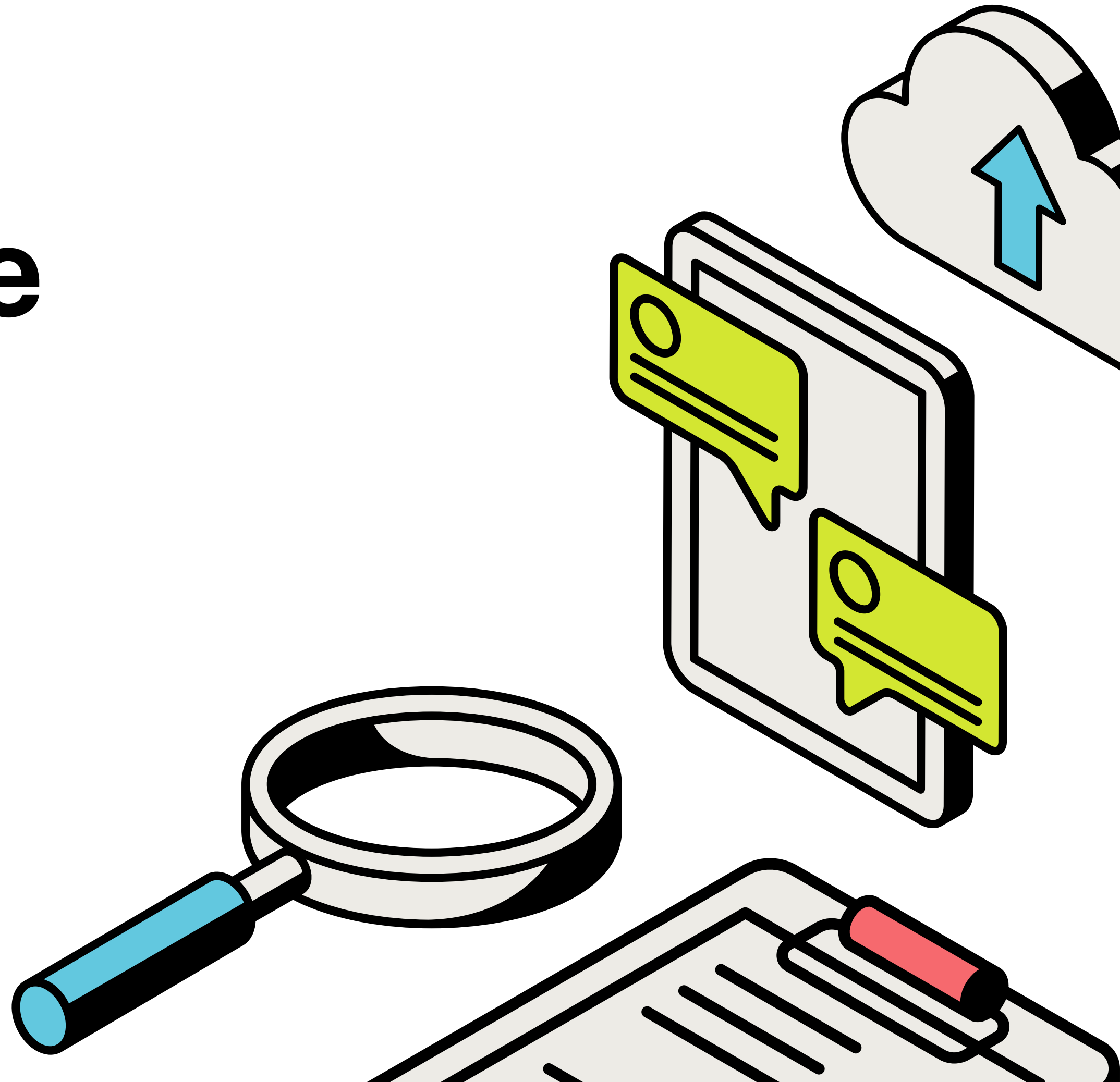




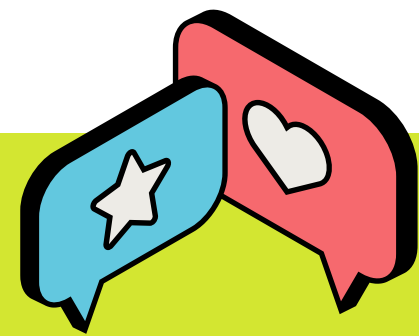
Open Dev Site Project

Date: 03 November 2023

Prepared by: Nico, Ryno, Tshwetso, Ungerer



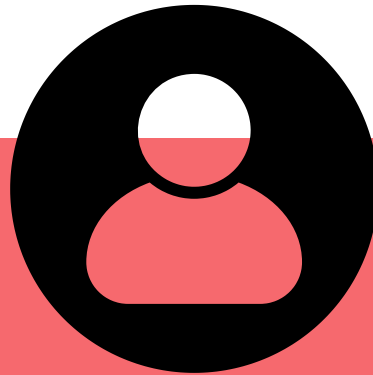
The Team



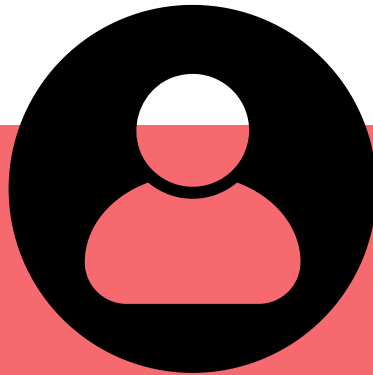
Nico van Wyk
Team leader



Ryno De Beer
Team Member



Tshwetso Mokgatlhe
Team Member



Ungerer Hattingh
Team Member



In This Document:

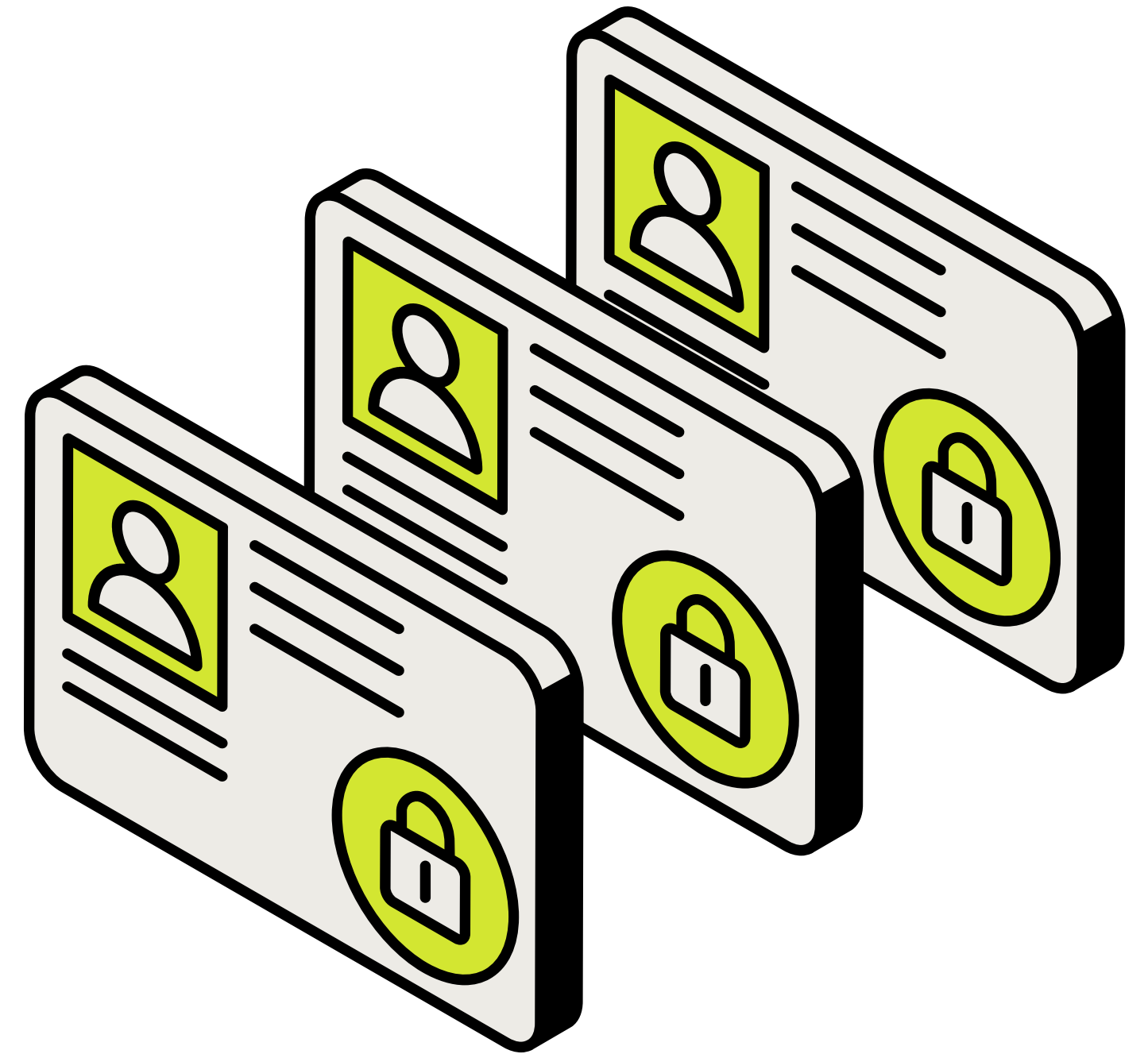
1. Introduction

2. Development Process

3. Planning & Documentation

4. Final Output

5. Conclusion

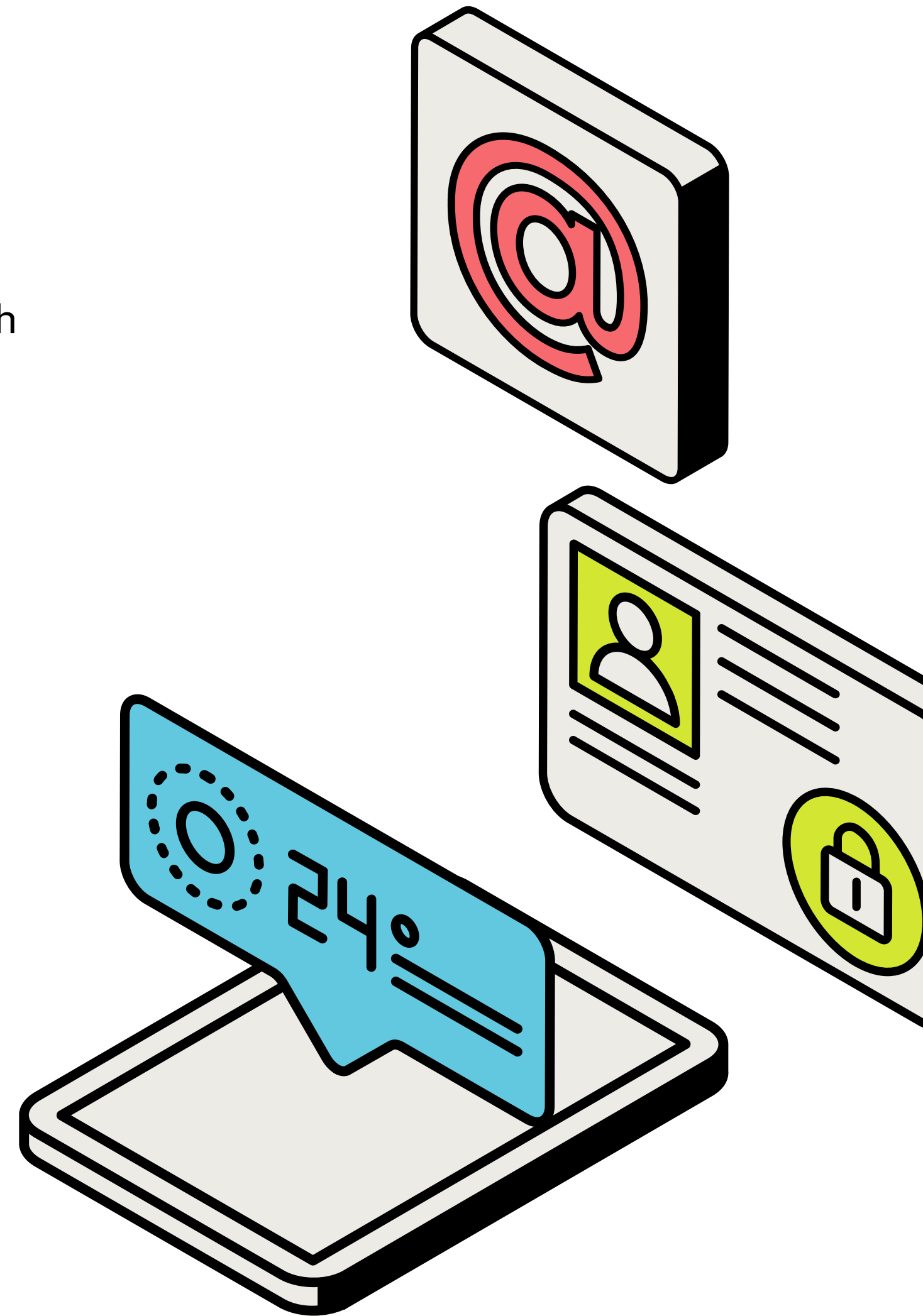


Introduction

In this presentation we will be going through and demonstrating the design and development of our website. followed with a demo walkthrough of the fully launched site.

Brief:

For this Term we were tasked with developing a Q&A website similar to that of Stack Overflow. Choosing Between a MERN stack website or a LAMP site, after which we are to launch the website.



Project Description

OpenDev is a forum where users can ask questions and receive answers regarding Interactive Development. Users can create, update and delete profiles, as well as view other profiles. They can also like or dislike questions, add questions, and answer questions.



Development Process



A

Style

B

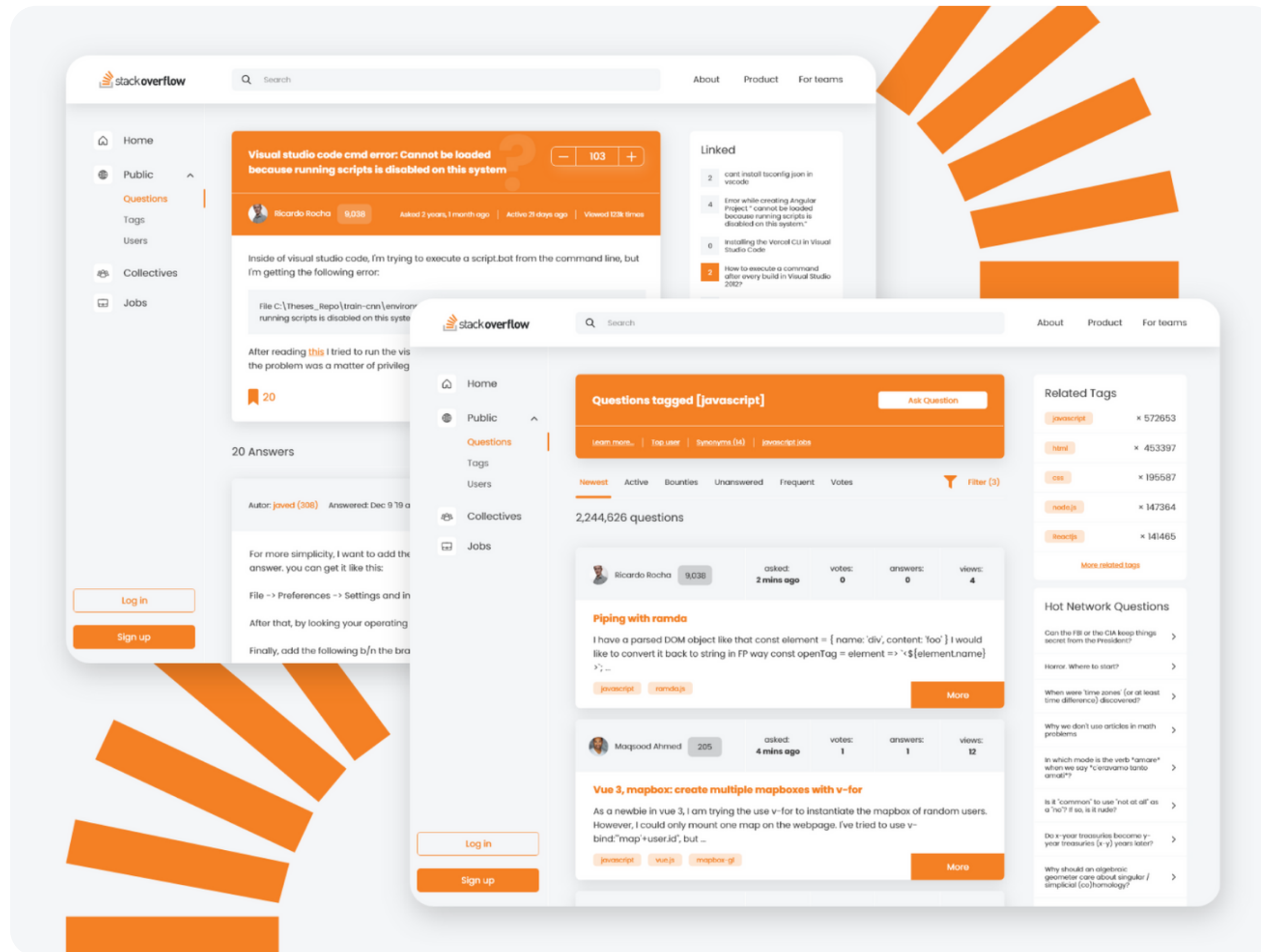
Wireframes

C

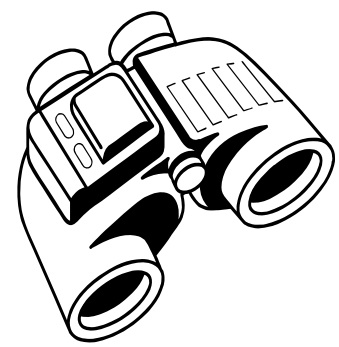
Code



Inspiration



Visual concepts



Colour Palette

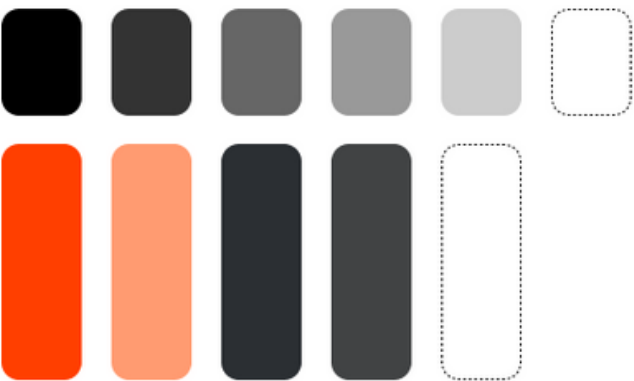


Font

Montserrat

Style Sheet

Color Palette



Typography

Mm

Montserrat

Description

Borem ipsum dolor sit amet, consectetur adipiscing elit. Nunc vulputate libero et velit interdum, ac aliquet odio mattis. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos.

Title

Montserrat Regular 48pt
Montserrat Regular 36pt
Montserrat Regular 30pt

Body

Montserrat Regular 28pt
Montserrat Regular 24pt
Montserrat Regular 18pt
Montserrat Regular 16pt
Montserrat Regular 14pt

Ui Elements

Large Button

Medium Button

Small Button

Login

Logout




Sign Up

Home



Wire Frames

Home page




[Home](#)[Profile](#)[Admin](#)[Sign Out](#)

Welcome to Open Dev Q&A site.


Description:
Yorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc vulputate libero et velit interdum, ac aliquet odio mattis.

Start tutorial



Question cards

Ask a Question

Search 

Question Title

Username

Question Snippet Lorem Ipsum Dolor Sit Amet Lorem Ipsum Dolor Sit Amet Lorem Ipsum Dolor Sit Amet Lorem Ipsum Dolor Sit Amet Lorem Ipsum

Tags

Tags

Tags

Tags

Read More

Question Title

Username

Question Snippet Lorem Ipsum Dolor Sit Amet Lorem Ipsum Dolor Sit Amet Lorem Ipsum Dolor Sit Amet Lorem Ipsum Dolor Sit Amet Lorem Ipsum

Tags

Tags

Tags

Tags

Read More

Question Title

Username

Question Snippet Lorem Ipsum Dolor Sit Amet Lorem Ipsum Dolor Sit Amet Lorem Ipsum Dolor Sit Amet Lorem Ipsum Dolor Sit Amet Lorem Ipsum

Tags

Tags

Tags

Tags

Read More



Question page



Username

Question Title

Question Snippet Lorem Ipsum Dolor Sit Amet Lorem Ipsum Dolor Sit Amet Lorem
Ipsum Dolor Sit Amet Lorem Ipsum Dolor Sit Amet Lorem Ipsum Dolor Sit Amet Lorem
Ipsum

Tags

Tags



Tags

Tags

Answers:



User Page


edit profile

User Name & Surname

Id number

Interests:


Javascript CSS html +

Questions Asked: 00


Questions answered: 00


Likes: 00


logout


delete profile

Asked Questions:

 User Name & Surname


0 Dislikes

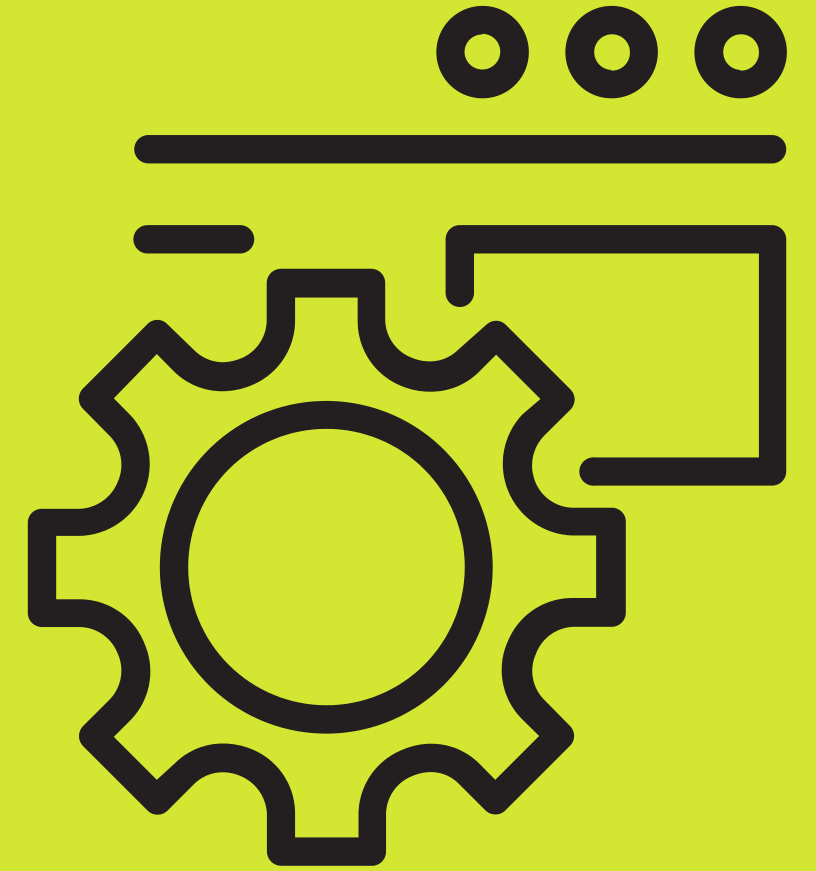

0 likes

View More



OPEN DEV

Planning & Documentation



MERN Stack App

Mongo

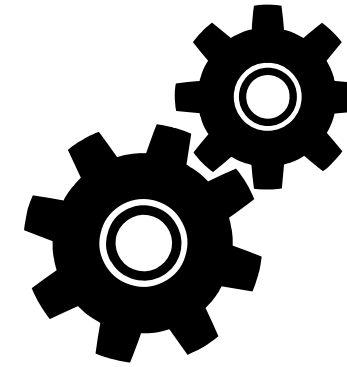
Express

Node.js

React



Technologies & Frameworks



Having decide to build the website using the MERN stack method

Technologies used

- MUI
- React
- Node.js
- Mongo DB
- Express

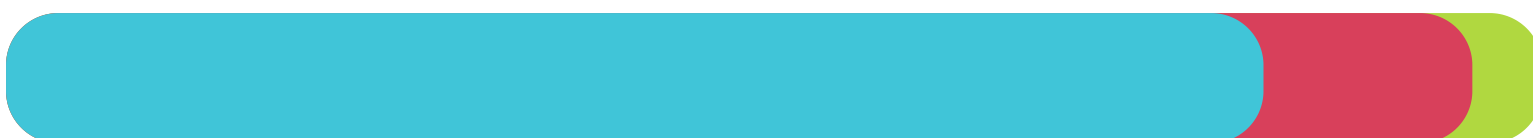
Code base used:

- HTML **1.4%**
- CSS **12.3%**
- JavaScript **86.8%**

JavaScript

CSS

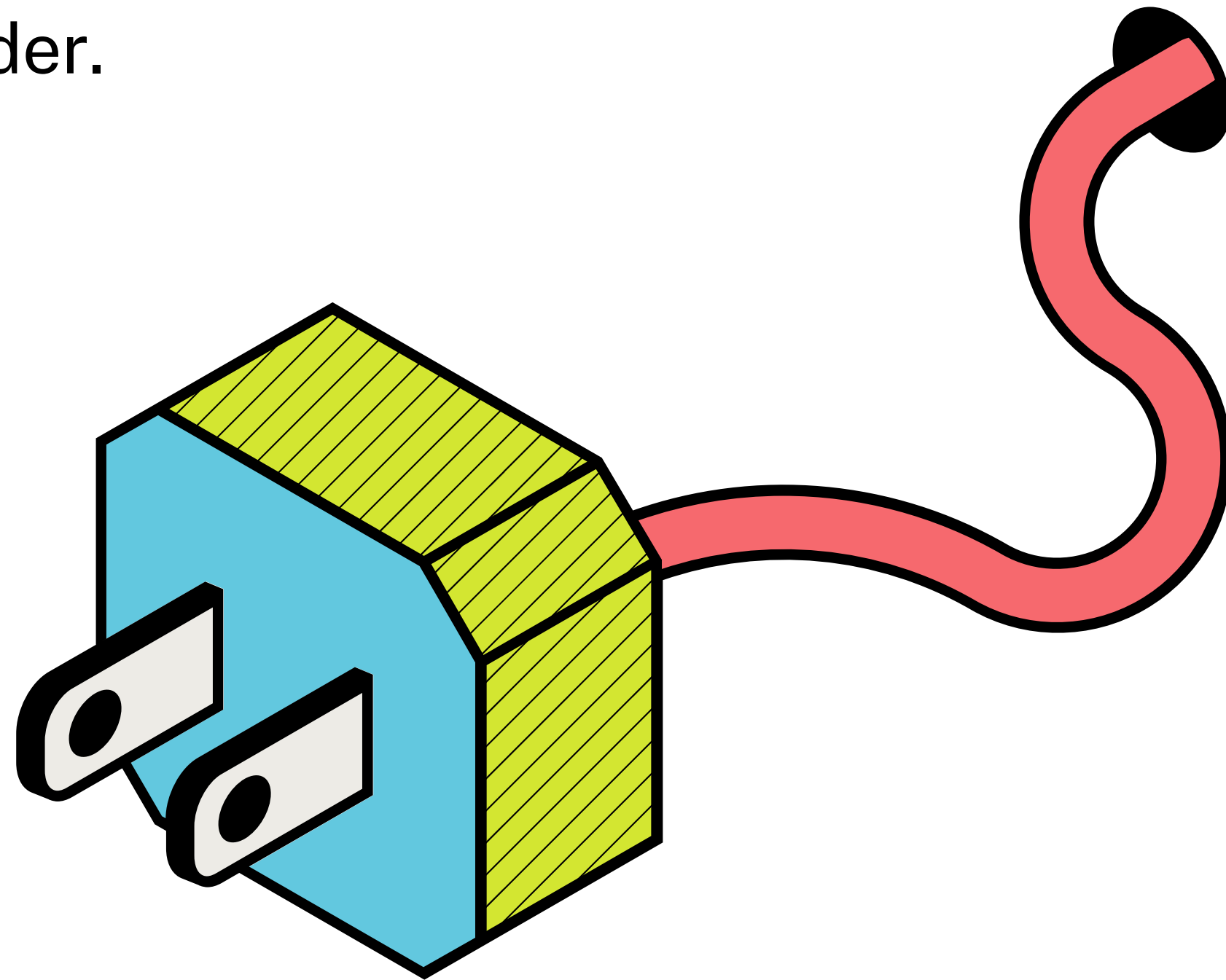
HTML



Additional Material⁺

Additional materials used in the development of the website

- Themeisle: A free illustration svg provider.
- React Icons
- MUI



Problems



1	Hosting the App
2	Bug Fixes reverting When the team pushes to origin
3	Finding the user who is currently Logged in (User ID) without security risks
4	Previously asked questions displaying for all users
5	Observer Loop Error
6	Login errors
7	Stashes and colliding group changes



OPEN DEV

Features and Functionality



Sign up functionality



Sign Up

Username

Email

Password

Already Have An Account?

[Log In](#)

Done

Register Users

If a user signs up, and their data is validated, they will be taken to the **SignIn.js page**. The **data** is validated in the same manner as the sign in functionality, and the data is saved to the database with an **axios.post** call.

Create function

```
// Create
router.post('/api/createUser', async (req, res) => {
  try {
    console.log("Data that was received from the client side:");
    console.log(req.body);

    const user = await User.findOne({ email: req.body.email });

    if (user) {
      return res.status(409).send({ message: "An account with that email already exists" });
    }
    const salt = await bcrypt.genSalt(Number(process.env.SALT));
    const hashPassword = await bcrypt.hash(req.body.password, salt);

    await new User({ ...req.body, password: hashPassword }).save();
    res.status(201).send({ message: "User created successfully!! Log In" });
  } catch (error) {
    res.status(500).send({ message: "Internal Server Error" });
  }
});
```

Login Functionality



Log In

Email

Password

Don't Have An Account?

[Sign Up](#)

Done

Bcrypt

```
const validPassword = await bcrypt.compare(  
  req.body.password,  
  user2.password  
)
```

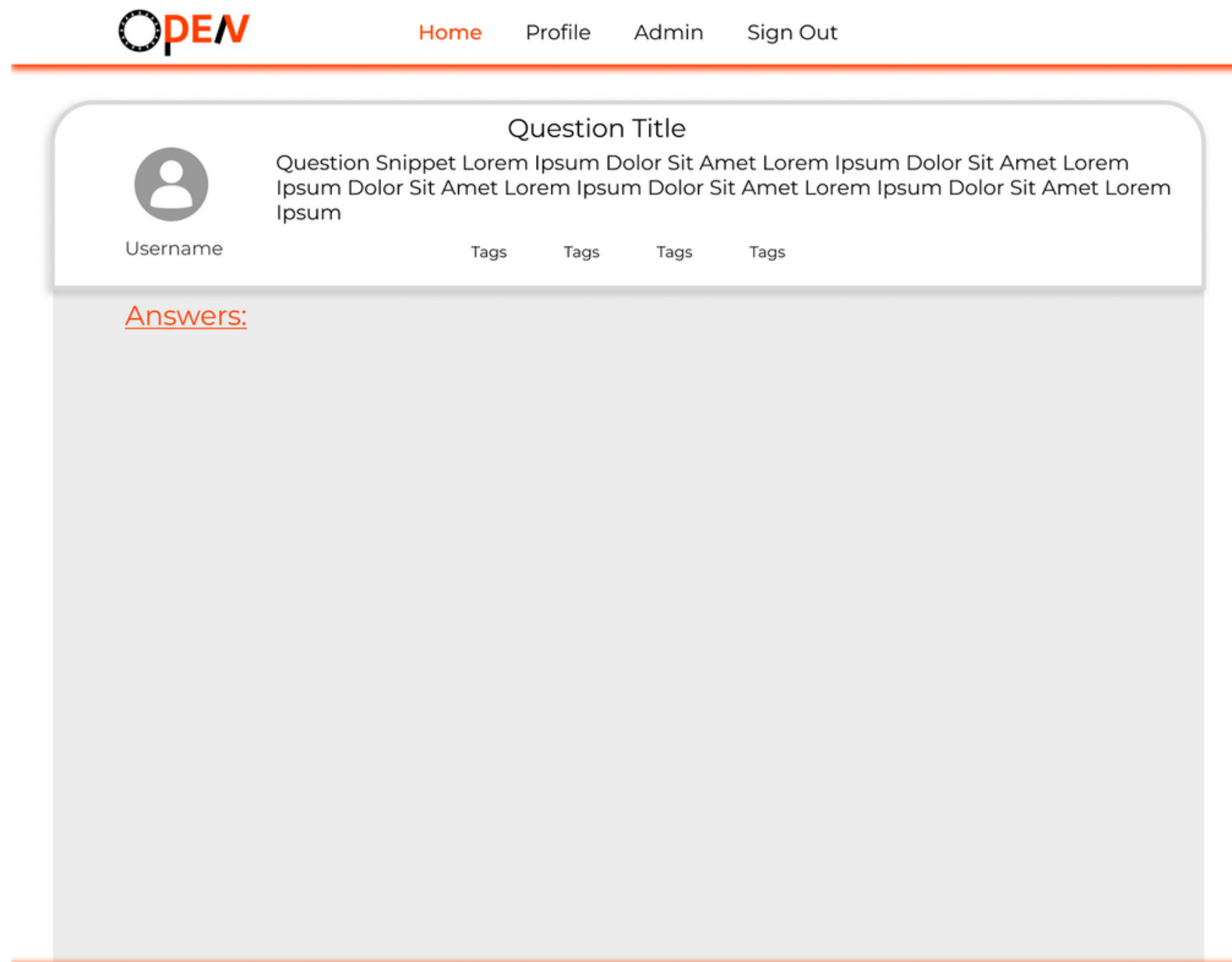
Token generation and error

```
if (!validPassword) {  
  return res.status(401).send({ message: "Invalid Email or Password" });  
}  
  
const token = user2.generateAuthToken();  
  
res.status(200).send({ data: token, message: "Logged in Successfully!" });
```

Authentication

```
const validate = (data) => {  
  const Schema = Joi.object({  
    email: Joi.string().email().required().label("Email"),  
    password: Joi.string().required().label("Password")  
  });  
  
  return Schema.validate(data);  
}
```

Individual Question Page



Get All Function

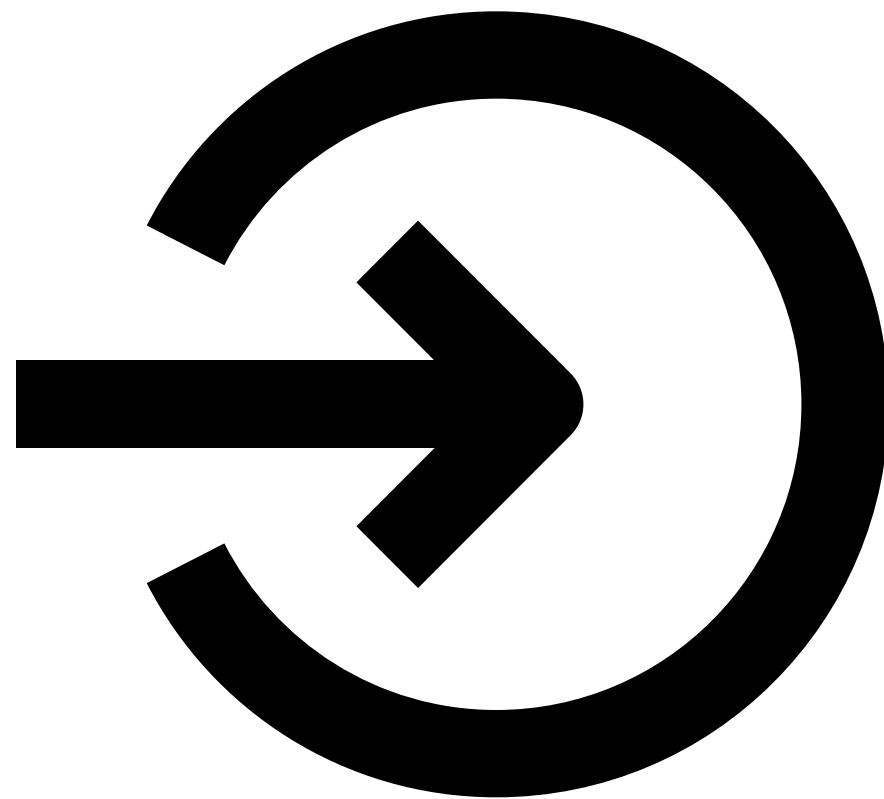
```
// Get all - latest first
router.get('/api/question_get_all/', async (req, res) => {
  try {
    // Find questions and sort by the createdAt field in descending order
    const page = req.query.page || 1;
    const skip = (page - 1) * 5;
    const findQuestion = await QuestionSchema.find().sort({ _id: -1 }).skip(skip).limit(5);
    const totalEntries = await QuestionSchema.countDocuments();
    res.json({entries: findQuestion, totalEntries});
  } catch (error) {
    console.error("Error fetching questions:", error);
    res.status(500).json({ error: "An error occurred while fetching questions" });
  }
});
```

Search Questions Function

```
//search questions
router.get('/api/searchquestion/:search', async (req, res) => {
  try {
    const searchTerm = req.params.search;
    const questions = await QuestionSchema.find({
      $or: [
        { title: new RegExp(searchTerm, 'i') },
        { text: new RegExp(searchTerm, 'i') }
      ]
    });
    res.json(questions)
  } catch (error) {
    console.error("Error fetching questions:", error);
    res.status(500).json({ error: "An error occurred while fetching questions" });
  }
});
```

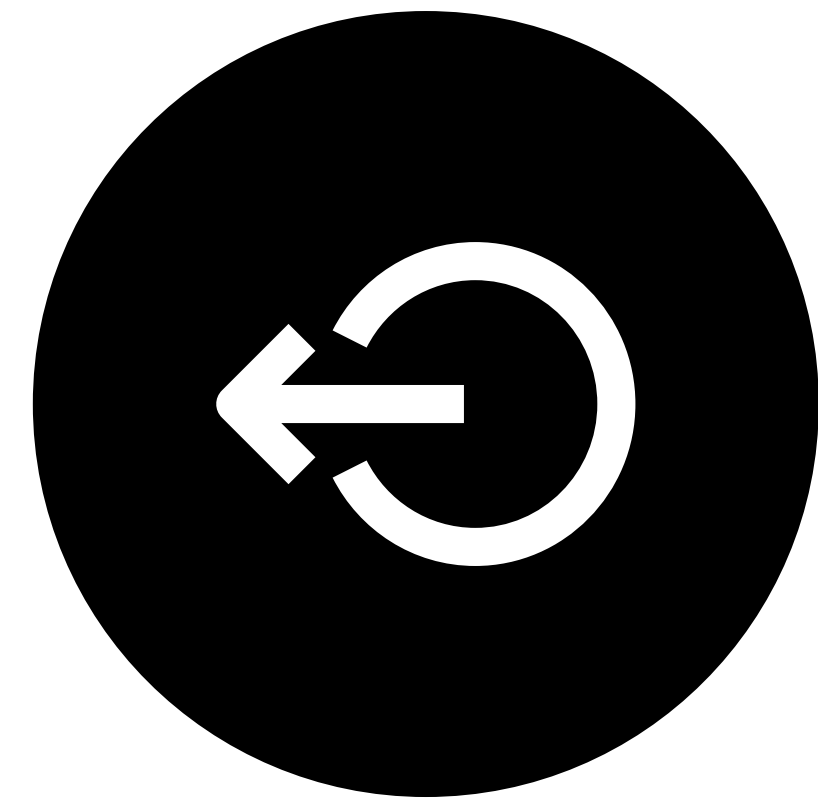
Login User

UseStates are connected to an **OnUpdate event**, which update each time an **<input>** is changed. These values are sent to the server using an **axios.post** call, specifically the **auth.js route**, where it is authenticated. After the data is validated, a **JWT is generated for additional safety.**

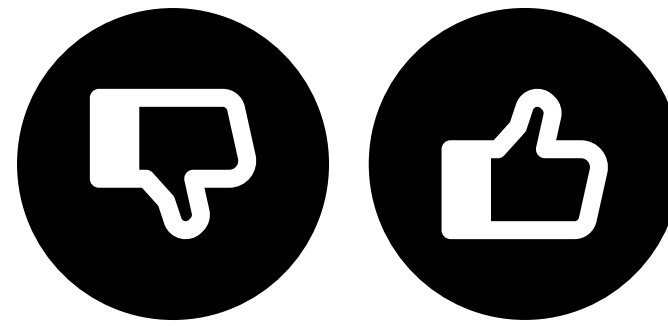


Logout user

When the Sign Out option is clicked on the **Navbar.jsx** component, **SessionStorage** will be cleared and the relevant useState will be updated to change the navbar appearance to remove options that logged out users would not require. **const handleSignOut = () => { sessionStorage.clear(); setIsLoggedIn(false); }**



Upvote and Downvote



Testing

after user is logged in = **Get all** the likes (or dislikes) to **test if the user has already done one or the other**. This is also done to **update** the **values** once the page loads or the question has been **liked/disliked**.

```
// If the user HASN'T liked this post yet if (bFound  
    === false) { // Increase likes by 1 addLike(1);
```

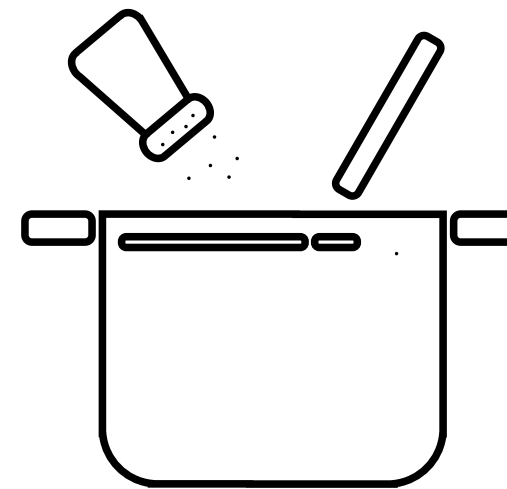


BCrypt

When user is registered

The bcrypt password is generated

Using SALT, a random value is added to the end of the encrypted password to further encrypt it



After the SALT is generated, the password is Hashed.

#

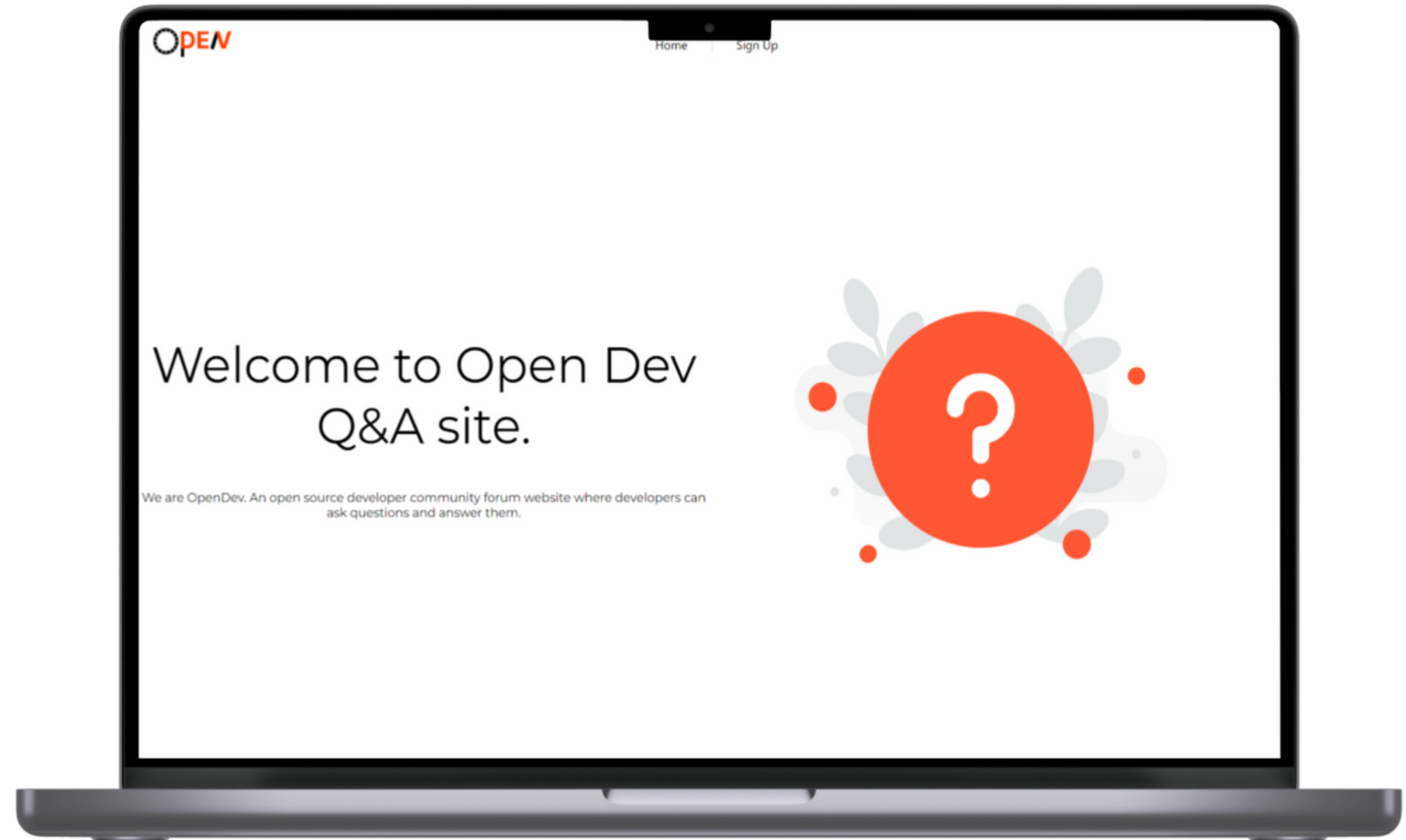
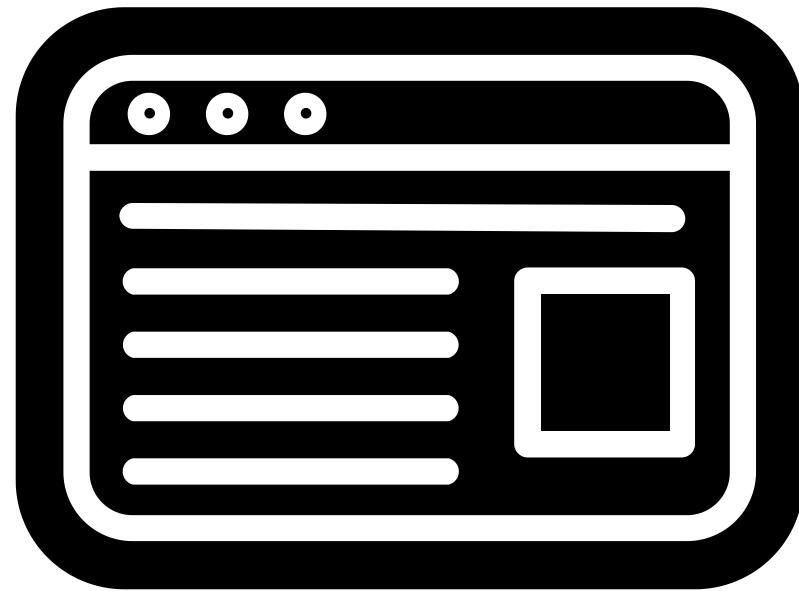


OPEN DEV

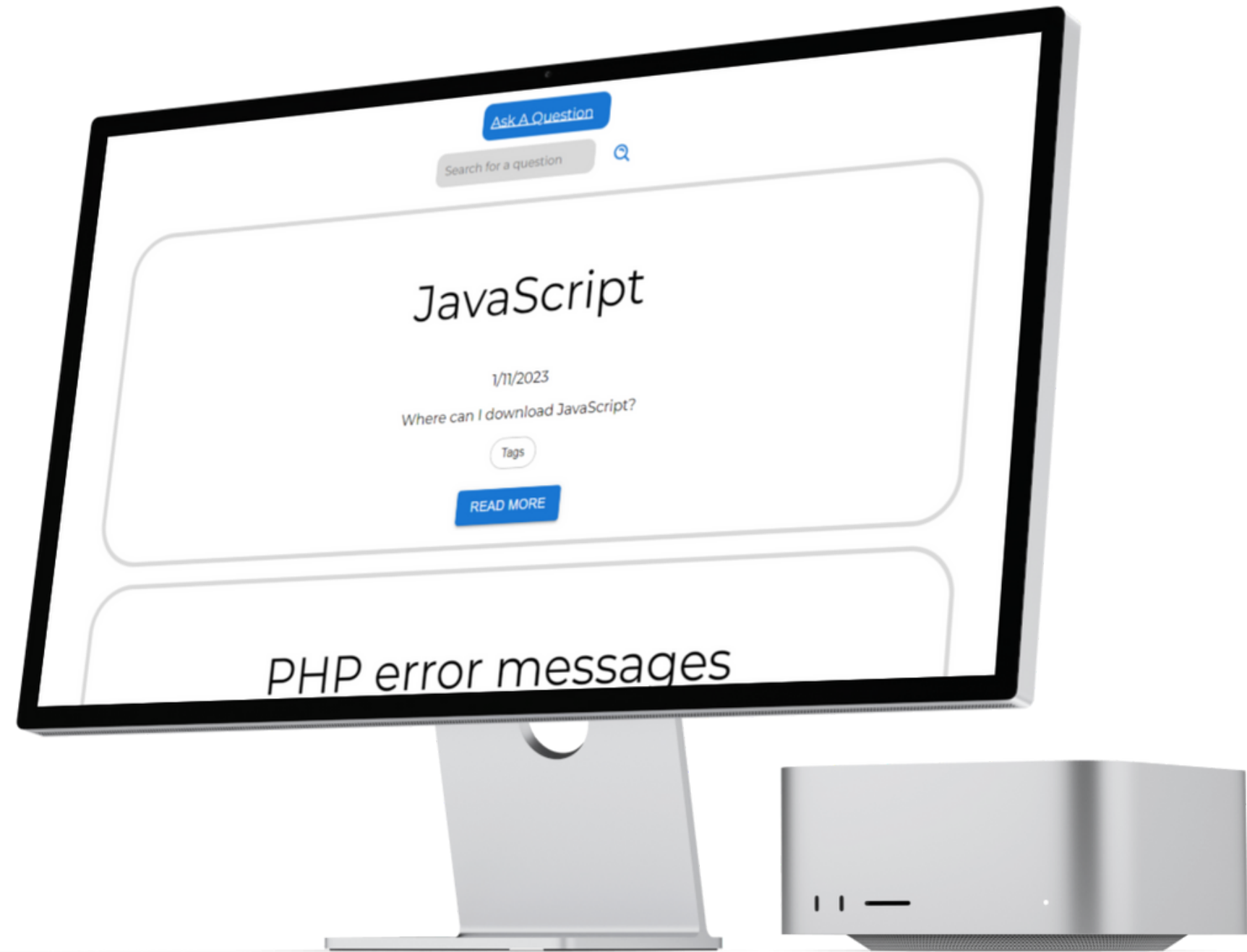
Mock-Ups



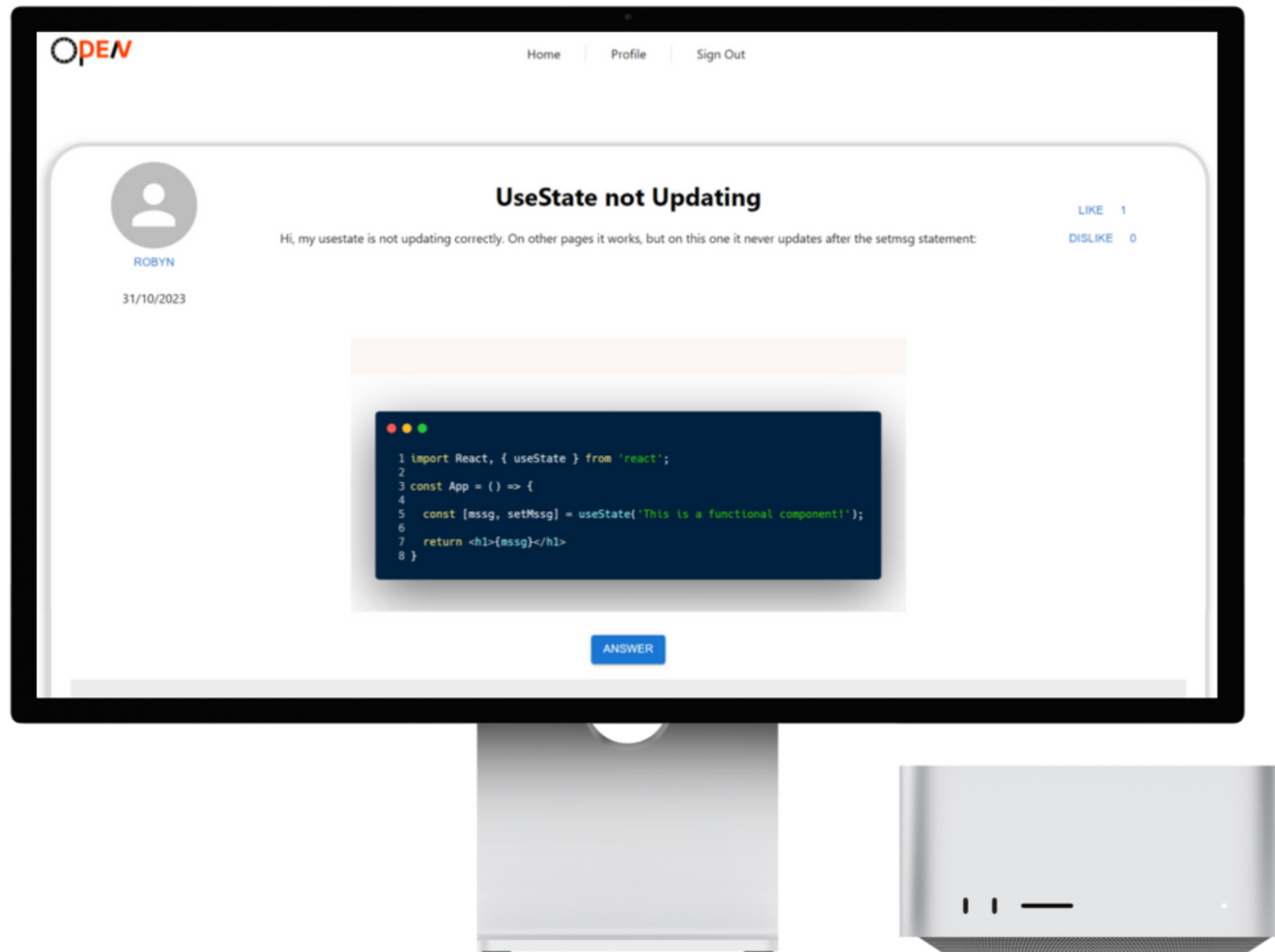
Landing Page



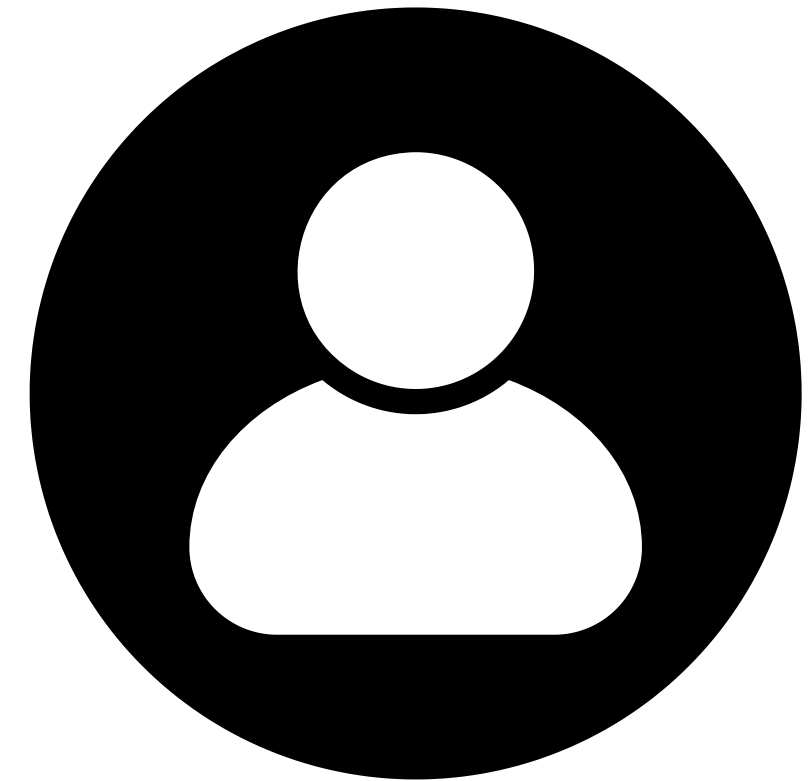
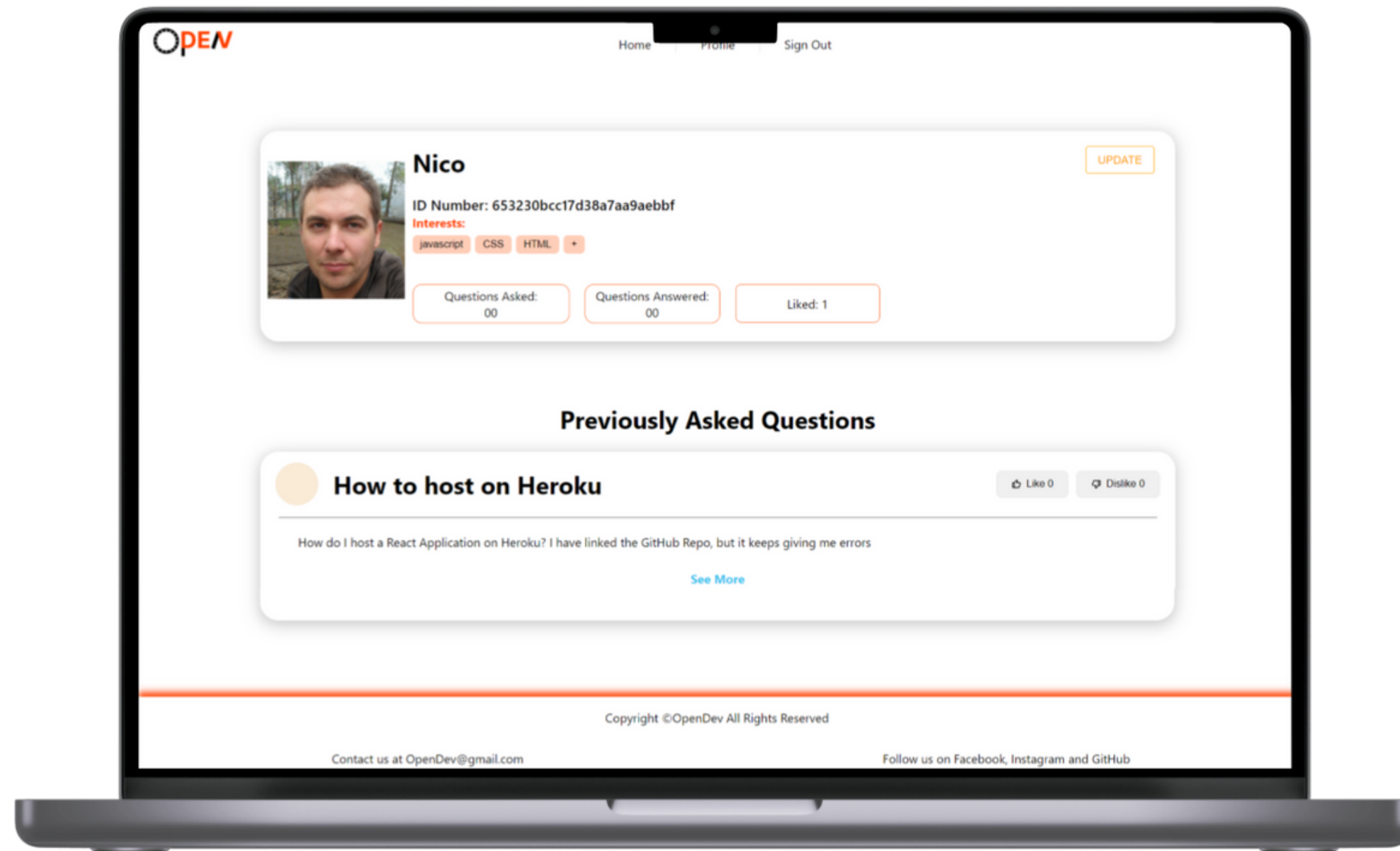
Question cards

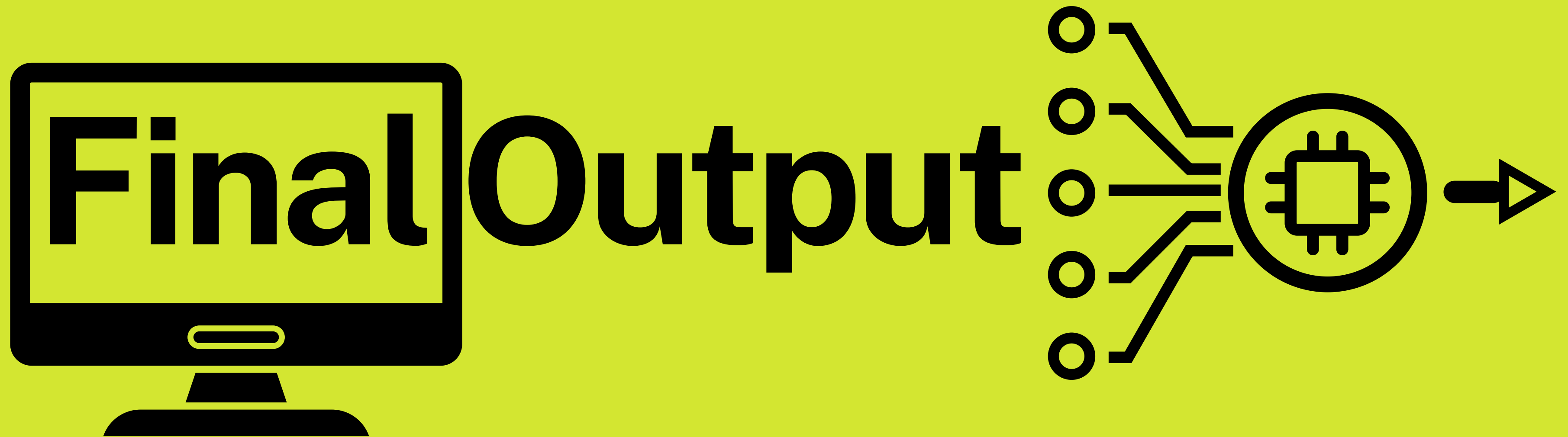


Question page



Profile page





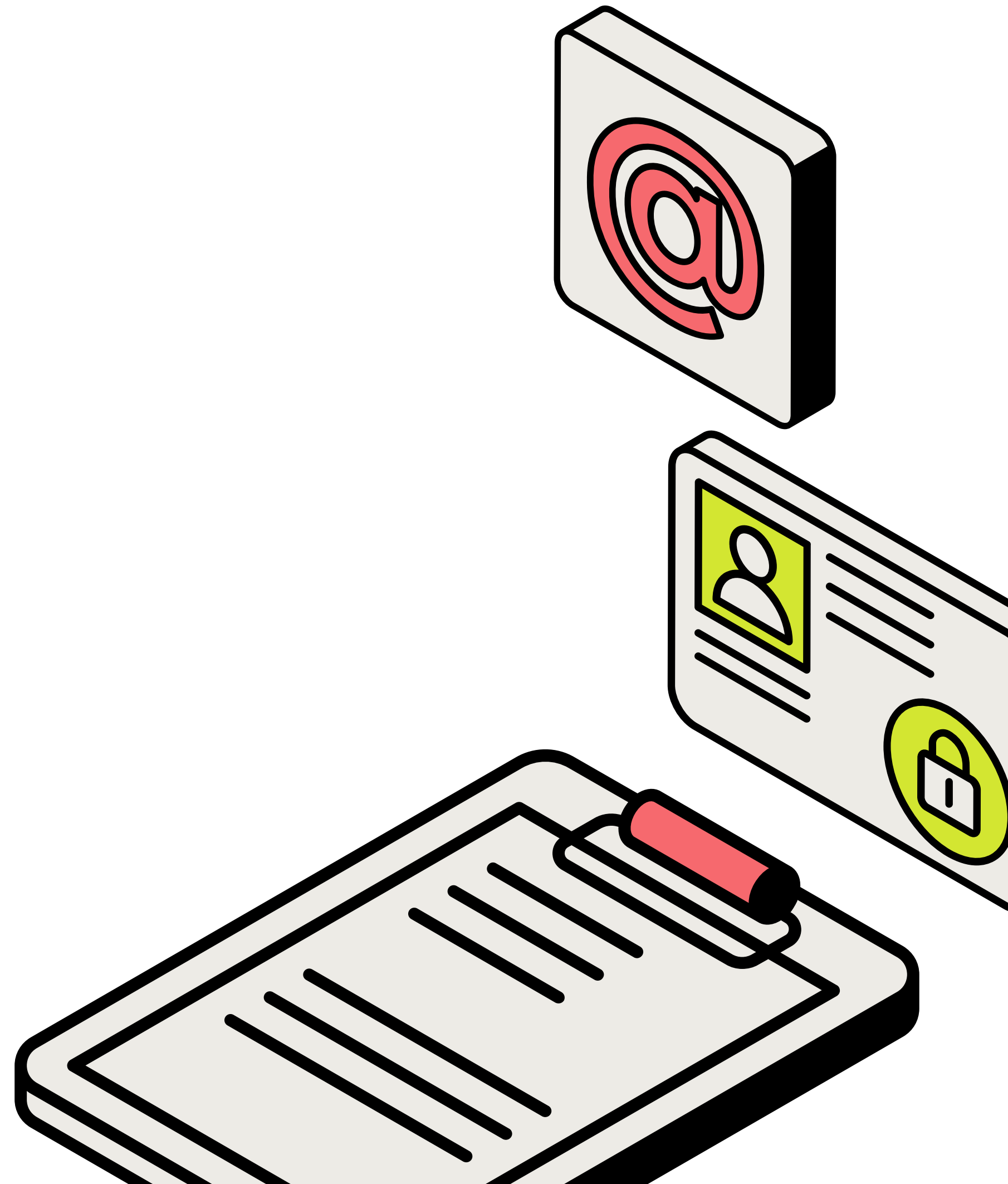
Site Demonstration

<https://opendevweb-51212536012a.herokuapp.com/Home>

<https://opendev-74d7e.web.app/Home>

Conclusion;

```
function Conclusion = () => {  
  return (  
    <p> That concludes our presentation. </p>  
  );  
};
```



Thank you.

