

Developing a Fully Homomorphic Encryption WebAssembly module

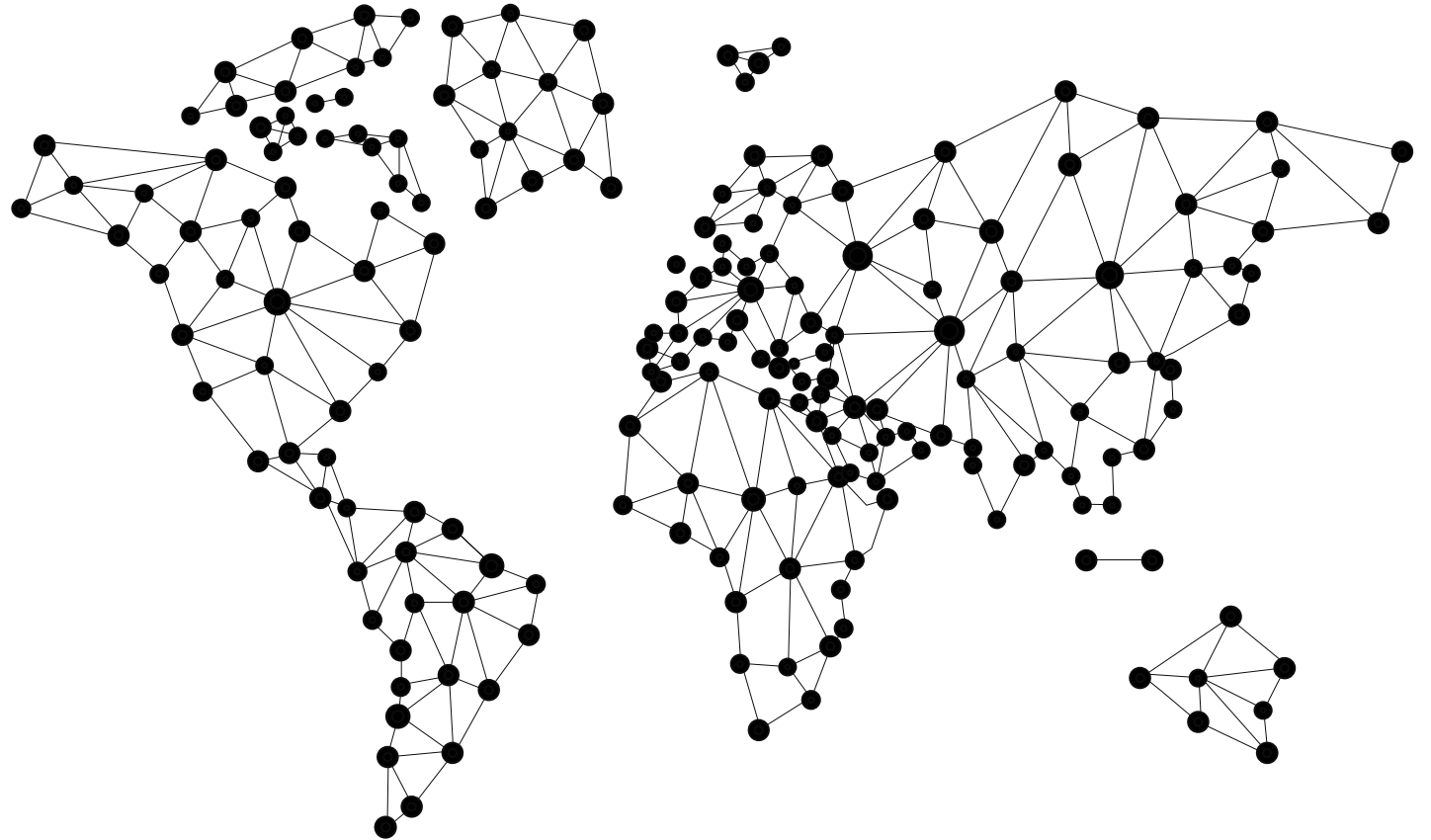


GRADUATE
PREDA MIHAIL IRINEL

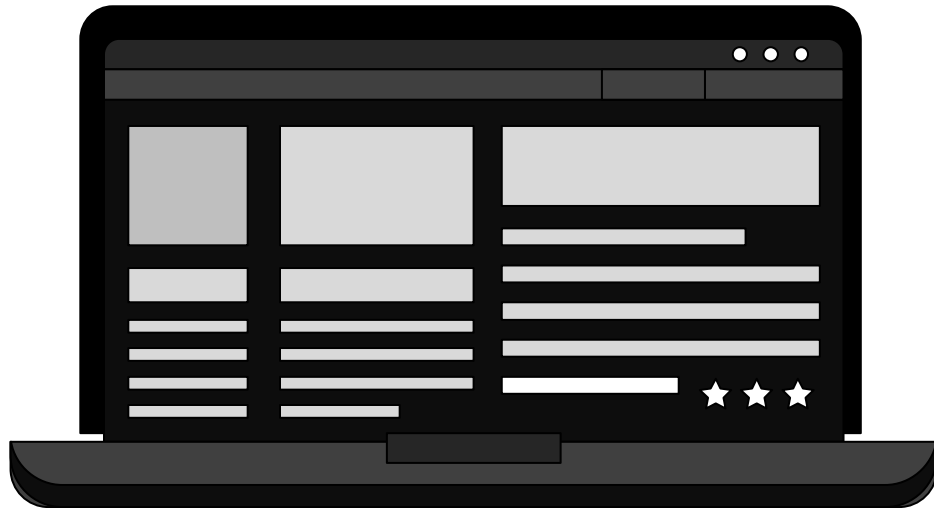
COORDINATOR
LECTURER PH.D. ANDREI TOMA

OBJECTIVE

**Creating a library
that could be easily
used on the web,
while making fully
homomorphic
encryption
concepts more
accessible.**



SUMMARY



01

INTRODUCTION

03

ARCHITECTURE

05

RESULTS

02

USED TECHNOLOGIES

04

DEMO

06

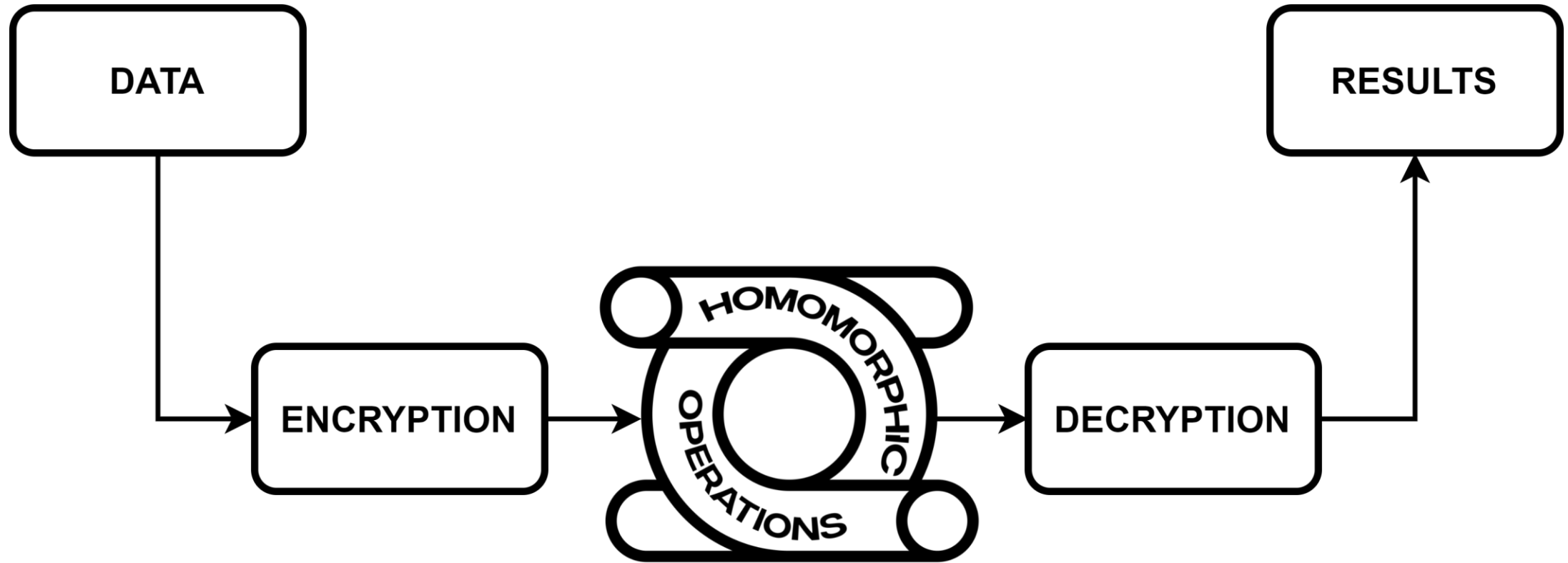
CONCLUSIONS

INTRODUCTION

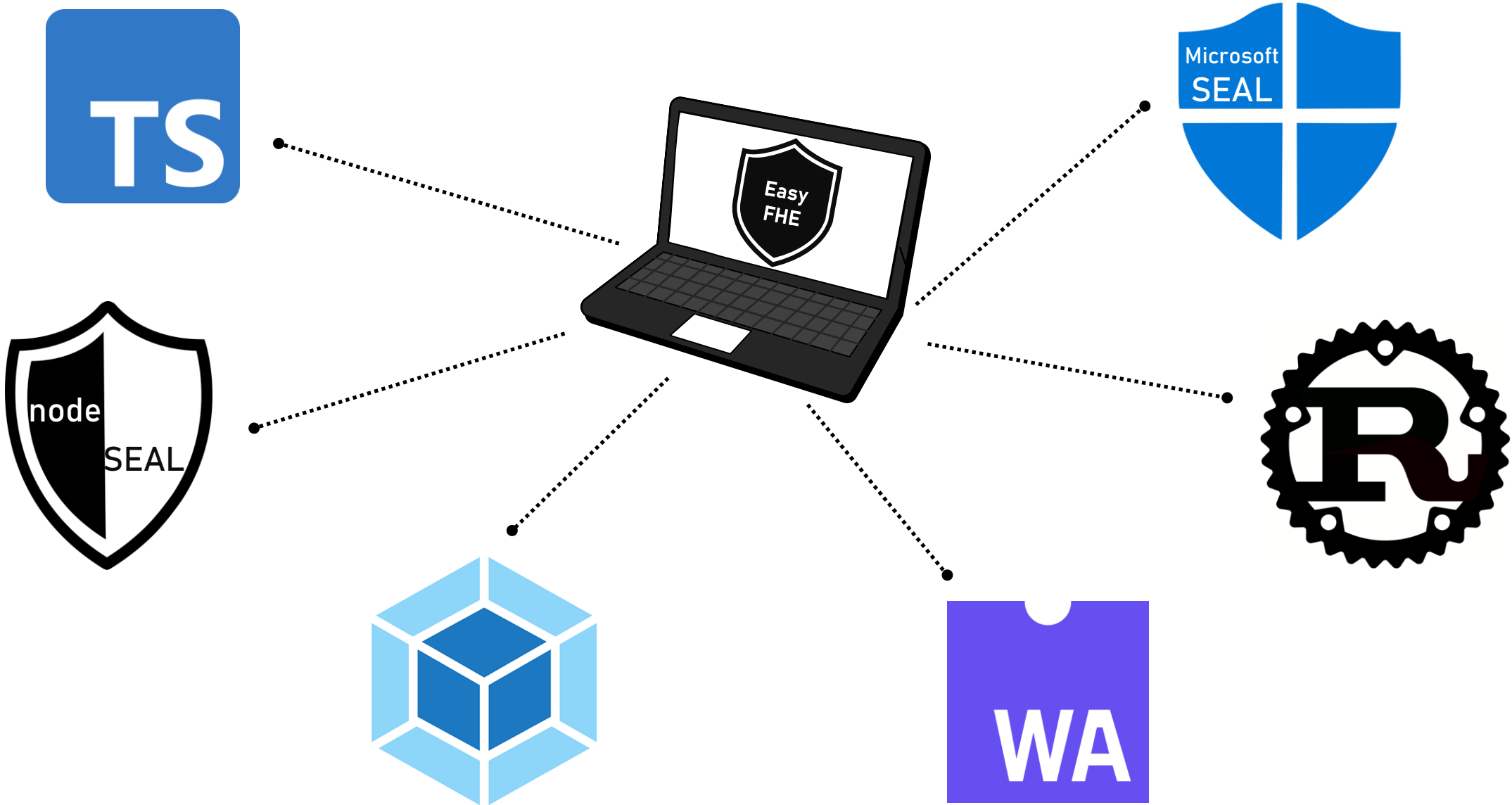
Every day, we create roughly 2.5 million terabytes. Around 9% is private information, meaning 225000 terabytes. The processing cost is immense. *Imagine if was something that could safely process all this data...*



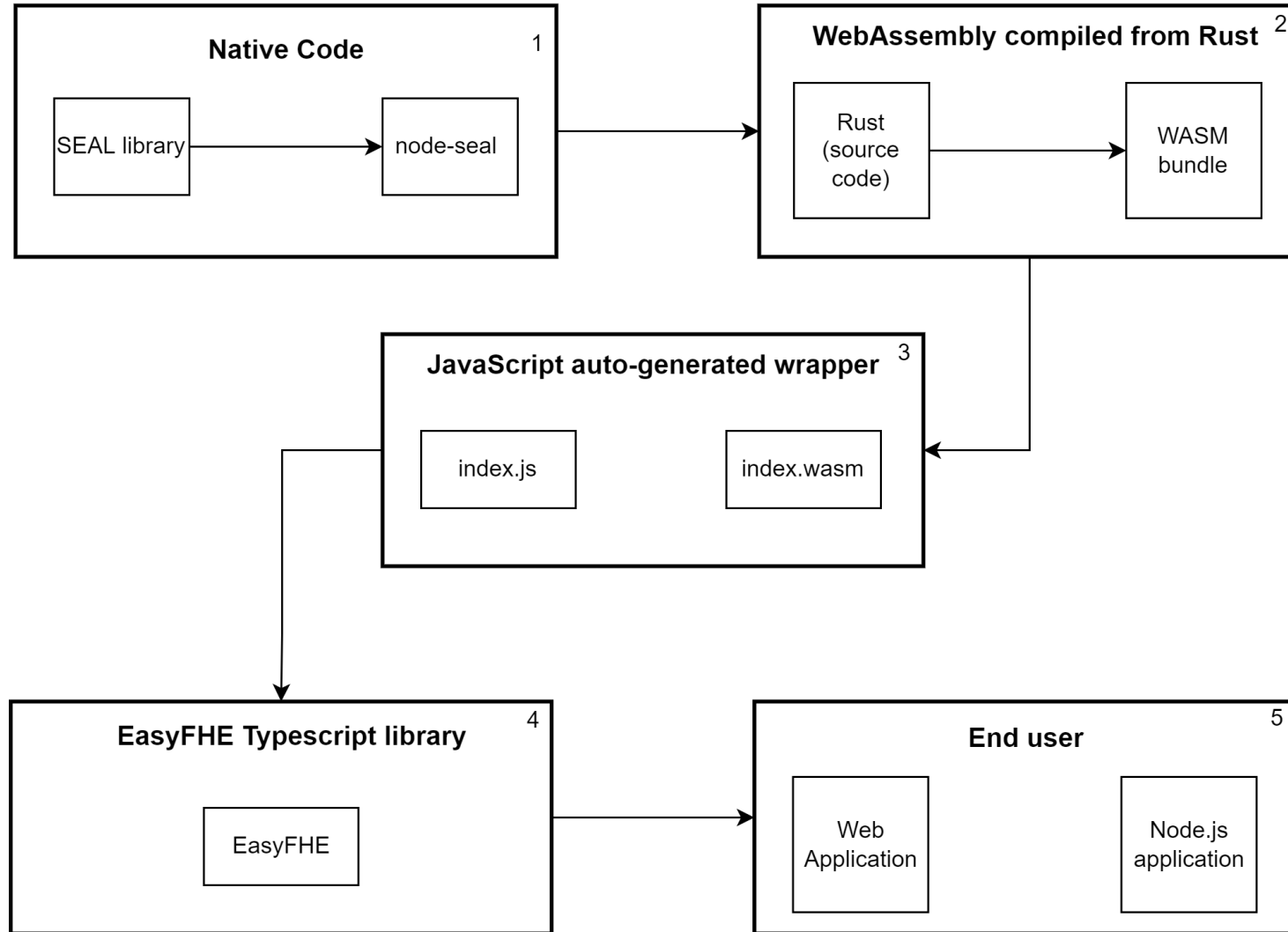
INTRODUCTION



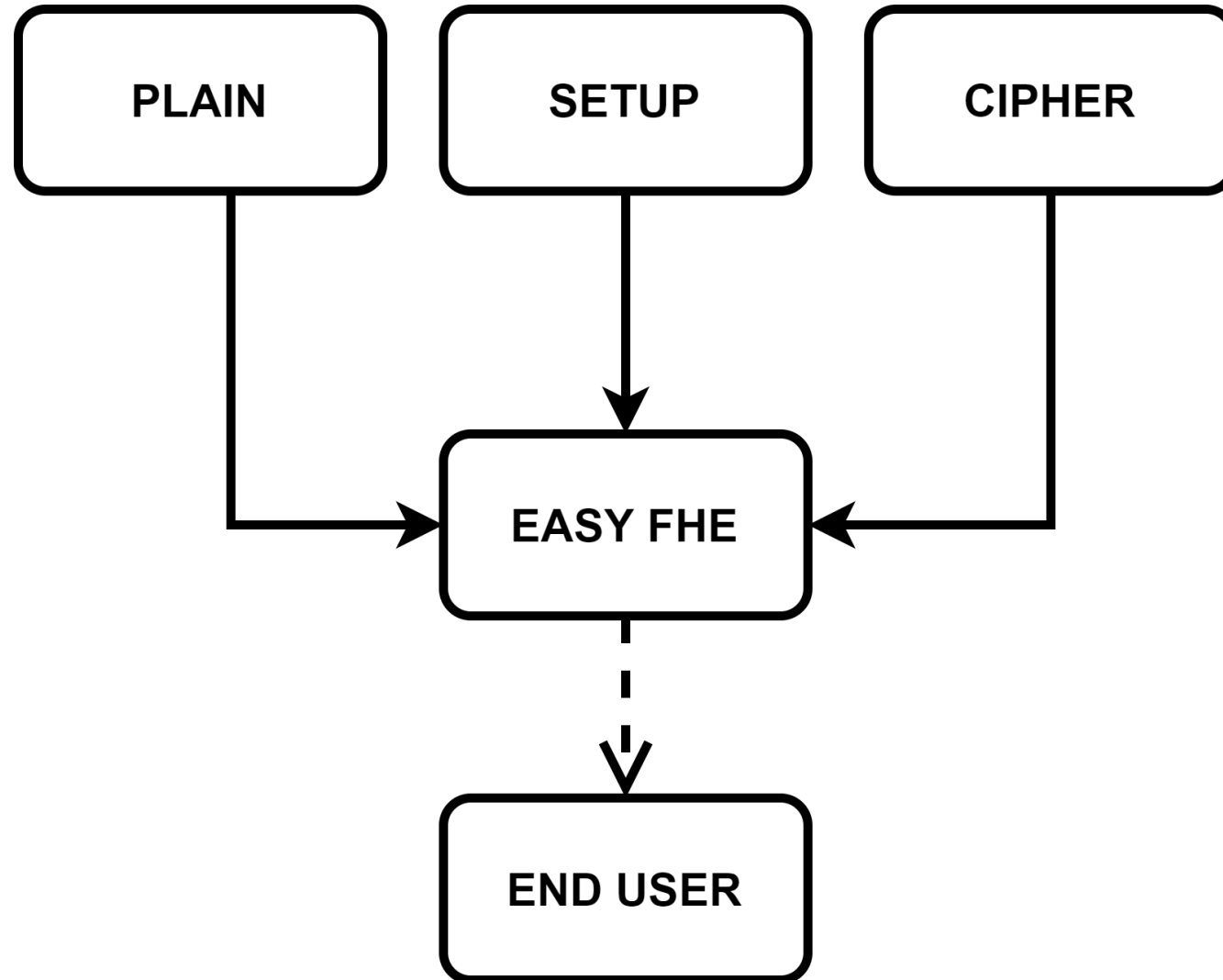
USED TECHNOLOGIES



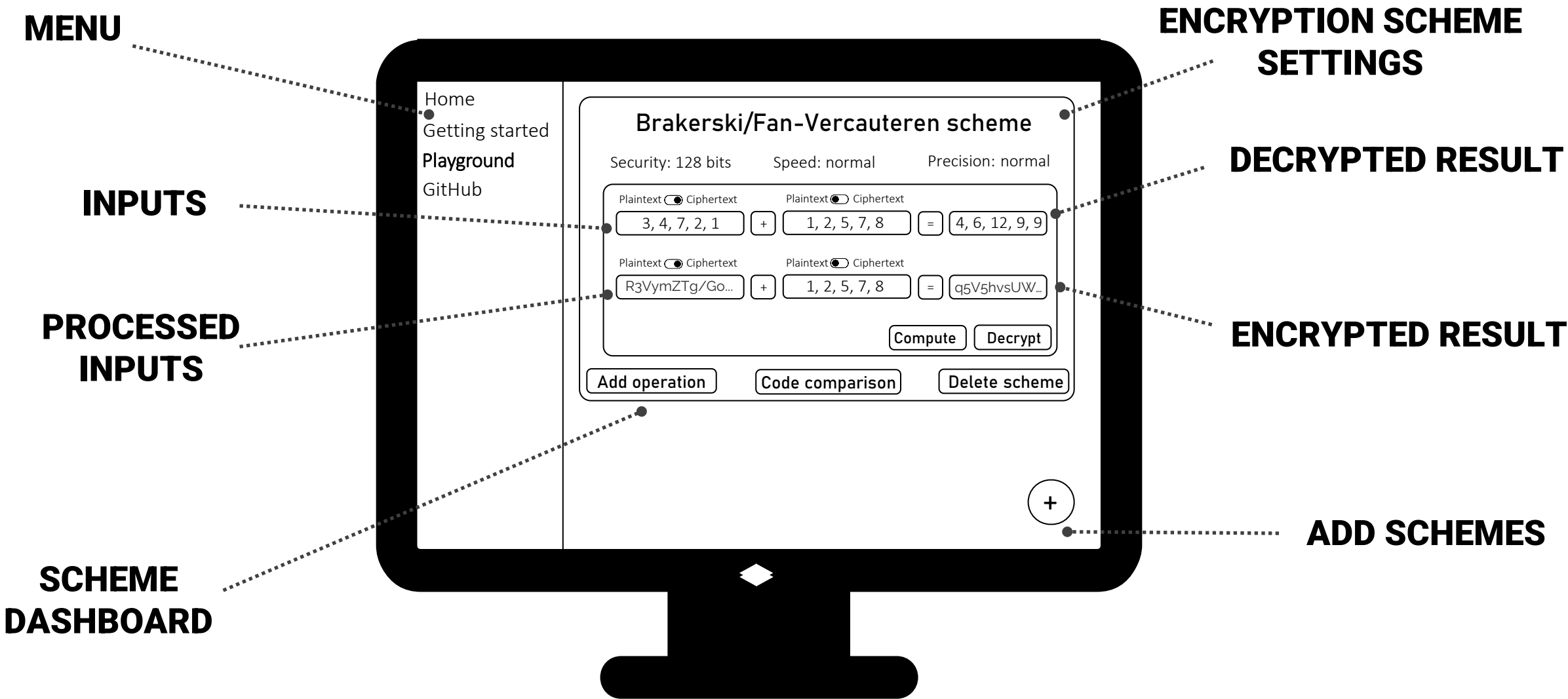
ARCHITECTURE



ARCHITECTURE

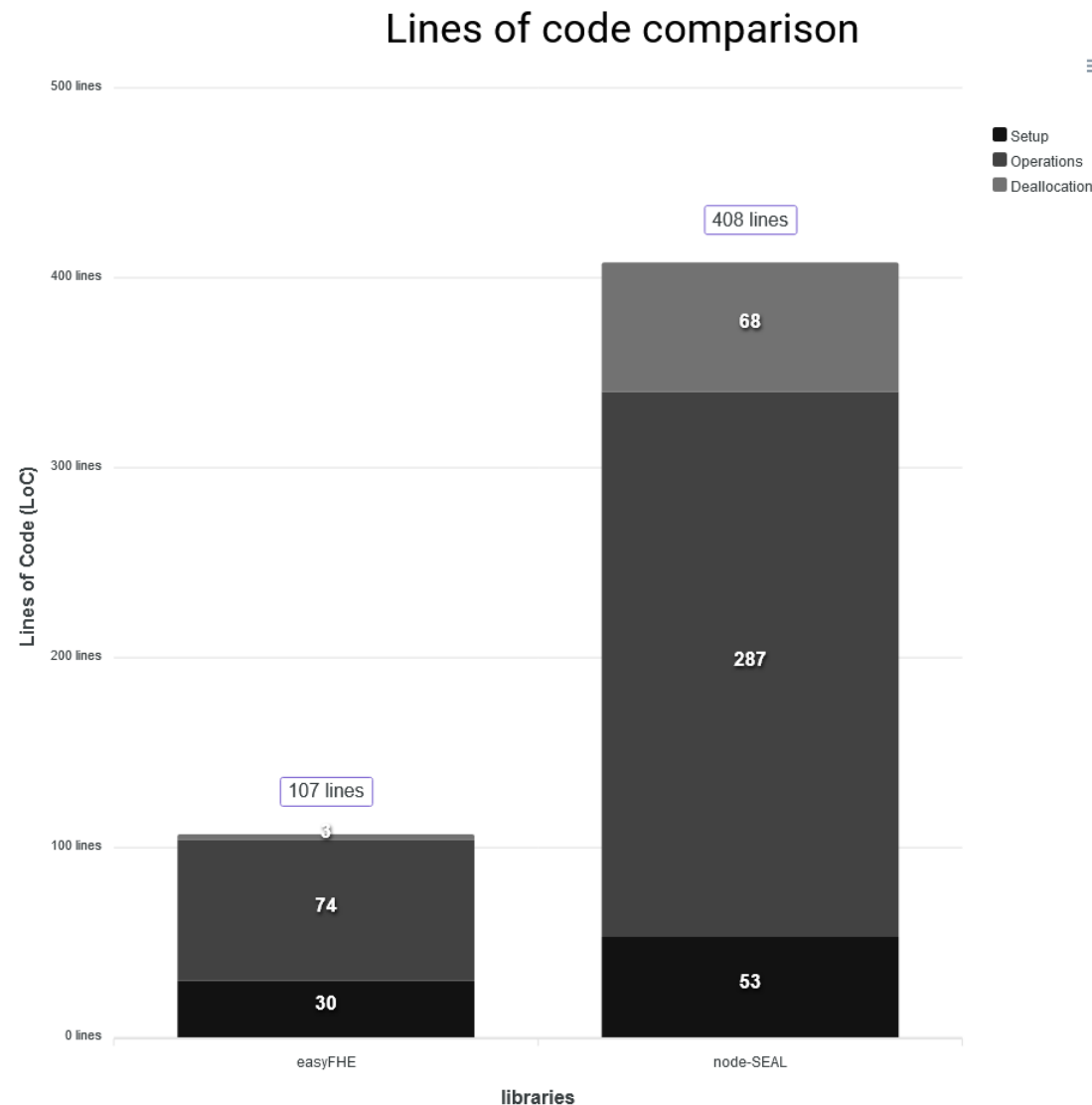


DEMO



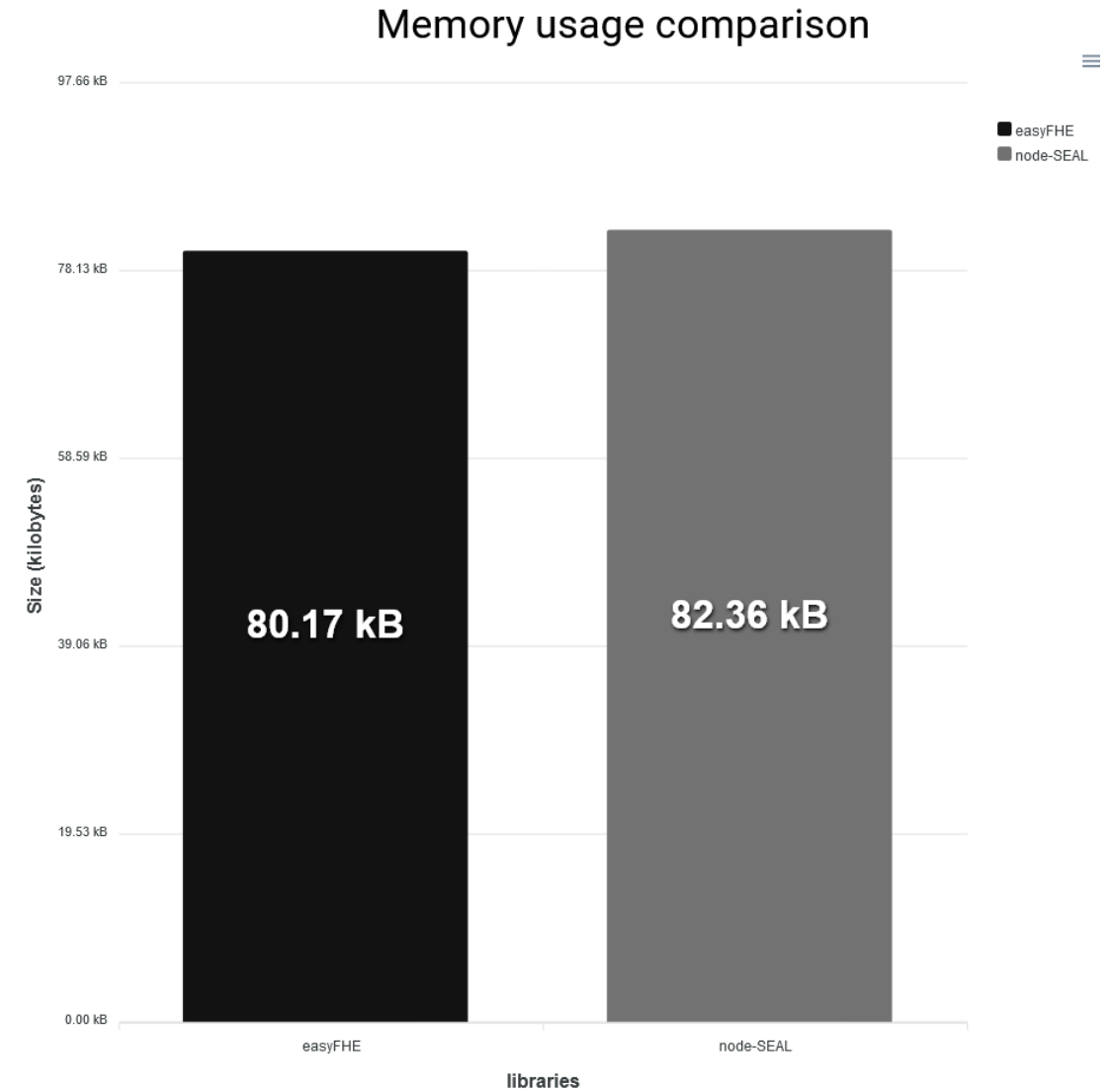
RESULTS

From a developer's perspective, source code written using EasyFHE is on average 26.22% from the node-SEAL equivalent or almost 4 times more compact.



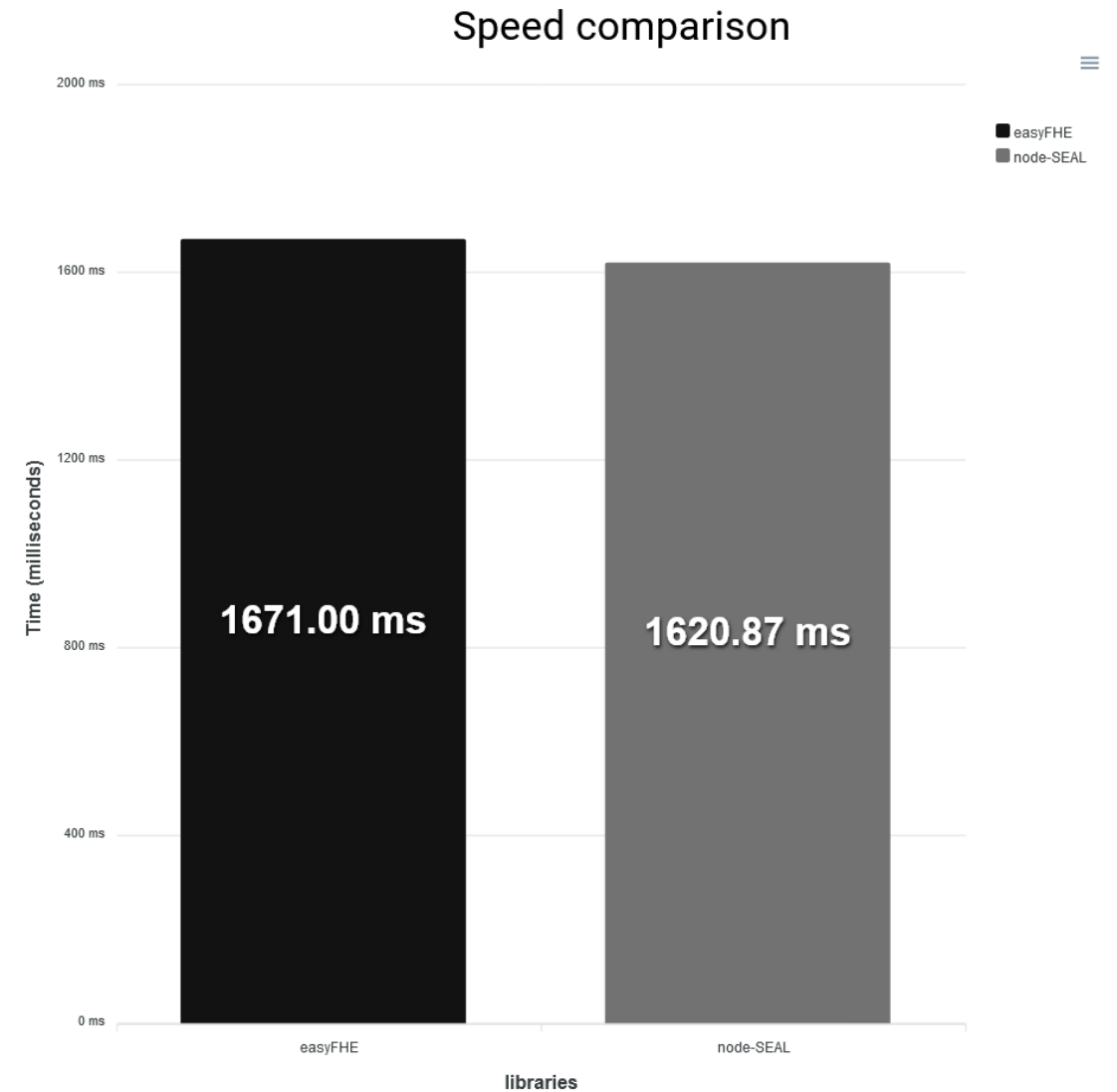
RESULTS

EasyFHE module uses on average with 2.73% less RAM memory due to applied optimizations.

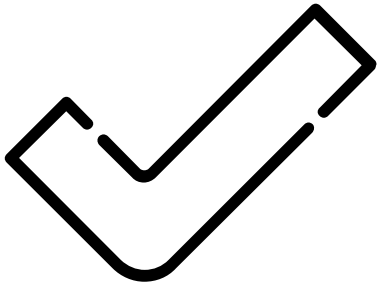


RESULTS

EasyFHE is on average with 3.09% slower than the node-SEAL counterpart.



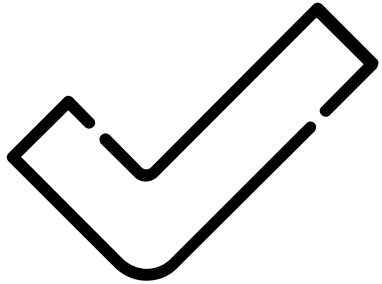
CONCLUSIONS



It is developer friendly.

EasyFHE offers a compressed alternative, with simplified and safe APIs implemented in Typescript.

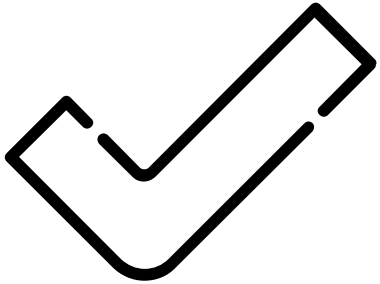
CONCLUSIONS



It is Plug & Play.

EasyFHE it is designed with interoperability in mind, being compatible with websites and Node.js applications.

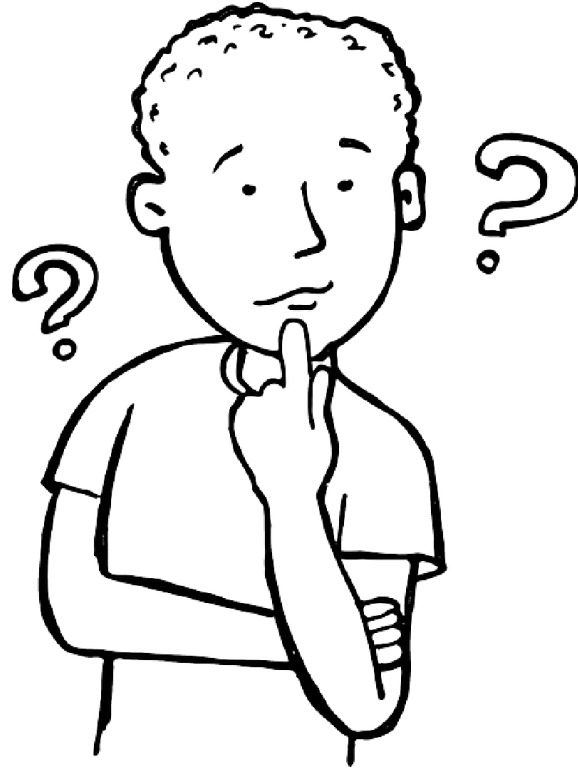
CONCLUSIONS



It is efficient.

It uses the resources intelligently, offering a reduced memory footprint.

Q&A



Preda Mihail Irinel

mihaipreda1997@gmail.com