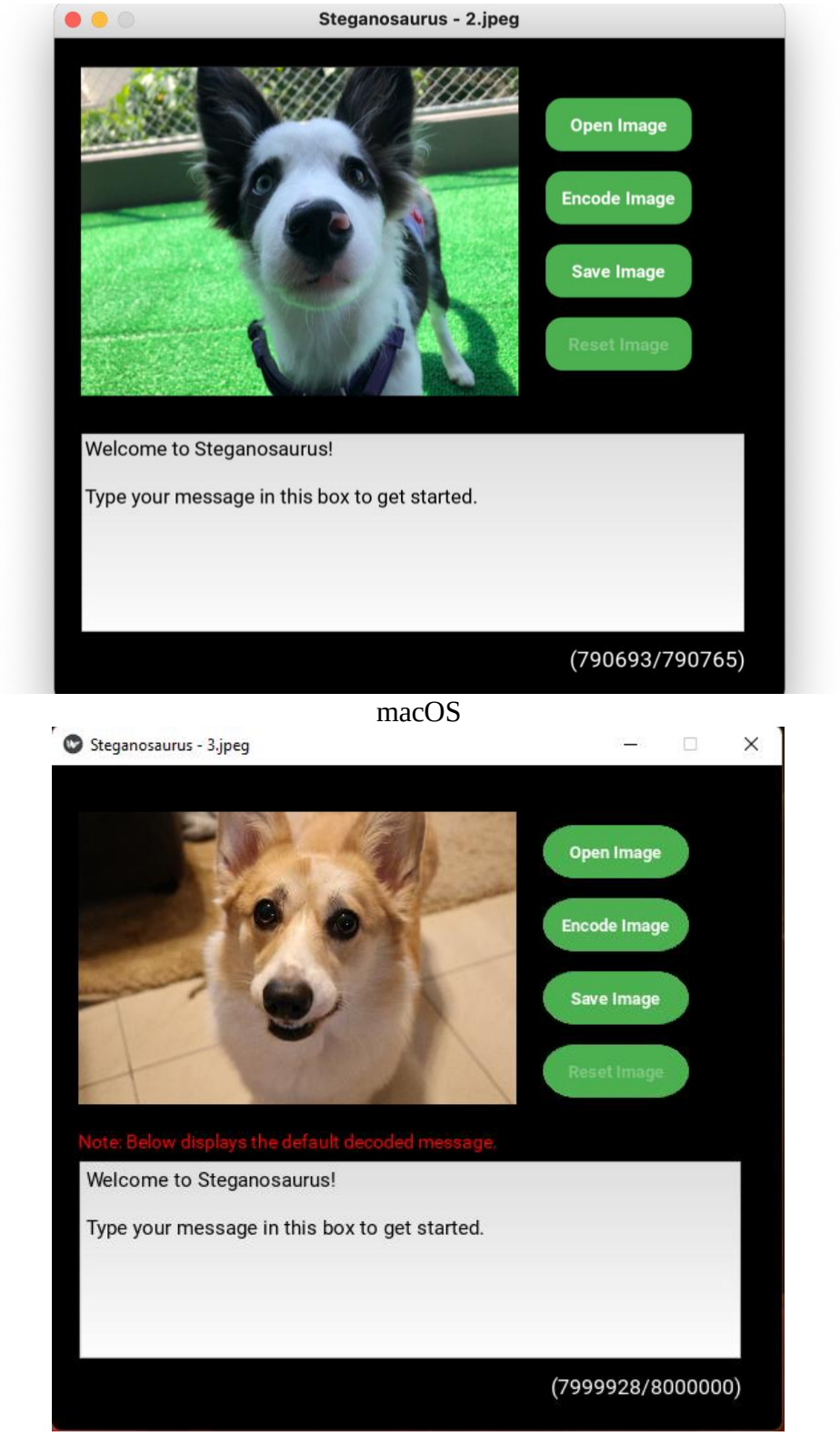
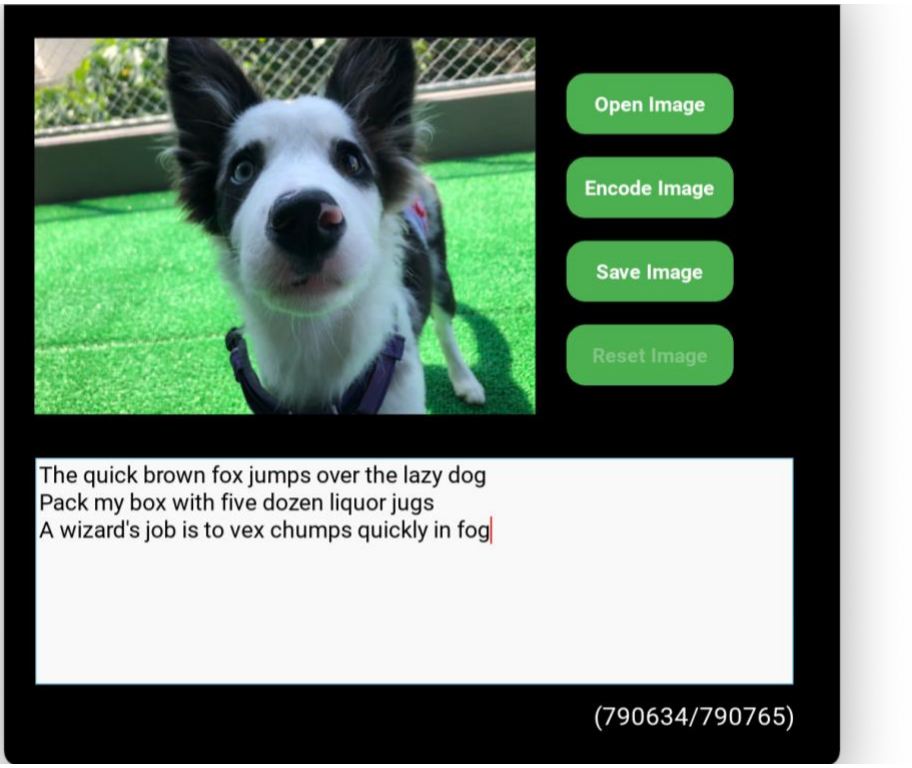
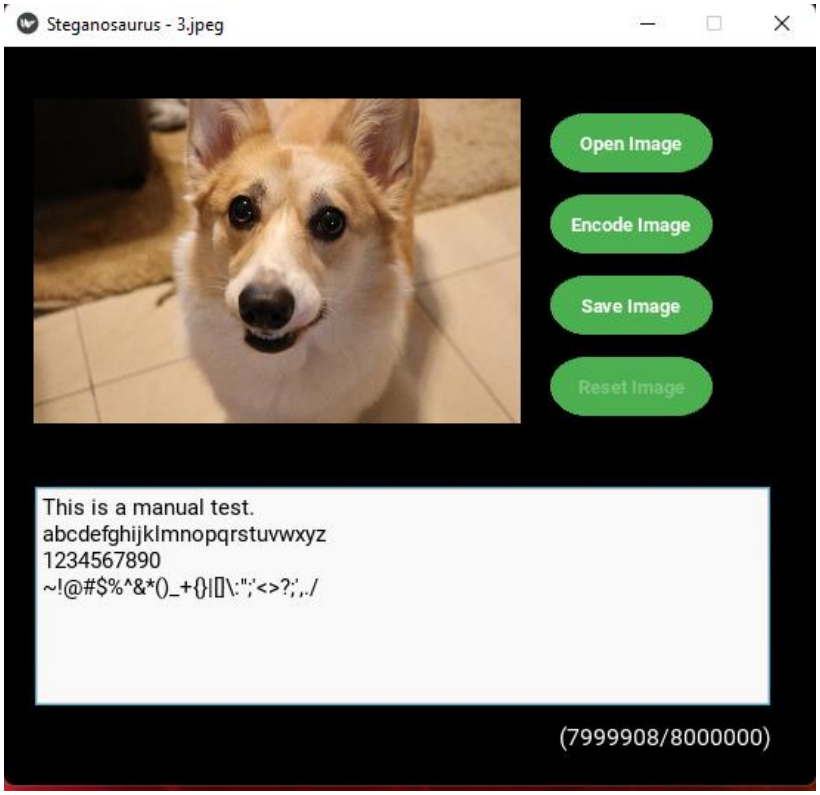


Figure 5. The application opens image files as expected (Test case 4-6).

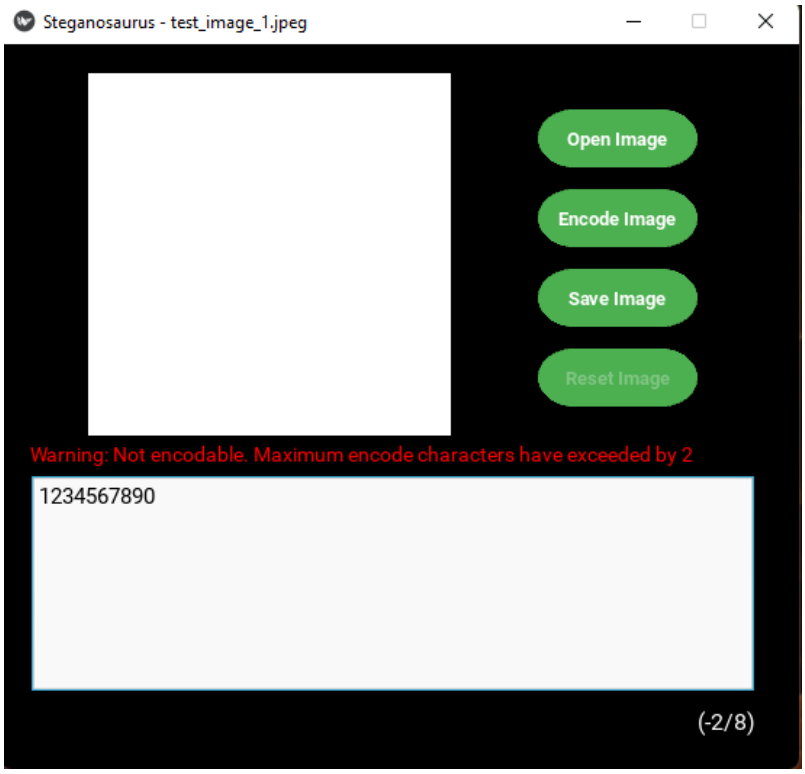


macOS



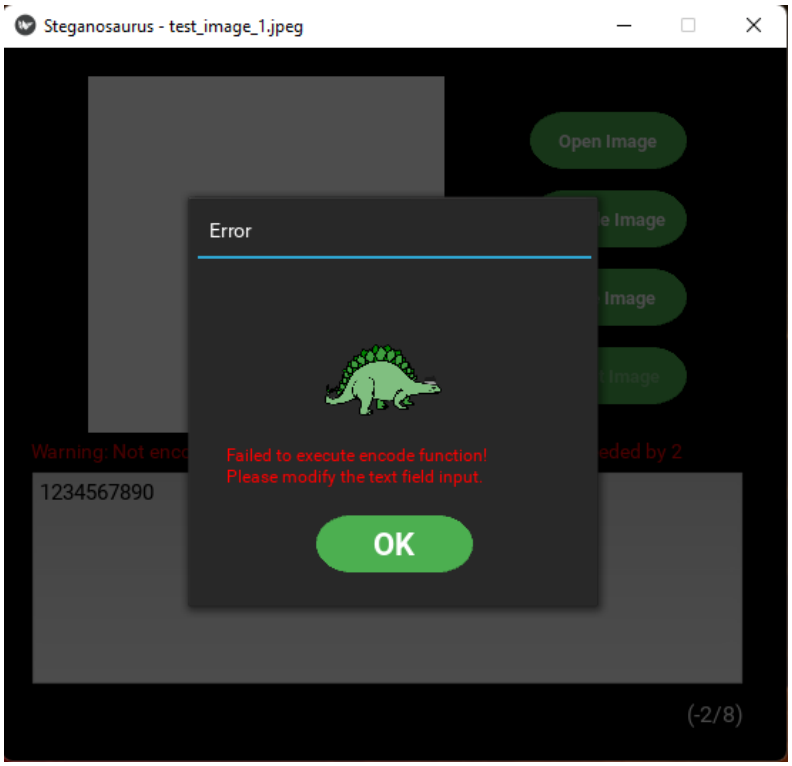
Windows

Figure 6. The remaining characters allowed decrease as text is input (Test case 4-8).



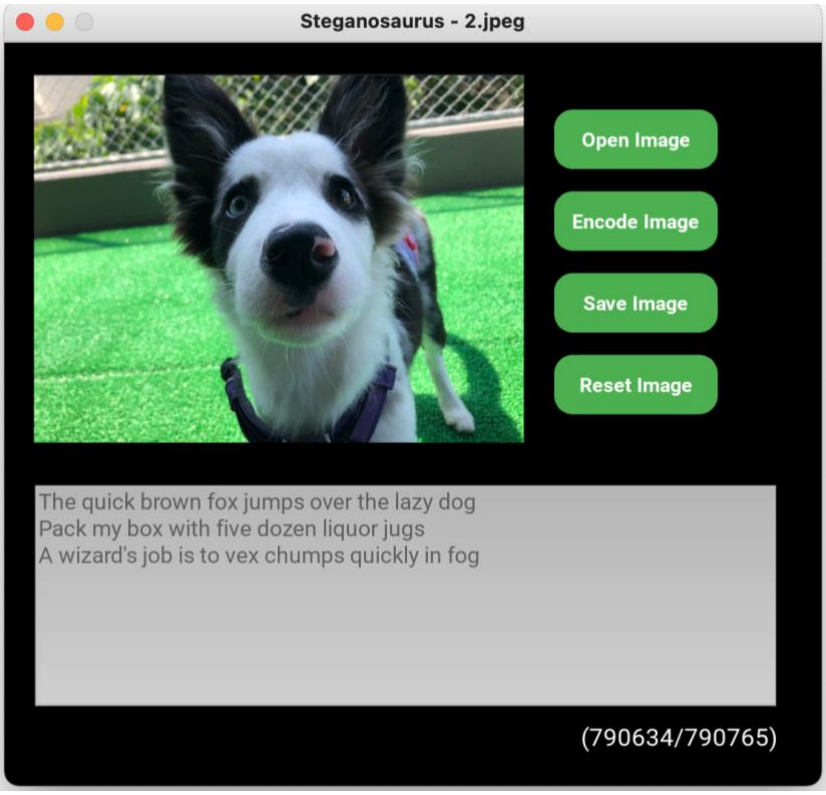
Windows

Figure 7. Entering too many characters displays a warning (Test case 4-9).

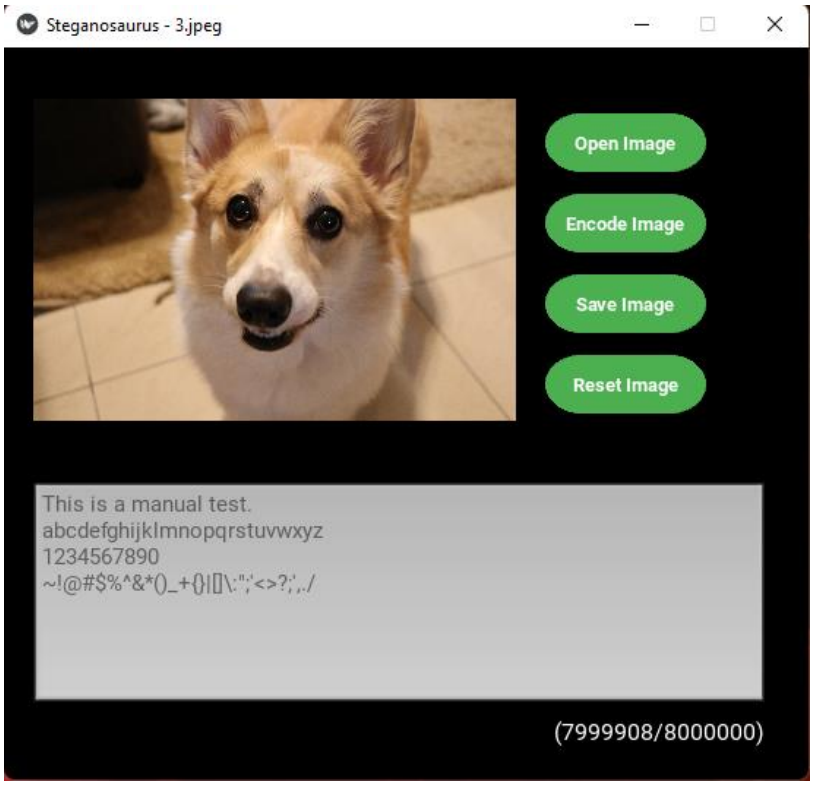


Windows

Figure 8. Attempting to encode an image with too many characters (Test case 4-10).

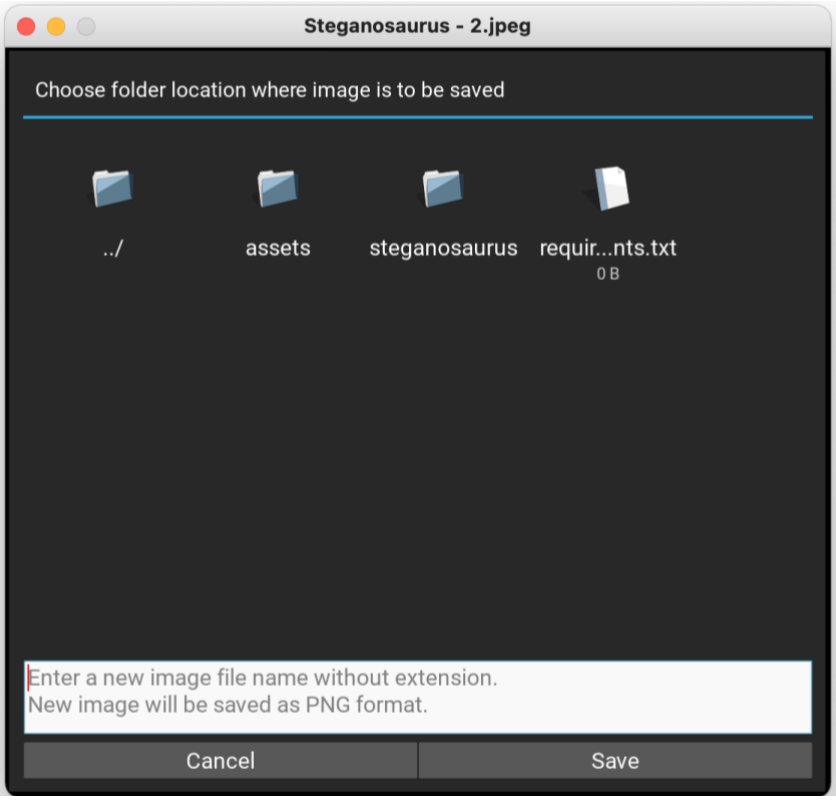


macOS

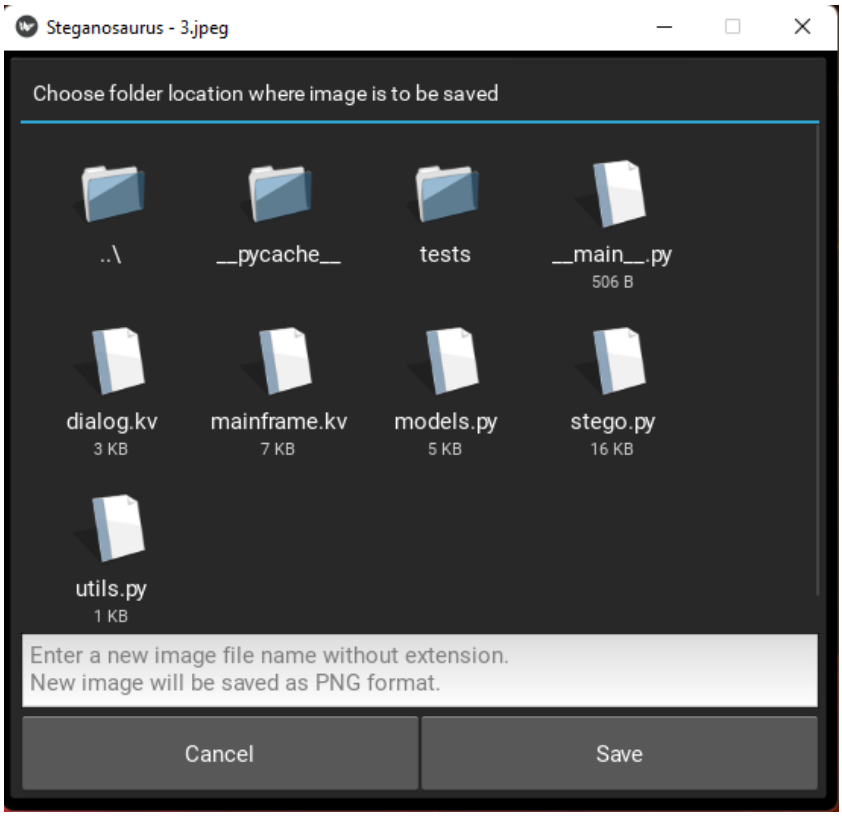


Windows

Figure 9. A message of the correct length will encode (Test case 4-11).

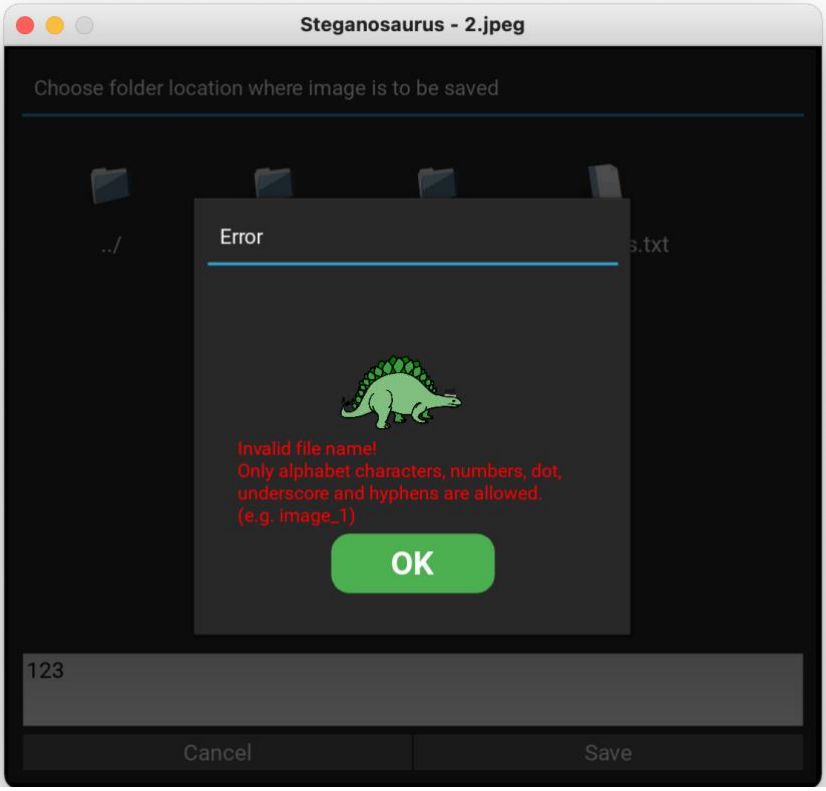


macOS

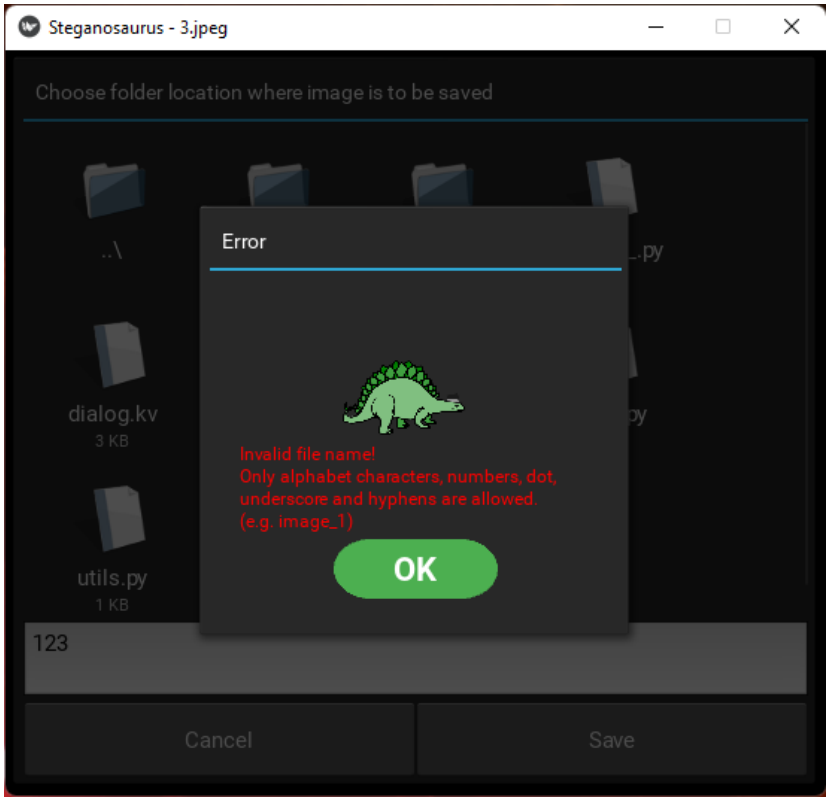


Windows

Figure 10. Save dialog (Test case 4-12).

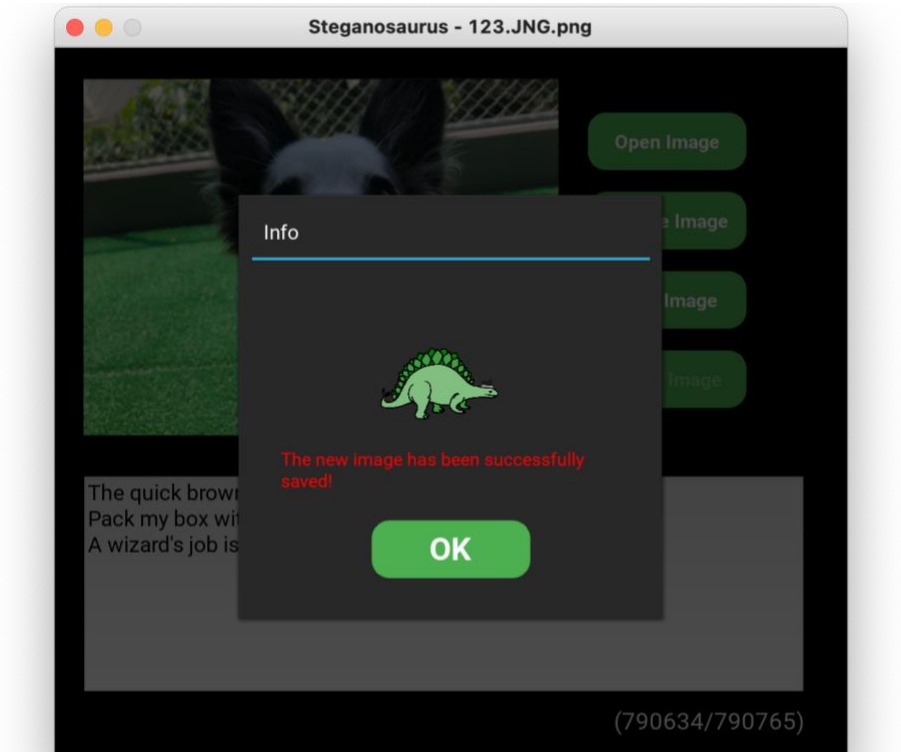


macOS

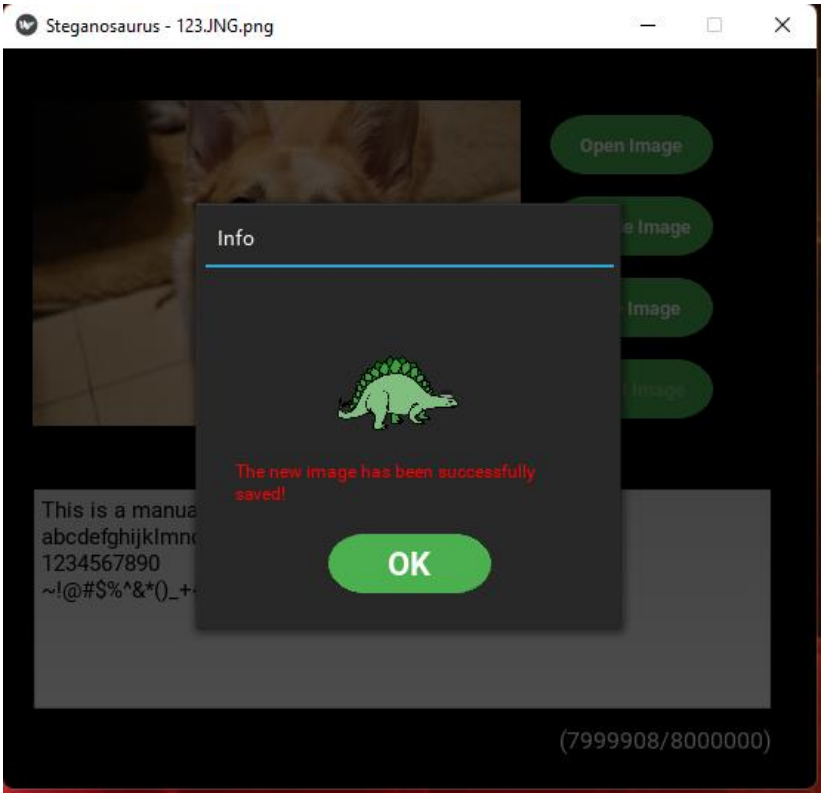


Windows

Figure 11. Attempting to save a file with an invalid name (Test case 4-13).

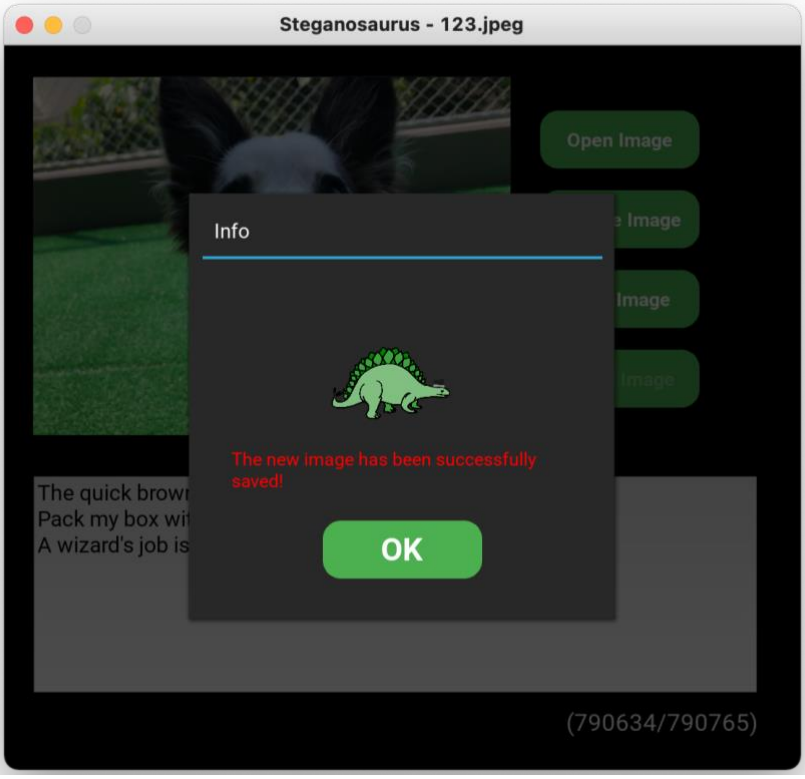


macOS

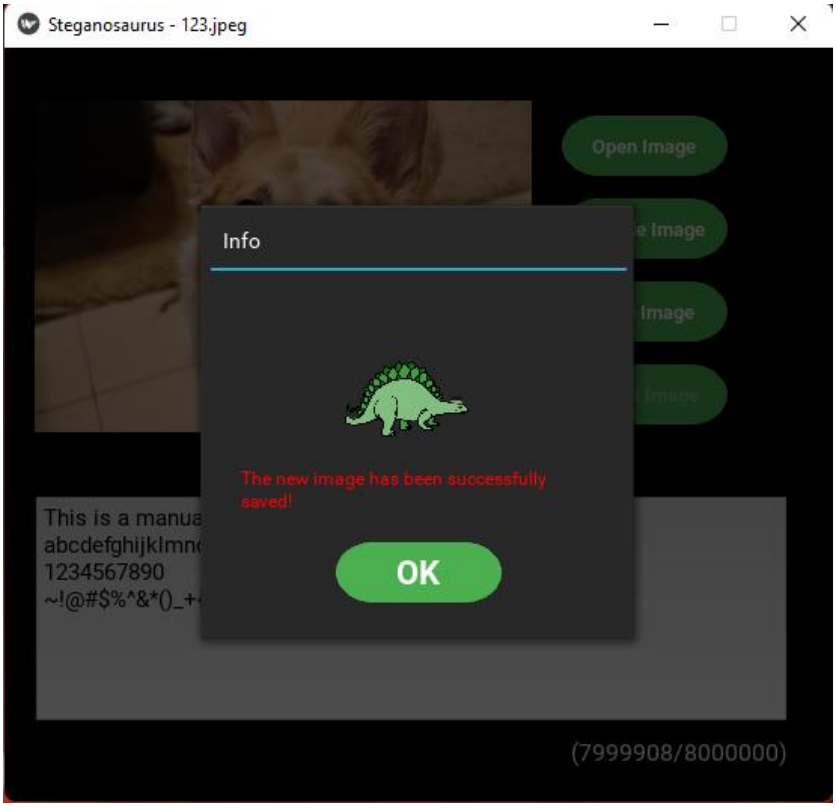


Windows

Figure 12. Successfully saving an image with default encoding (Test case 4-14).

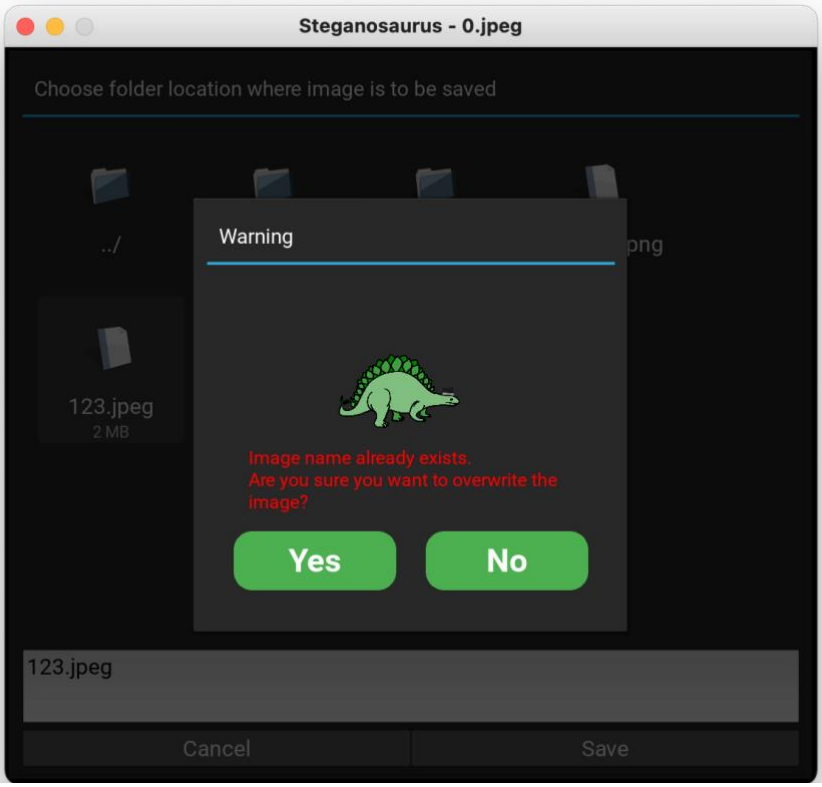


macOS

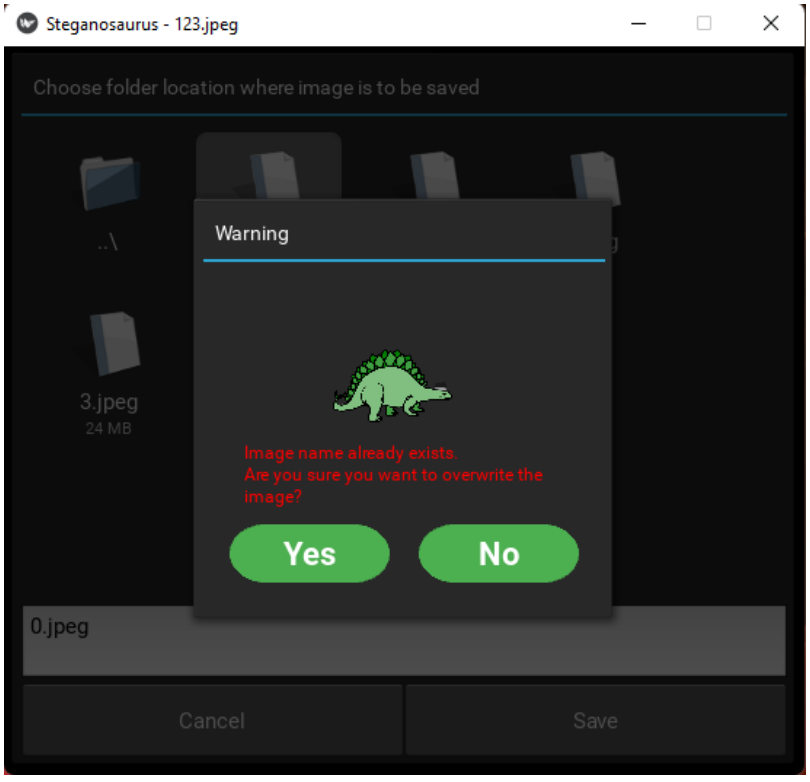


Windows

Figure 13. Successfully saving an image as JPEG (Test case 4-15).

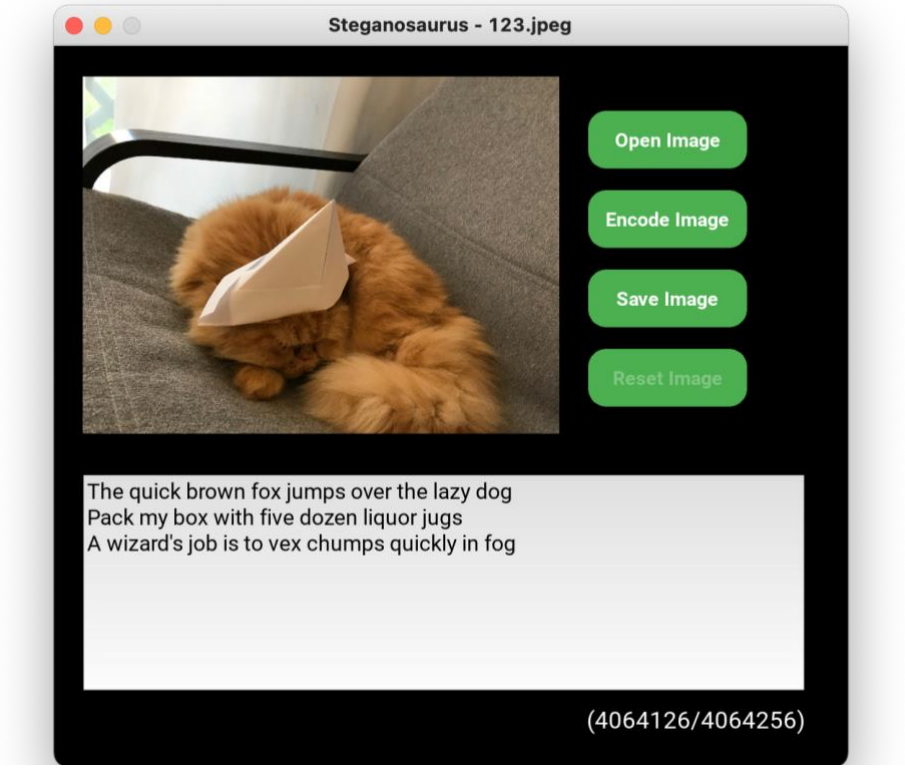


macOS

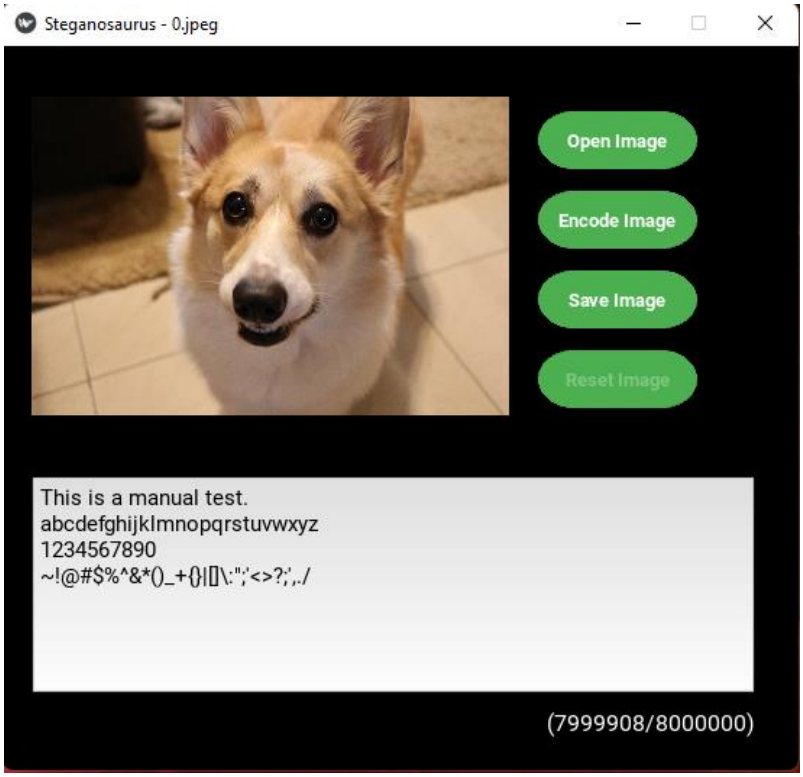


Windows

Figure 14. Attempting overwrite triggers warning (Test case 4-17).

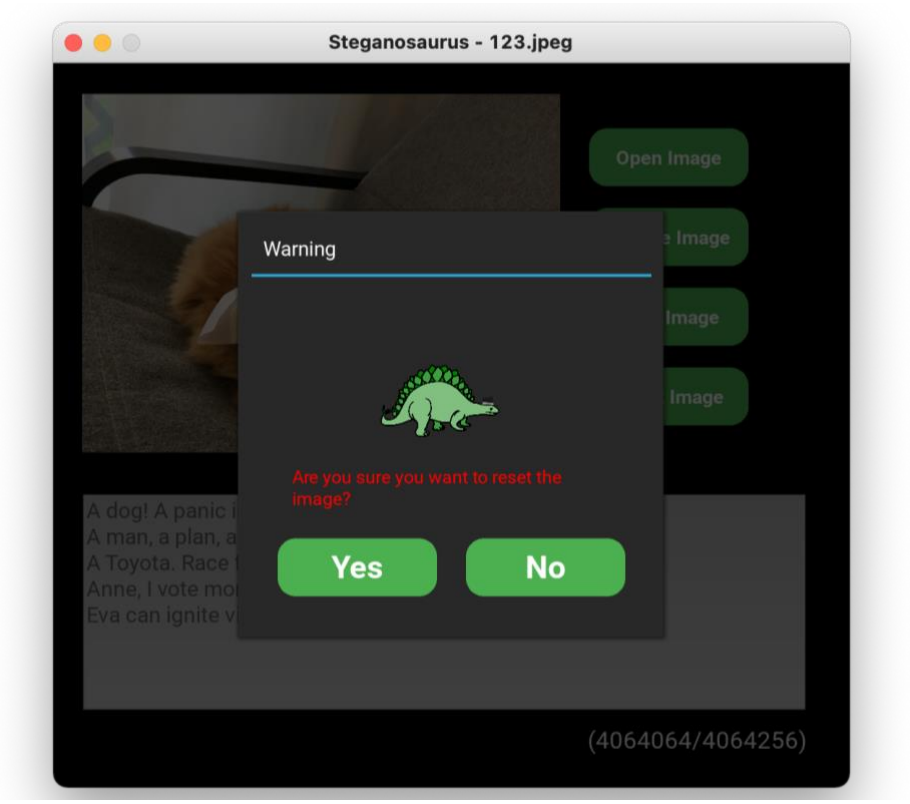


macOS

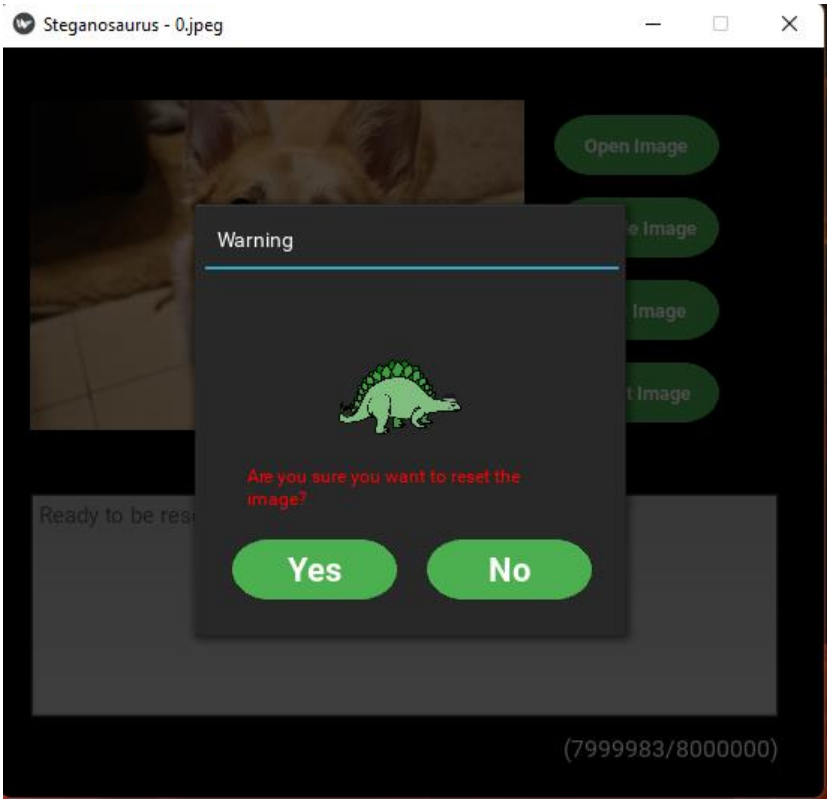


Windows

Figure 15. The files have been overwritten (Test case 4-17).

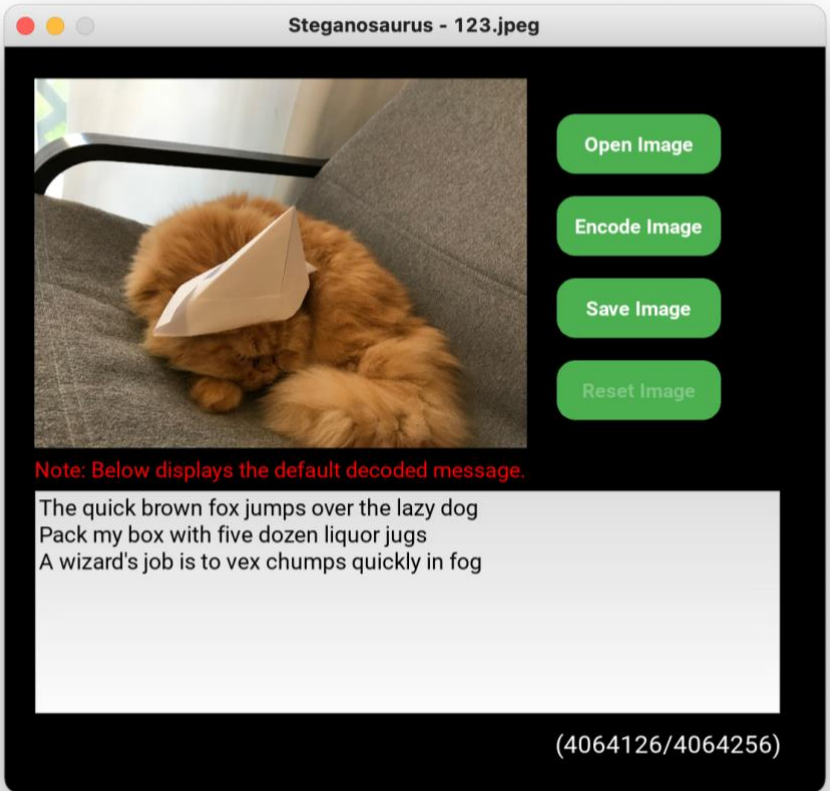


macOS

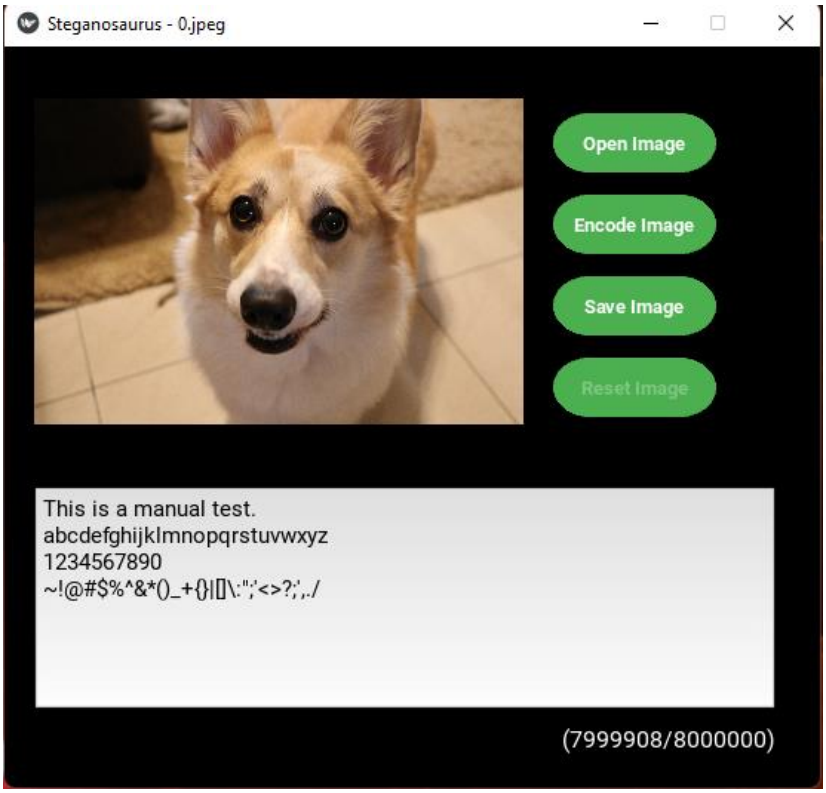


Windows

Figure 16. Reset button triggers warning (Test case 4-18).

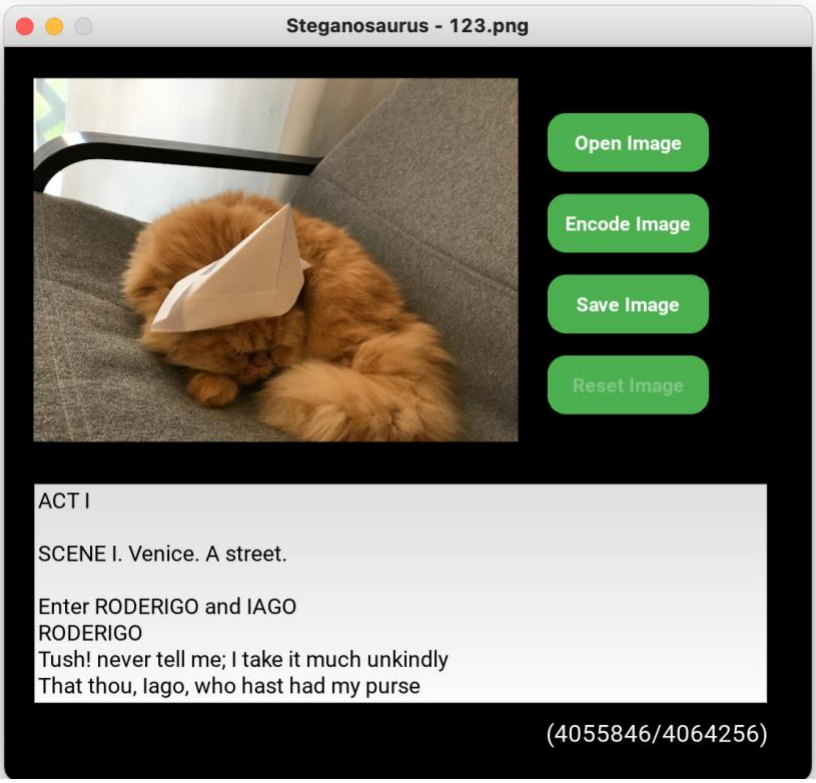


macOS

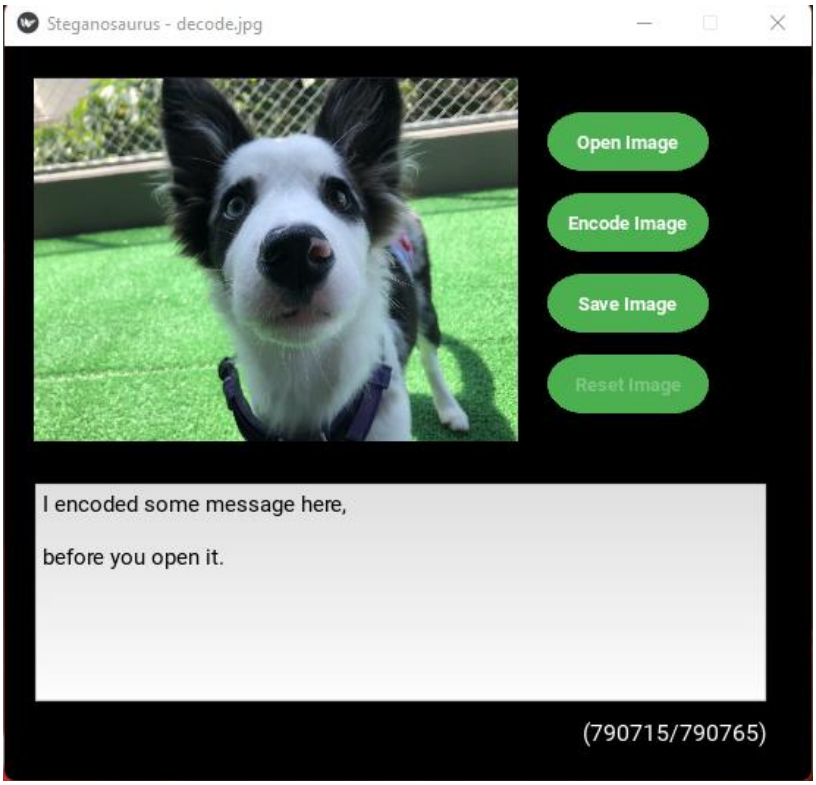


Windows

Figure 17. Image resets (Test case 4-18).



macOS



Windows

Figure 18. Image is decoded when opened (Test case 4-20).