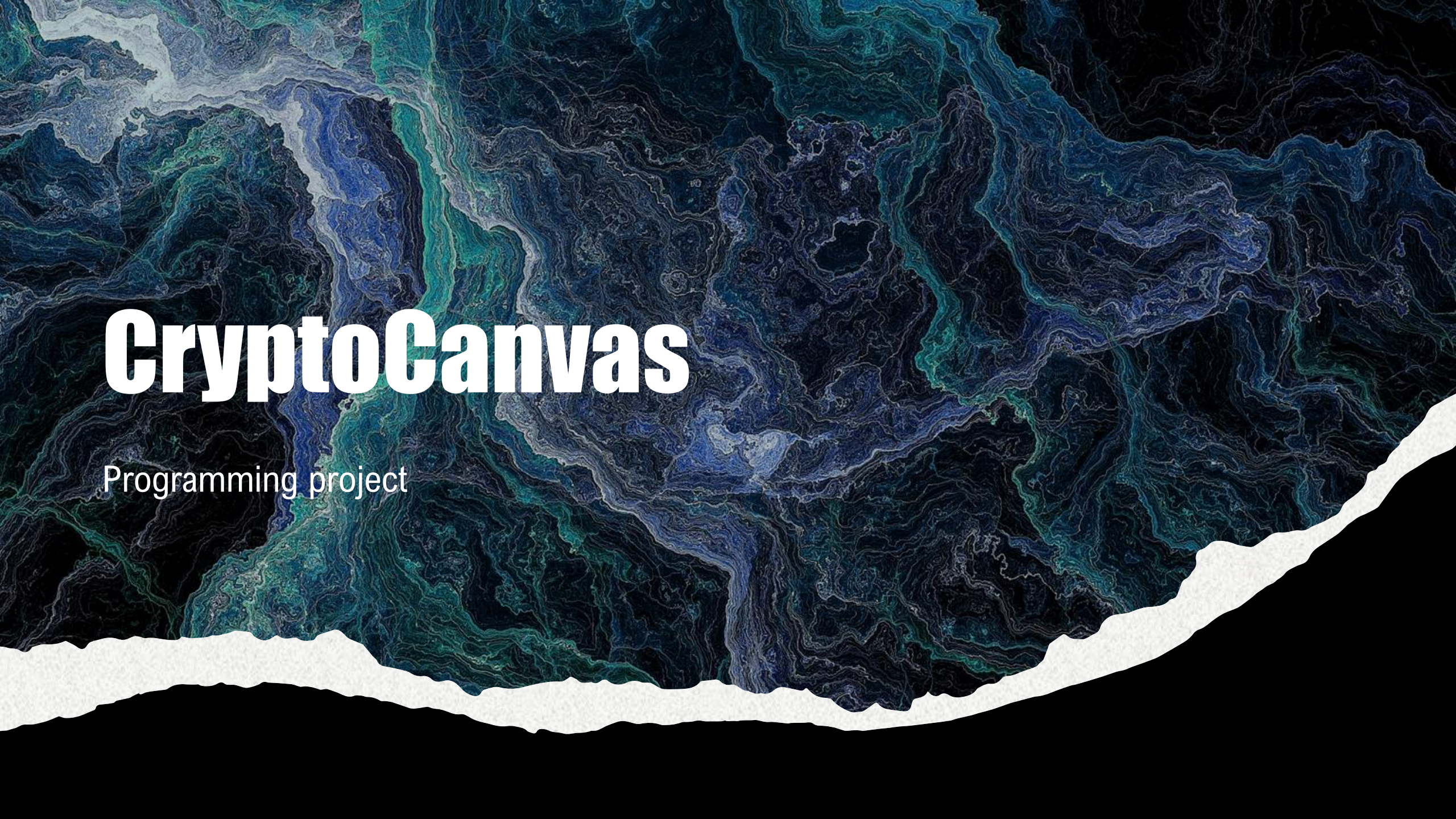


CryptoCanvas

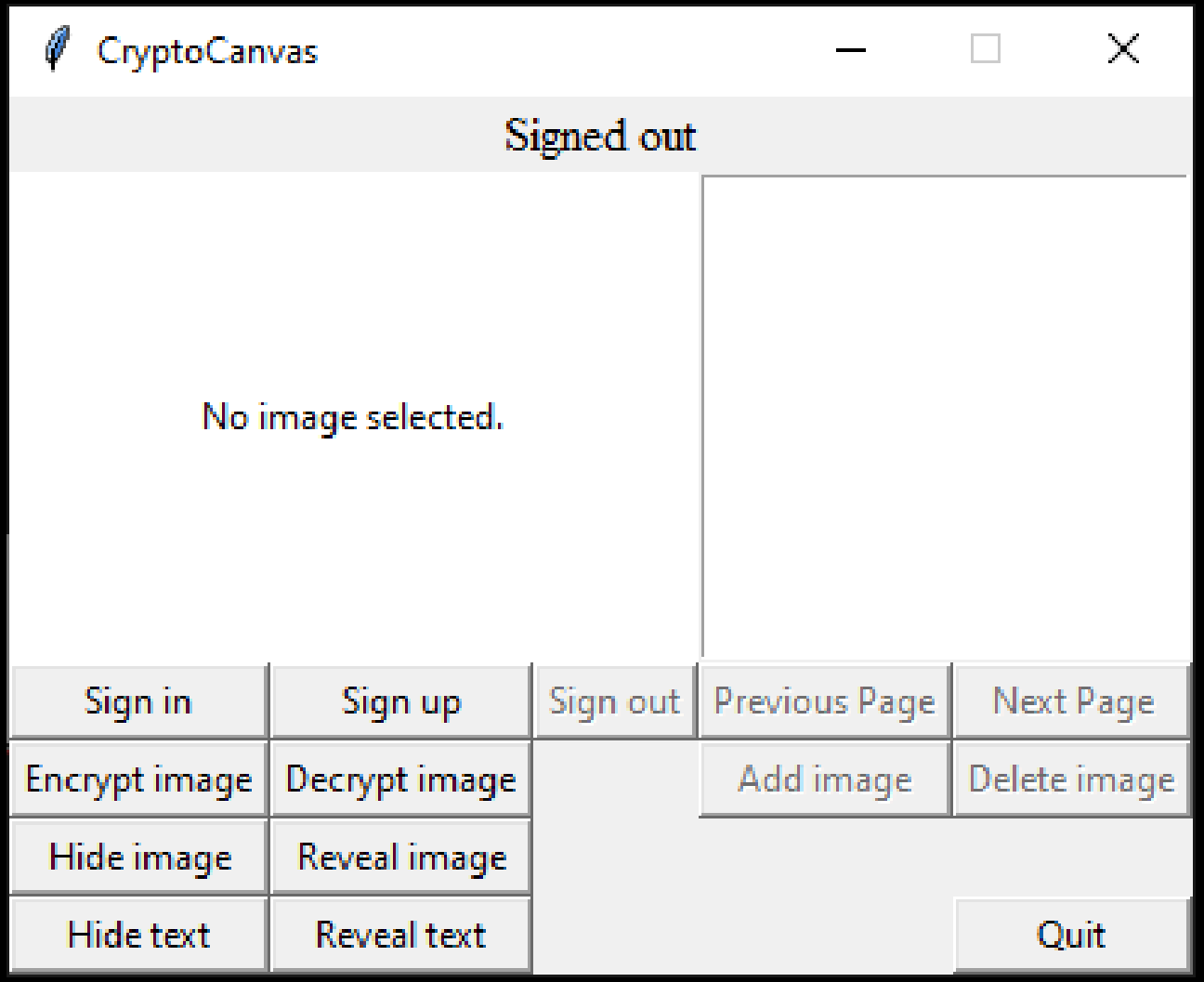
Programming project

What is it for?

- Image encryption/decryption
- Steganography
  (hiding data within images)
- Written in Python
- Tkinter GUI

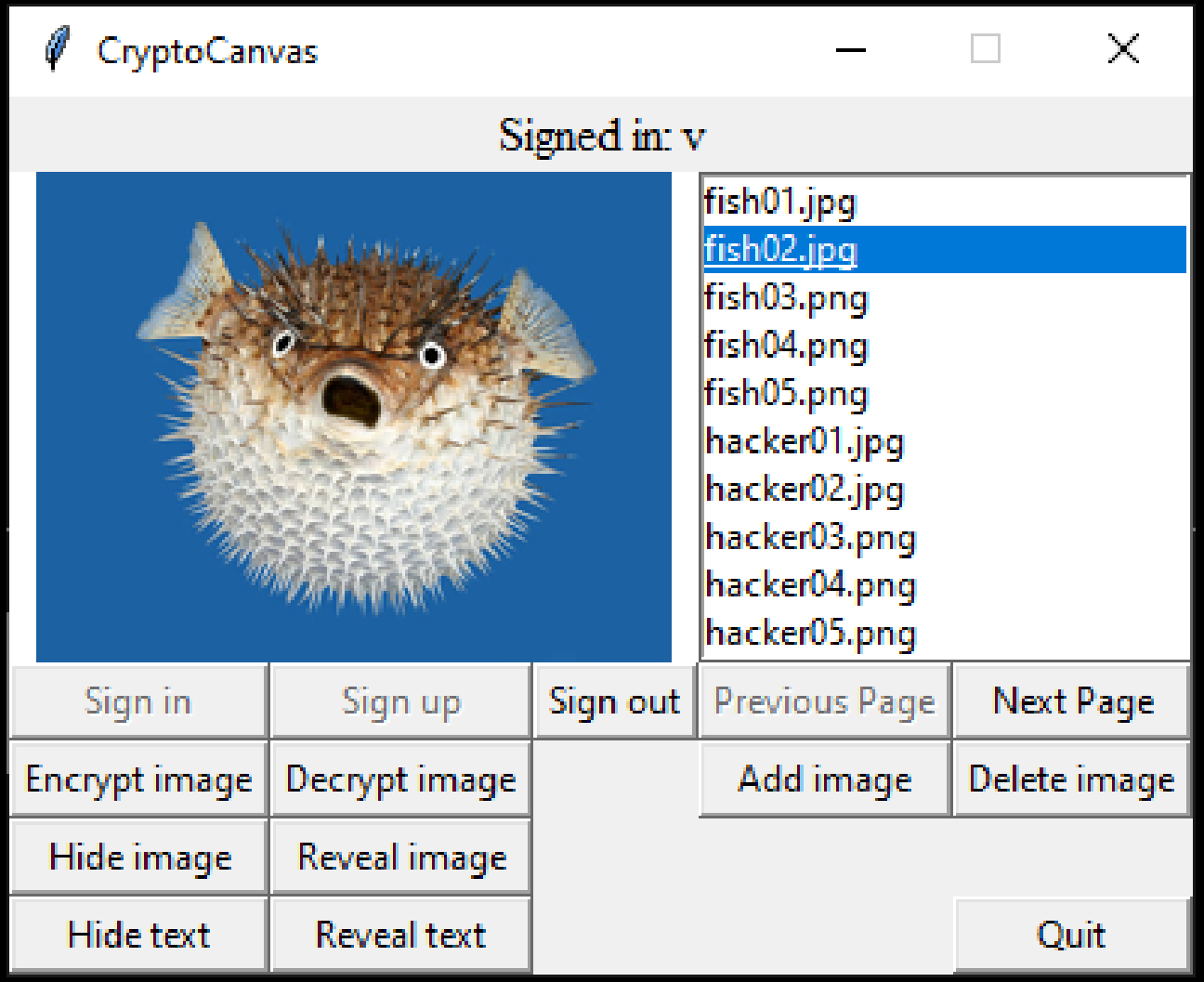
How does it work?

- Image operations are open for everyone
- Signing in grants access to the database feature
- Resulting files are saved on the user's device



How does it work?

- The user has signed in
- Database navigation and operations are available
- Database image preview
- Image operations can use images from database and device



Secure programming

- Encryption with AES-256-GCM algorithm
- LSB-steganography (Least Significant Bit)
- Argon2id KDF for derivation of encryption keys and hashing of user passwords
- AES-256-GCM and Argon2id both recommended by OWASP

Secure programming

- Exceptions are caught and errors are shown with message boxes
- Temporary files are created for database selections in some cases
- User emails are unique, regular expressions used for both email and password

Encryption

AES-GCM requires 3 things:

- Nonce or IV (Initialization Vector) generated as random bytes
- Encryption key (derived from password with Argon2id)
- Password (Given by the user, can be omitted)

Salt for key derivation is generated as random bytes

Nonce and salt are stored at the start and end of the ciphertext respectively

For decryption, only the password must be remembered

Hashing

Argon2 has 3 modes: Argon2i, Argon2d, and Argon2id

Argon2id is the balanced version between Argon2i and Argon2d

3 parameters that control execution time, required memory, and degree of parallelism

Parameters can be adjusted according to your needs, specific configurations are recommended by OWASP

LSB-steganography

Carrier image hides a secret message (Can be image or text)

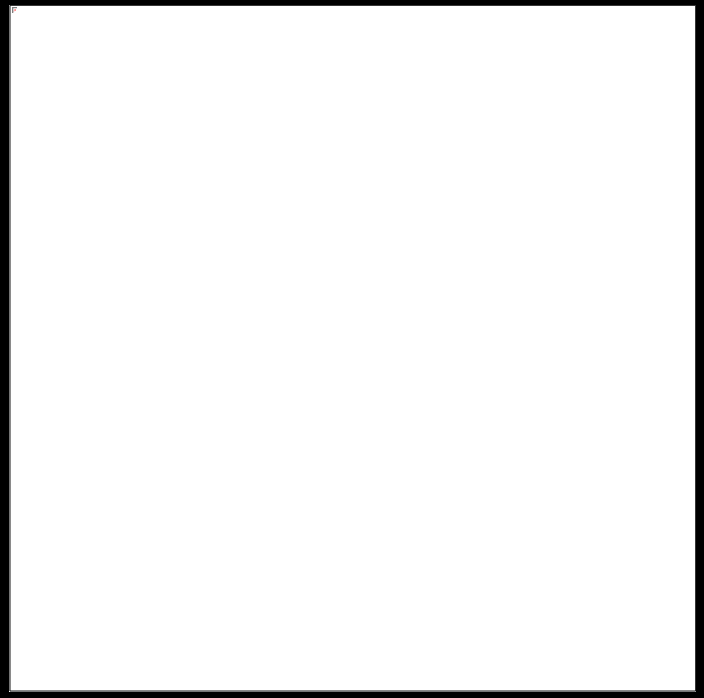
Data of the secret message is hidden in the least significant bits of the pixels of the carrier image

Difference cannot be seen by the human eye, but uncovered with steganalysis tools

Randomization in pixel selection helps obfuscate the data

Encryption demo

- Image on the right is encrypted (Cannot be displayed)



Steganography demo

- The image below has a secret message
- It says:
  “This is an example for the slideshow”

Testing

- Manual testing
- Use of libraries (AESGCM, argon2, stegano, PIL, sqlite3, Tkinter)
- File selection logic (Database or device? Image data or file path?)
- Uncaught exceptions (Image operation failures, cancelled operations)

Use of AI

- ChatGPT-3
  - Code generation
  - Troubleshooting and debugging errors
- Some takeaways:
  - Saves time
  - Sometimes generated code works,
    sometimes needs modifications,
    sometimes not at all what you wanted
  - Not helpful with complex issues or specialized tasks
  - Very helpful with syntax errors or wrong use of libraries

Thank you