Testing The Solution:

Testing Process Overview

The testing process is a critical phase in the software development lifecycle that ensures the software functions as intended and meets all the specified requirements. This section outlines the various stages of testing carried out to validate the software's functionality, performance, and usability.

1) Unit/Module Testing - this form of testing involves focusing on the individual components of the software in order to ensure that it functions as intended. I conducted module testing on the website's profanity filter, sentiment analysis system and all of the forms/inputs as well as their connections to the database.
2) Program Testing - involved running the program as a cohesive unit to make sure everything worked together. This stage also involved testing edge cases where there was a higher probability of throwing an error.
3) System Testing - this form of testing involves running the whole program on various systems in order to determine compatibility with operating systems, device specifications as well as usability across various devices. Valuable user feedback was gained during this stage, allowing me to adjust the user experience to accommodate all users.

Timeline of the Testing Process

Continuous (throughout entire development process)

- Module Testing (as new modules are added)
- Program Testing (ensuring new modules work together)

Term 4 Week 3

- Module Testing
    - Profanity Filter
- Implementing fixes

Term 4 Week 4

- Module Testing
    - Sentiment Analysis system
- Implementing fixes
- Comparison with initial specifications

Term 4 Week 5

- System testing
    - Computers/Operating systems
    - User experience
- Program testing
- Comparison with initial specifications

Term 4 Week 6

- System testing
    - Mobile/Tablet devices

Unit/Module Testing

Module testing is a form of software testing that focuses on evaluating individual units or components of the software. Its objective is to isolate a specific segment of code to ensure its accuracy and functionality.

Testing of the profanity filter

This module is intended to prevent all swear words from making it past form submissions and onto my website. In testing this section I chose to add the word “Skibidi” to the blocklist so that my documentation is not filled with swear words.

I tested the filter using the following method, found in tests.py:

```
testData = ["Skibidi Skibidi Skibidi", "sKiBiDi", ...]
for test in testData:
    print(check_profanity(test))
```

Test	Result	Comment
Skibidi Skibidi Skibidi	blocked	As expected
sKiBiDi	blocked	As expected. Lower() used to prevent these cases getting through.
s.k.i BIDI	blocked	As expected. Punctuation and spaces are removed to prevent these cases getting through
BaulkhamskibidiHills	blocked	As expected

Sk1b1di	blocked	As expected. Numbers are replaced with corresponding letters to prevent these cases getting through
I love sk?ibidi	not blocked	Did not consider punctuation other than “.” And “,”. **Fix:** cycle through all punctuation and remove each individually. `punct = ['!', '#', '"', '%', '$', '&', ')', '(', '+', '*', '-', '@', '^', '_', '`', '{', '}', '[', ']', '|', '~', ':', ';', '<', '>', '=', '?', ',', '.', '/']`
s k i. b i di	blocked	As expected
WhAt tHeSki Bidi???	blocked	As expected
sssskiiiiiiibbideee	not blocked	As expected. Unfortunately there is no simple way to check for letter repeats or letters that sound the same since the input is compared to a list of existing swear word strings. Minor compromise to the accuracy of the filter but unavoidable. Otherwise I would need a complex LLM to find these cases.
S(*& KIB^&*ID. %^ &I	blocked	As expected. Testing an extreme case.
Skib Id I	not blocked	Did not consider swear words written over multiple lines. **Fix:** split the input into its separate lines and then concatenated them `for line in textInput.splitlines():`

I’ve found that since I’m taking quite a strict approach to blocking profanity, there are occasionally cases where it's triggered accidentally. For example, the use of “… but the …” during testing has influenced me to remove “butt” from the profanity filter. Unfortunately there is nothing that can be done to completely solve this issue but I think I have managed to strike a good balance between blocking profane words and allowing the user to express whatever is on their minds.

Testing of sentiment analysis model

The sentiment analysis function is responsible for taking in a post as its input and assigning it a value from 1-5 based on how positive it is, where 5 is very positive and 1 is very negative.

I tested the model using the following code in tests.py:

```
testData = ["I love Glowup!!", "I hate this website!", ...]
for test in testData:
    print(sentiment_score(test))
```

Input	Case	Result	Comment
I love Glowup!!		5	As expected
I hate this website!		1	As expected
I'm having an okay day		3	As expected
I wish I was cooler		3	As expected
I can't wait for school to end!		5	As expected
I'm having the time of my life.		5	As expected
I'm walking on sunshine	Figurative language	5	As expected
I feel like the cat that got the cream	Figurative language	4	As expected
Great, now my phone's broken too.	Sarcasm	3	As expected
I think racism is great!	Controversial statement	5	Unfortunately the system only classifies positivity, not the actual content of the input. Would need to use a proper LLM to weed out all these cases but I have added a list of controversial words to

			the blocklist to minimise this occurring.
I'm drowning with happiness	Positive sentiment expressed negatively	5	As expected
The movie wasn't bad, I guess.	Double negative	3	As expected
The food was fantastic, but the service was terrible.	Mixed feelings	2	Slightly more negative than expected.
That was a bummer :/	Emoticon	1	As expected
OMG this is lit!	Slang	5	As expected

The program does an exceptional job at ranking almost every input it was given. It is surprisingly accurate even when phrases contain slang, figurative language, sarcasm, or other obscure language features. The only downside is that it cannot understand the content of the input semantically so some controversial statements may be ranked higher than they would otherwise. I have added sensitive words to the blocklist to help prevent this.

Testing of Forms, Inputs & Database

In carrying out the database tests I used the online postgres database hosted on railway.app so that in addition to database functionality I could also observe the server response times. The architecture of the local postgres database is identical to that of the online one though so any unintended responses were common across both, and have been addressed accordingly.

Testing was conducted in accordance with the UI elements to ensure that there were no errors with the data collection. The relevant software versions included Python 3.12.0 and Django 4.2.6. The server specifications are as follows: 512 MB of RAM, 1 GB of Disk, and 2 vCPU.

Whilst my original plan was to host the entire project, Vercel has a size limit of 250mb which my project has greatly surpassed (The sentiment analysis model is 700mb alone (sorry Mr Dunne's hard drive)). As such all tests were conducted locally, with only the backend online.

Test	Data	Response	Comment
Create account	`[name: “testuser”, email: “testuser@gmail.com”, password: “asdf”, password2: “asdf”]`	Created account	As expected.
Create account with profanity	`[name: “testuser2”, email: “testuser2@gmail.com”, password: “asdf”, password2: “asdf”]`	Sent error message to user *“Username Contains Sensitive Words”*	As expected.
Create account with duplicate username	(user testuser already exists in database) `[name: “testuser”, email: “testuser2@gmail.com”, password: “asdf”, password2: “asdf”]`	Sent error message to user *“Account Taken”*	As expected.
Create account with duplicate email	(email asdf@gmail already exists in database) `[name: “testuser3”, email: “asdf@gmail.com”, password: “asdf”, password2: “asdf”]`	Sent error message to user *“Email Taken”*	As expected.
Create account with missing fields	`[name: “testuser3”, email: “”, password: “”, password2: “”]`	Account created	**Error.** Users shouldn’t be able to submit a form without filling in all their details.

Create account with missing fields	`[name: "testuser4", email: "", password: "", password2: ""]`	Sent error message to user *"Please fill in this field"*	Fixed by adding the 'required' attribute to the form inputs.
Passwords not matching	`[name: "testuser5", email: "testuser5@gmail.com", password: "hello", password2: "helo"]`	Sent error message to user *"Passwords not matching"*	As expected.
Log in	`[name: "testuser", password: "asdf"]`	Logged in	As expected.
Login with invalid credentials	(testuser's password is 'asdf') `[name: "testuser", password: "qwerty"]`	Sent error message to user *"Invalid Credentials"*	As expected.
Log in to non-existent account	`[name: "testuser100", password: "asdf"]`	Sent error message to user *"Invalid Credentials"*	As expected.
Update profile	`[image: "~/Desktop/img.png", bio: "Hello"]`	Profile updated	As expected.
Update profile with profanity	`[bio: "skibidi"]`	Sent error message to user *"Bio contains sensitive words. Try again."*	As expected.
Create post	`[textInput: "hi"]`	Post created	As expected.
Create post	`[textInput: "skibidi"]`	Comment not	As expected.

with profanity		posted.	
Send message	`[message: "hi"]`	Message created	As expected.
Send message with profanity	`[message: "skibidi"]`	Sent error message to user *"We've detected some sensitive words in your message. Please try again."*	As expected.

Server response times using the online database were typically between 3-4 seconds as indicated by the Chrome Web Tools network tab. This is slower than I would like but unfortunately I can only use the free plan of railway.app to deploy my server so there is nothing I can do about these times.

Running the code on the local database is significantly faster, with no noticeable loading times. The chrome tools network tab records a consistent 1.1-1.2 second for all resources to be loaded in. I have connected this database for the final submission.

Program Testing

Program testing is a type of software testing that involves evaluating the complete and integrated software program to ensure it meets the specified requirements and functions correctly. The goal is to identify any defects or issues within the entire program.

Program testing was carried out by testing each aspect of the website in succession in order to determine the overall functionality. It involved running the program from the terminal in Visual Studio Code. Unless otherwise specified, testing was carried out using the default username and password ('matej', 'asdf'). Furthermore, as the purpose of program testing is to isolate errors in the code, I included additional tests with higher probabilities of throwing an error.

It is also worth noting that program testing was conducted throughout the development of the entire program as individual modules were added. My logbook entries provide a detailed recount of issues that arose during this process.

Test	Methodology	Result	Comment
Site does not allow navigation to pages when not signed in	Attempt to navigate to subpages whilst no user is signed in. Eg. *127.0.0.1:8000/post*	Redirected to sign in page. Except for the *info* page which I allow everyone to access regardless of if they have an account	As expected.
User can create an account	Clean install website, Sign up to a new account	User account created	As expected.
User can sign in	Log in to an existing account	User signed in	As expected.
User can update profile image and bio	Log in to existing account, Navigate to settings page, Input image and bio and submit	Profile updated	As expected.
Website fetches posts correctly	Log in to existing account, Navigate to the home page	Posts displayed correctly, in order of positivity	As expected.
User can like a post	Log in to existing account, Navigate to the home page, Like a post	Post liked	As expected.
User can comment on a post	Log in to existing account, Navigate to the home page, comment on a post	Comment created correctly	As expected.
User can create a post	Log in to existing account, Navigate to the post page, create a post	Post created successfully	As expected.

User can complete a kindness challenge	Log in to existing account, Navigate to the kindness challenge page, Press complete button	Successfully completed	As expected.
User can increase their kindness streak each day	Log in to existing account, Navigate to the kindness challenge page, Press complete button Repeat the next day	Current streak updated successfully	As expected.
User's kindness streak resets if missed	Log in to existing account, Navigate to the kindness challenge page, Press complete button Check 2 days later	Streak reset	As expected.
User can send kindness message	Log in to existing account, Navigate to the kindness message page, Send message	Message sent correctly	As expected.
User can see received kindness messages	Log in to existing account with kindness messages sent to it already, Navigate to the message page	Messages displayed successfully	As expected.
User can view their liked posts	Sign up to a new account, Like three existing posts, Navigate to liked posts page	All liked posts displayed	As expected.

User can view their profile	Log in to existing account, Navigate to profile page	Profile page displayed correctly. Recognises that it is another user	As expected.
User can view another user's profile	Log in to existing account, Click on another user	Profile page displayed correctly. Recognises that it is another user	As expected.
User data not exposed when next user signs in	Log in to existing account, Log out of existing account, Log in to another existing account	All user data reset	As expected.

Program testing indicates that all features work as expected. Thus, no fixes are required.

System Testing

System testing is a comprehensive form of testing where the complete system is evaluated as a whole. The aim is to verify that the integrated system functions as intended and meets the specified requirements by testing the interactions between various components and the overall system behaviour.

Through using system tests I hope to address these main areas: does my system function correctly on all devices, is the performance of my system consistent across all devices, are there any bugs or issues within my system, and is my system intuitive and easy to use.

System	**Issues**	**Feedback**
Tester: Matej Groombridge **System:** Macbook Air M1 chip 8GB memory 13.3-inch (2560 × 1600)	None	Ran 100% as intended on my device without any issues. Good performance. [don't feel qualified giving feedback as its my own project]

Mac Os Sonoma 14.4.1 **Browser:** Brave		
Tester: Risith Senaratne **System:** Acer Nitro AN515-57 11th gen intel(r) core(TM) i7-11300-H @ 2.30GHz 16GB RAM (15.8 Usable) 64 bit operating system Windows 11 Home **Browser:** Chrome	Time displayed incorrectly (Off by 10 hours)	"I love the website" "I'm not using any other social media from now on" Intuitive flow - similar to other social media platforms Became noticeably happier after use Disappointed with lack of users on the platform → Due to limited testing scenario
Tester: Manav Jain **System:** Macbook Pro M1 chip 16GB memory 14-Inch 3024 x 1964px Mac Os Sonoma 14.5 **Browser:** Chrome	None	Experience was good No visible errors or bugs Intuitive user interface "Instagram should watch their backs"
Tester: Grace Sung **System:** Macbook Air M1 chip 8GB memory 13.3-inch (2560 × 1600) Mac Os Sonoma 14.4.1 **Browser:** Safari	None	Very positive experience "I feel as if I've been a kinder person using your website" Design is very aesthetically pleasing Runs smoothly with no glitches Very user friendly Icons are very easy to

		understand Made a new friend (Grace is not very technologically oriented so this was exceptional feedback)
Tester: Matej Groombridge **System:** Samsung Galaxy A20 720 x 1560 pixels, 19.5:9 ratio (~268 ppi density) Android 11 Exynos 7884 (14 nm) Chip 2x1.6 GHz Cortex-A73 & 6x1.35 GHz Cortex-A53 CPU 32GB Storage, 3GB RAM **Browser:** Google	None	Works incredibly well on mobile All pages displayed a mobile view correctly Despite being an extremely low end device there were no issues with performance whatsoever Note: the site was hosted on my laptop and exposed to the local network so the phone didn't have to do a lot of the backend work. This is how it would perform if deployed in real life.
Tester: Everyone Hosted Glowup and the local database on the school wifi network so everyone joined the site at the same time during our software lesson and tested it out. Also got more friends to sign up throughout the day and use the platform. Obviously there was a large range of devices and browser specifications being used.	No issues	Very positive feedback from everyone. It was great to see Glowup being used by many people at the same time as well. Lots of happiness too. Big success.

System Testing Feedback Implementation:

- Made all datetime.now() functions timezone aware so that times are displayed correctly across all devices

Comparison with initial design specifications

The key design specification for this website was that users must be able to undertake all activities associated with a social media site. After undertaking testing, gaining feedback, and reflecting on this project I have come to the following conclusions regarding the project's initial specifications. I have provided the design specifications verbatim from my proposal and included a detailed reflection on their achievement which includes a status and any areas of change or improvement.

Specification	**Result**
Users must be able to create and view posts tailored to them.	**Status:** Achieved Glowup provides an easy-to-use post system where a user can input their desired text. Users are encouraged to be creative and express themselves authentically on the platform. As Glowup is a website for positivity, I have successfully implemented a sentiment analysis system that ranks posts in a user's feed based on how positive it is deemed by a language model. Thus, posts are tailored to Glowup's users. Furthermore, I chose to implement a liked posts page so that a user can even better curate the posts that resonate with them. Aside from profane language, I was able to achieve this design specification without banning posts. Thus in addition to providing a curated feed I was able to uphold the human right to free speech which I think is always a benefit to social media websites as censorship leads to a lot of ethical and governance issues.

Users must be able to communicate with other users through messages and post comments. As this social media platform aims to spread love and positivity, all text inputted by users must be screened for offensive or inappropriate content before being published.	**Status:** Achieved Messages → I built out a robust messaging system which allows users to send kindness messages to other users of the platform. Whilst initially planning to use an API (stream-chat), I found that with the skills I've learnt from building out the database I was able to achieve this specification without needing this external software. Although more simple of a system than I originally anticipated, I think it is very intuitive to use and contributes uniquely to Glowup's broader mission of promoting kindness between individuals by encouraging them to send specifically kindness messages. Comments → Users are able to share their thoughts on any post they come across. This promotes greater expression and builds community on the platform. In building out these systems I ensured to put positive connections at the forefront. This was achieved through rigorously blocking all forms of profanity as well as promoting kindness and positivity between users as they use these features. By filtering out harmful content, Glowup can create a space where users feel safe and can focus on spreading love and positivity.
Users must be able to create a custom user profile that reflects their online identity.	**Status:** Achieved Glowup provides users with the ability to create a custom username and bio. Furthermore, users have the option to

	upload a profile photo, giving them a digital face on this platform. In giving users these opportunities I have successfully facilitated their self-expression. Users are capable of creating a unique profile that reflects their digital identity. Furthermore, I chose in the development of this project to include all of a user's posts on their profile. This feature allows them to further display their interests, ideas and ruminations leading to a more complete online identity.

Users must be able to interact with the website intuitively and all inputs must produce desired outputs. Glowup will have a clear, well-thought-out design meaning that the general population will be able to use it without any advanced technical knowledge. Whilst the target demographic is individuals aged between 15-30 years old, Glowup should be able to be used by people of all ages.	**Status:** Achieved Based on extensive testing and user feedback, Glowup provides an intuitive user experience. The GUI has been labelled “easy to use” by individuals with varying technical knowledge and follows the design conventions of major social media platforms such as Instagram and Twitter (𝕏). The IPO chart within my major project proposal stipulates a list of inputs and their required outputs across my website. I have remained consistent with these decisions and all inputs within my program produce the desired output. Changes I have implemented include user inputs now displaying error messages if the user does not submit data correctly and posts are outputted based on their positivity rather than sequentially based on the time they were posted.

All fonts must be readable by users of all ages. Icons and symbols must be intuitive and large enough to click with ease. Furthermore, images and multimedia should be stored efficiently such that users can access Glowup with minimal bandwidth usage. Any multimedia, such as profile images, must retain their quality and be displayed at a size that can be easily viewed by the users.	**Status:** Achieved Glowup uses the easy-to-read Sans Serif fonts “Poppins,” and “Montserrat.” In order to ensure users of all ages can read the text, I have used a suitably large font size. Furthermore, icons on my website are large and intuitive to use, catering to all users. All of my website’s icons are stored in the svg file format. This is incredibly resource efficient and contributes to fast page speeds without losing any quality. User profile images are easily viewable and stored in the cloud. As such, they contribute to greater bandwidth usage, however as they are the only images on my site this does not contribute to significantly slower page speeds.
Inclusivity in design will extend to accessibility features, allowing users with diverse abilities to navigate the platform seamlessly. This includes providing alternative text for images and ensuring compatibility with screen readers to accommodate users with specific preferences or accessibility needs.	**Status:** Achieved In developing this website I put a lot of intentional thought into the design, creating it in such a way that no user is disadvantaged from using Glowup. As stated previously, all text and icons are of a suitably large size for those with poorer eyesight. In addition, every <img> section includes an alternate text tag (such as *alt*="profile image") so that screen readers are able to pick up on what is being displayed on the page and relate this to disabled users. The website has been tested with peers of varying technological capacity and experience, with positive feedback from these users revealing that Glowup is in

	fact intuitive to use for individuals of all experience levels. Furthermore, Glowup has been tested on low end and mobile devices, revealing that user's will not be hindered by the device that they use to access the platform.
Glowup should be able to be accessed across a wide range of devices. It should consistently work across all modern browsers, displaying correctly on phones, computers and tablets. The platform should be optimised to function effectively even on lower-end devices, ensuring that users with diverse technological capabilities can access and enjoy the site without any constraints.	**Status:** Achieved Glowup aligns with all website specifications set out by the W3C and has been proven to work over the internet with many people accessing it simultaneously. With monetary investment it could easily be pushed to the web permanently. In creating Glowup I took into account devices of all shapes and sizes, with dedicated mobile and tablet views ensuring that no users miss out or are disadvantaged by their device of choice. Glowup has been tested to work across the most common operating systems and maintains all of its functionality. Glowup has been tested on low end and mobile devices as well, revealing that user's will not be hindered by the device that they use to access the platform. Thus, Glowup works on any modern device and can be used all over the world without issue.

Minor additions to program not in specifications:

- Users are able to create a network of friends by following other users. Users are also now able to view a list of other users on the platform encouraging them to check out their posts and potentially connect with them too.

- Users can undertake daily kindness challenges which contribute to their “kindness streak”. This forms a sense of positive competition between users and forces them to get out of their comfort and make the world a better place.
- Users are provided with an information page welcoming them to the website, outlining Glowup’s vision and putting in place some guidelines for them to have the best user experience possible.
- Every input field on the website is now checked for profanity, including usernames, bios and post comments. Rigorous testing methods mean that no swear words whatsoever can make it through.

Conclusion

The testing process involved a thorough evaluation of individual modules, the complete program, and the integrated system. Each phase of testing helped identify and resolve issues, ensuring the software functions correctly and meets user expectations. The final product is a unique, sophisticated and engaging social media website that I’m proud of creating.