# Iteration 3 – Development log

Iteration 3 is about the accounts system and making sure everything is functioning correctly both front-end and back-end wise. In this iteration, I may be editing styles as well as functionality.

I first checked if the registration form was functioning correctly.

I conducted a test to see what would happen if I submitted the form with no data. An error occurred, and I made the following changes to fix it:

A screen shot of a computer screen

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

I then found an issue with my registration form not functioning properly. It wasn’t picking up any of the data that was being submitted. After a length amount of testing, I was able to find the issue. The form element in the HTML must be the first element in the page. I’m not sure if this is a global rule for flask, but whenever I made that change, it worked.

A screen shot of a computer program

AI-generated content may be incorrect.

And I changed the back-end Python code to the following:

A computer screen shot of a program code

AI-generated content may be incorrect.

Next, I need to add validation checks to ensure that users are only entering emails into the database.

I added this to check:

A black background with white text

AI-generated content may be incorrect.

And now it works.

I added some SVG images to the register & login pages to liven them up a bit.

A screenshot of a computer screen

AI-generated content may be incorrect.

A screenshot of a cartoon of a person walking

AI-generated content may be incorrect.

I made them automatically hide themselves when the screen gets too small.

Next, I setup the account page. This uses a tab system as per the design in the design documents. I used dynamic routes in Flask to achieve this.

A black background with white text

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

I then had to construct the page. Here’s how I did that:

A white card with black text

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

I wanted to make a way that would show which tab was currently active. This is what I ended up with:

A close-up of a login screen

AI-generated content may be incorrect.

It keeps the hover styles after you click on it. This is how I achieved it:

A screen shot of a computer program

AI-generated content may be incorrect.

Then, I added an additional link which was the logout route. Since this couldn’t be active, I didn’t need to do anything apart from make it link to /logout.

A screenshot of a login form

AI-generated content may be incorrect.

I added some user feedback when the user logs in successfully.

A green field with trees in the background

AI-generated content may be incorrect.

It redirects them to the home page and shows them this message.

Now, I’m going to make the password reset feature. Users will enter their email address, and it will send an email to that address if it is associated with an account in the database.

I will use Flask-Mail for this.

I added lots of documentation to make the code more readable and maintainable.

A computer screen with colorful text

AI-generated content may be incorrect.

Next, I wanted to add 2 factor authentication to the accounts system. To do so, I will be using a QR code library for flask called Flask\_QRCode.

I first made the 2fa page:

A screen shot of a computer program

AI-generated content may be incorrect.

The IMG element is how the QR code is rendered. The Secret 2fa variable is what is stored in the user’s data.

I then had to change the content dynamically based on if the user had 2FA activated or not:

A screen shot of a computer program

AI-generated content may be incorrect.

This is what I ended up with for the backend of the 2fa functionality:  
A screen shot of a computer program

AI-generated content may be incorrect.

I had a slight issue with the system not detecting when the user hadn’t setup 2fa, so I had to add this additional condition to the if statement:

A screen shot of a computer

AI-generated content may be incorrect.

Now everything works, I will make a way for users to disable their 2FA.

The final code:

A screen shot of a computer program

AI-generated content may be incorrect.

A computer screen shot of a code

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

Now I will make the system ask for the 2FA code when the account is trying to be logged into.

I had to add another route:

A computer screen shot of a program code

AI-generated content may be incorrect.

And redirect to it accordingly:

A screen shot of a computer screen

AI-generated content may be incorrect.

2fa is all working now!