



Professional Practice in IT Project Document

Cryptolio By

Patrick Murray-G00344530

& Cory O' Donoghue - G00376678

Supervised by John French

Contents

<i>Introduction.....</i>	<i>2</i>
<i>System Requirements.....</i>	<i>3</i>
<i>Technology Used.....</i>	<i>3</i>
<i>Design methodology.....</i>	<i>4</i>
<i>Features of the implementation.....</i>	<i>6</i>
<i>Testing</i>	<i>10</i>
<i>Limitations and known bugs.....</i>	<i>12</i>
<i>Improvements for future.....</i>	<i>12</i>
<i>Conclusions.....</i>	<i>13</i>

Links

Link to Selenium Testing: <https://youtu.be/OV3wD4INPIA>

Link to Project Screencast: <https://youtu.be/G1vI0h1nJb8>

Link to GitHub: <https://github.com/Cryptoloo/Cryptolio>

1.Introduction

For our 3rd year project in professional practice in IT, we decided to develop a full stack web application called 'Cryptolio' which enables a user to register/login to our webpage. Once logged in they will be able to create a portfolio of all the crypto currencies they hold and their current overall value. The website also offers a contact us support desk for any queries the user has. All the user's information will be stored in our MongoDB database.

To develop the client side of our application we used a JavaScript library called React. We chose React because it lets you build high quality user interfaces, its capable of speeding up the development process with its reusable components and development tools and it is currently one of the most popular frontend technologies in the market.

As we came to decide what database to use in the backend of our application, we felt MongoDB was the superior choice. We chose MongoDB over MySQL because of our experience with both has showed us MongoDB is much less complex being schema free making it easy to change or delete documents. In the future if we decided to keep working on our website, MongoDB would make this much easier to build on and enhance applications without needing complex schema as with a relational database.

Our main objectives for the project were:

1. Have a working register and login system.
2. Have a backend which stores user details and their portfolio information.
3. Set up communication between user and contact us.
4. Deploy the final product.

Project Components	Cory	Paddy
Landing Page	Login and Register	Portfolio (create, edit, delete)
Login and Register	Forgot Password	Position Size calculator
Forgot Password	Reset Password	Contact Us
Reset Password	Backend	Profile Page
Portfolio (create, edit, delete)	MongoDB	Landing Page
Position Size Calculator	Testing	Backend
Contact Us	Document & READ.ME	MongoDB
Profile Page		App Deployment
Backend (Express)		Document & READ.ME
MongoDB		
App Deployment		
Testing		
Document & READ.ME		

2. System Requirements

Our system requirements for the finished product are:

- The user should be able to register if they do not have an account.
- The user should be able to login using previously created credentials.
- Once the user has logged in they should be brought to the portfolio page where they can add/edit/delete any cryptocurrency we have in our database
- From the home page there should be a drop-down menu accessible in the right-hand corner which give the user options to go to profile page, a contact us page, a position size calculator page, and a logout button.
- On the profile page the user should be able to edit their details such as their first name, surname, or email. If they wish to edit their password, they must first enter their current password.
- On the contact us page the user should be able to enter their email, what they wish to enquire about which they can select from the drop-down menu on the input field and also specify in more detail what they are emailing us about.
- The position size calculator page the user can work out the monetary risk on the trade they wish to have.
- The contact us and position side calculator should both also be accessible from the landing page.

3. Technology Used

1. JavaScript:

JavaScript was the chosen programming language for our project. We went with JavaScript for several reasons. One of the biggest reasons was it is very fast as it can be ran immediately at run time, with JS you can also create very rich interfaces as well as it is the most in demand language for web development.

2. React:

When it came to developing the UI, we chose React. This was an easy decision as React is one of the best front-end technologies on the market right now with some of its benefits being its speed, usability, and performance. On top of all that it is slightly easier to learn than other frameworks like angular which we had experience with previously in the course.

3. Express.js:

Express is a framework that runs withing Node.js that allows developers to create and maintain servers. We used express to write the server-side logic for our web and mobile application. Express is well adopted and there is a lot of documentation available for it online.

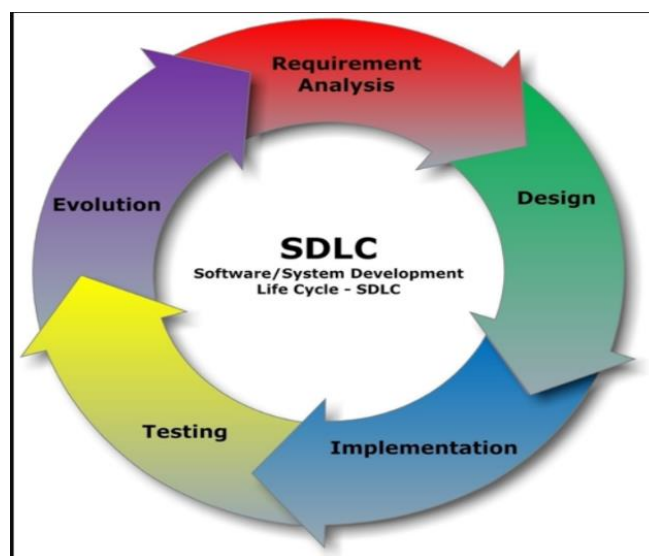
4. MongoDB:

For the backend we went with MongoDB. After considering MySQL we realised the obvious choice for us was MongoDB. It is much easier to work with MongoDB because of its flexible document schemas which allow virtually any structure to be modelled and manipulated effortlessly in comparison to a complex relational schema.

5. GoDaddy & Google App Engine:
To deploy our website, we required a domain for which we bought from GoDaddy. We changed the name to cryptoloo.com and edited ipv4 and the ipv6 to match the google app engines ipv's. We used the app engine over the compute engine as both achieve same task however app engine is much more time efficient.
6. SendGrid: We chose SendGrid as our cloud-based SMTP provider which offers us and manages an email server enabling us to send emails about sign ups, forgot passwords, newsletters and provide contact us support for our users.

4. Design Methodology

We spent time considering what was the best approach to use in developing our product and concluded that the Software Development Life Cycle (SDLC) would be the best to help us achieve our end goal.



1.Requirement Analysis: We established a high-level view of the intended project and determined what the end goals of our application were.

2.Design: We described desired features in detail, including screen layouts and style decisions. See image (a) and (b) below.

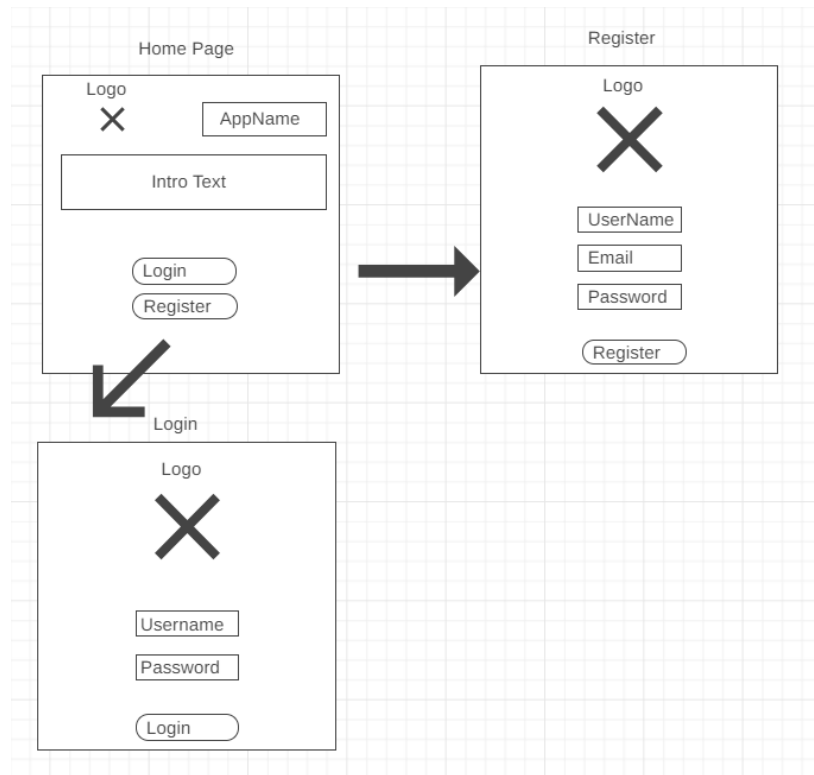
3-Implementation: The html and CSS code were written at this point along with the JavaScript and other necessary components.

4-Testing: To carry out testing on our front end we used a software called selenium which we discovered from our module software Testing. Selenium enables us to screen record a set of tasks which once completed the recording is manually stopped. Selenium then repeats steps checking the tested functionality works presenting the tasks. In the example below the following test is checking to see if a user can register with the website. For the back end we used white box testing as we were both very familiar with the internals of the application.

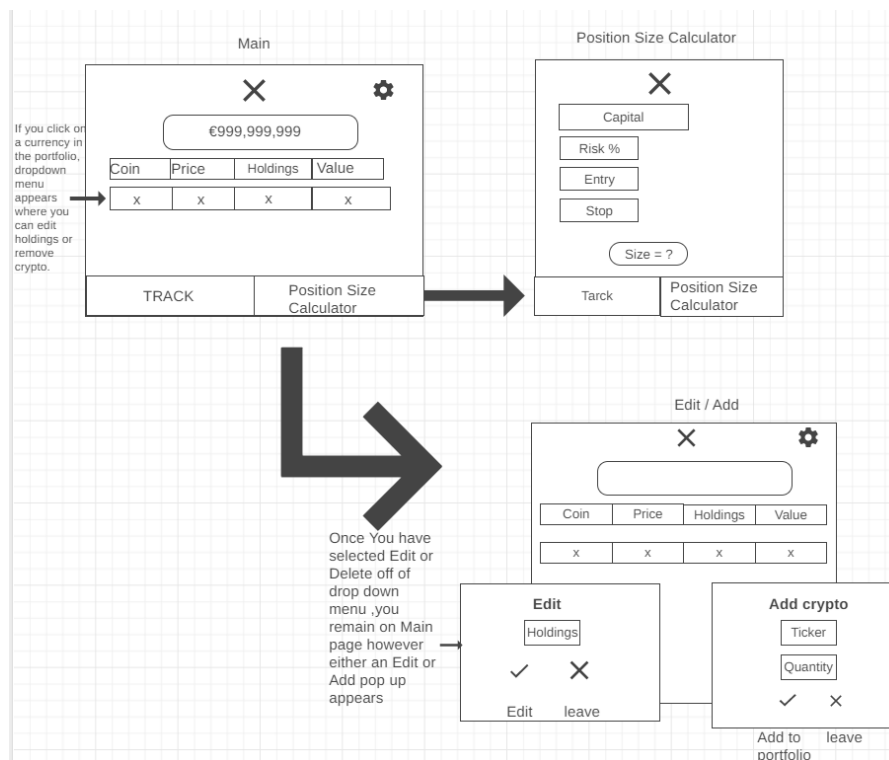
5-Evolution: Looked at further development ideas and opportunities for the future.

Design:

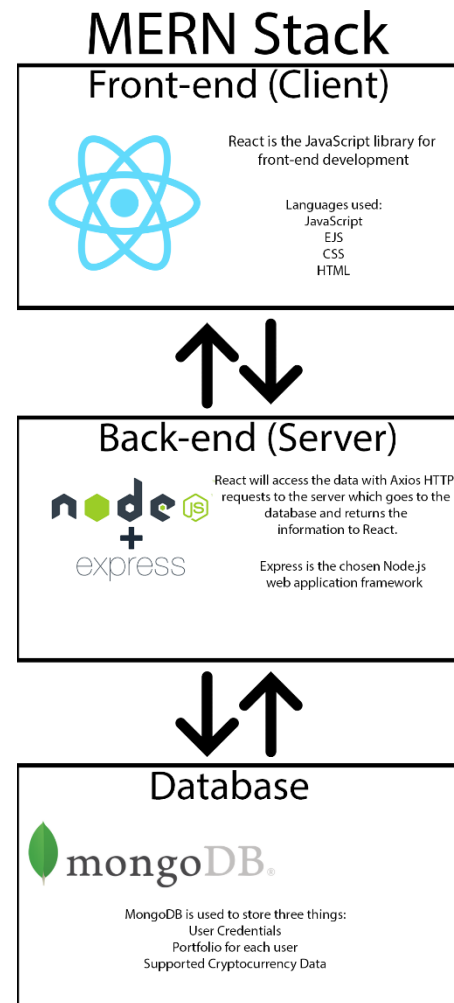
Example (a)- and Login and Register design



Example (b): Portfolio Page and Position size calculator



Example (c): Mern Stack Development



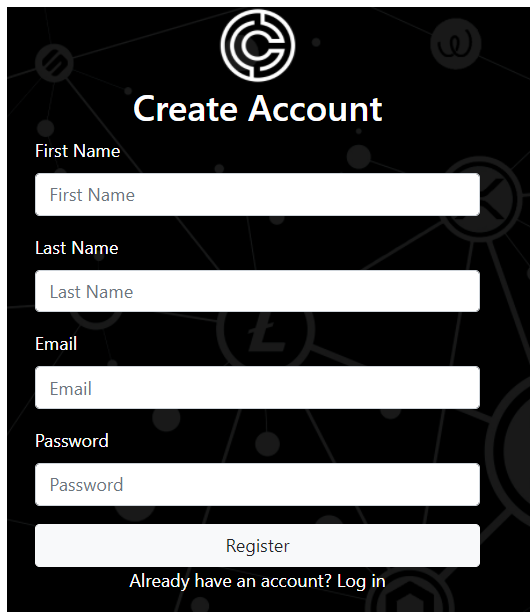
5. Features of the Implementation

Our project consists of several features such as login/register, forgot password, reset password, edit password/email, add/edit/delete cryptos, drop down menu, contact us, and a position size calculator.

Register: To create an account the user is required to enter a first name, surname, email, and password. These details are then stored in our database. Every username must be unique, if a duplicate email is entered when creating an account, the register will fail. See image (b) below.

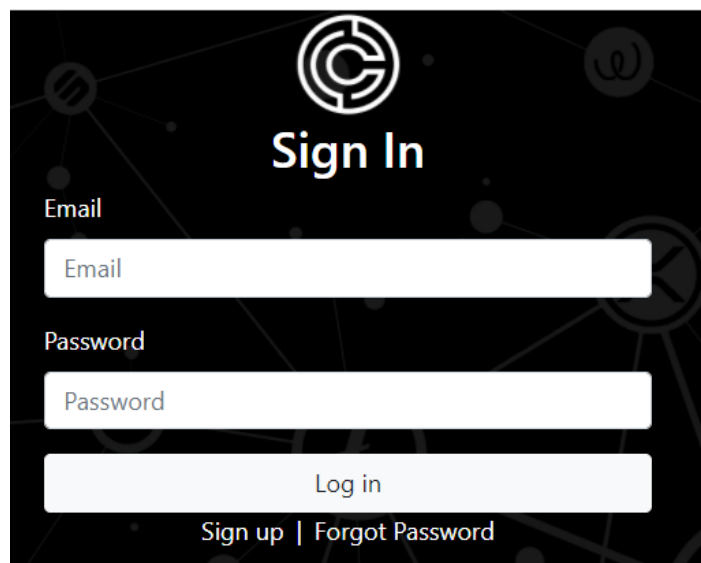
Login: To login the user must enter previously created credentials. If the email or password does not match what is stored in the database, the login will fail. The website can accept multiple users to be logged on at once. See image (b) below.

Image (a)



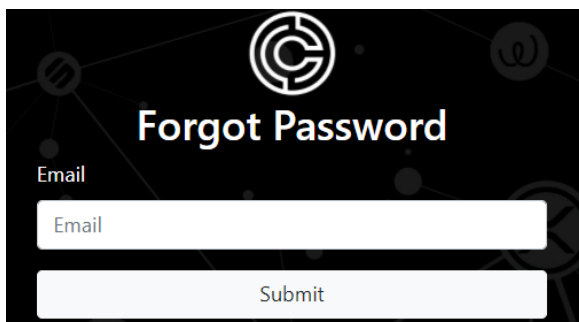
The 'Create Account' form features a dark background with a subtle pattern of cryptocurrency icons. At the top is a logo consisting of a stylized 'C' with a vertical bar. Below the logo, the title 'Create Account' is centered. The form includes four input fields: 'First Name', 'Last Name', 'Email', and 'Password'. Each field is preceded by its respective label. At the bottom, there is a 'Register' button and a link that says 'Already have an account? Log in'.

Image (b)



The 'Sign In' form has a similar dark background and logo as the 'Create Account' page. The title 'Sign In' is centered at the top. Below it are two input fields for 'Email' and 'Password', each with its label to the left. A 'Log in' button is positioned below the password field. At the very bottom, there are two links: 'Sign up' and 'Forgot Password'.

Forgot password: Forgot password page uses SendGrid to send an email from our domain to the user with a link to reset their password.



The 'Forgot Password' form maintains the dark theme and logo. The title 'Forgot Password' is centered. There is a single input field for 'Email' with the label to its left. Below the input field is a 'Submit' button.

Password reset Inbox x



no-reply@cryptoloo.com
to me ▾

You requested password reset

Click this [link](#) for password reset

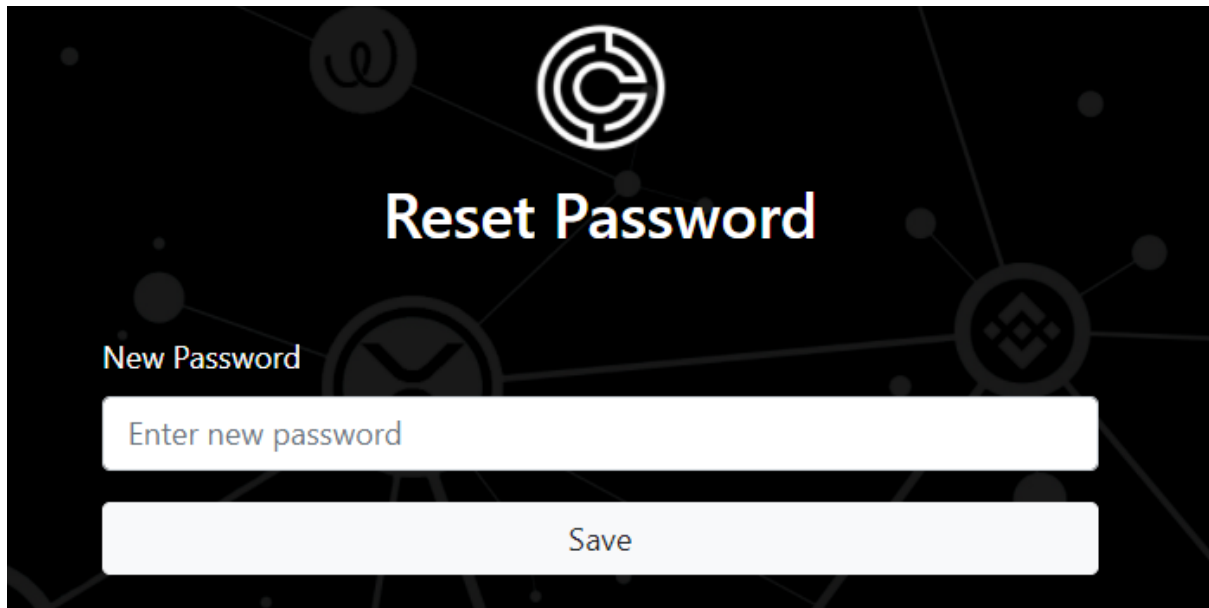


Reply



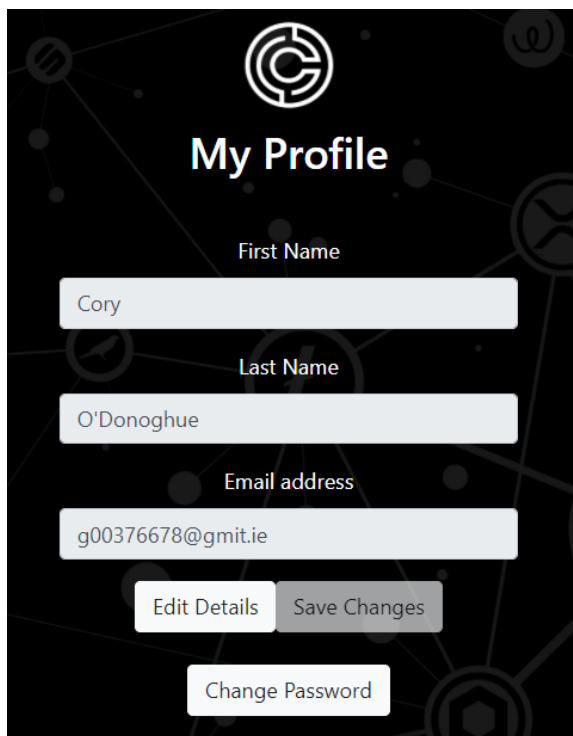
Forward

Reset password: This page is only accessible through a link received via email when user has forgot password

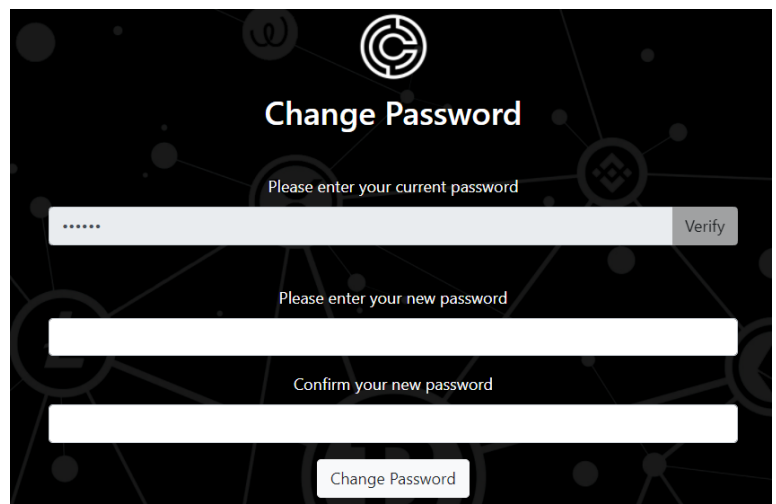


The image shows a 'Reset Password' form on a dark background with a network diagram and a logo at the top. The form has a title 'Reset Password' in white. Below it is a label 'New Password' followed by a text input field containing the placeholder 'Enter new password'. At the bottom of the form is a 'Save' button.

Edit Email/Password: This page allows the user to edit either their email or password. The new details then replace their old details in our database.

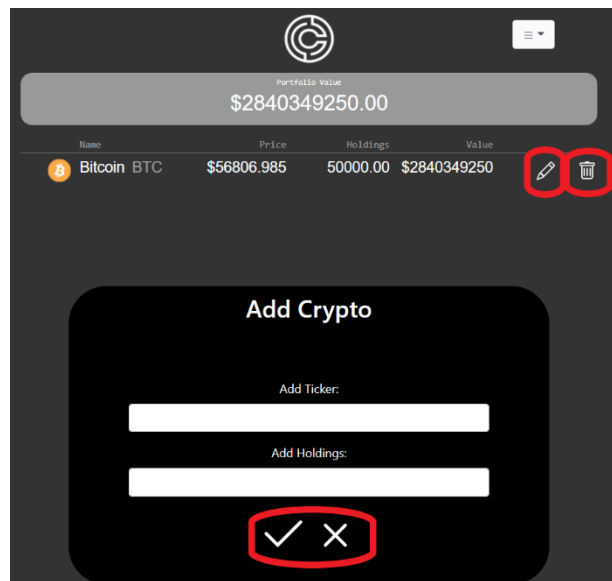


The image shows a 'My Profile' form on a dark background with a network diagram and a logo at the top. The form has a title 'My Profile' in white. Below it are three labels: 'First Name', 'Last Name', and 'Email address'. Each label is followed by a text input field. The 'First Name' field contains 'Cory', the 'Last Name' field contains 'O'Donoghue', and the 'Email address' field contains 'g00376678@gmit.ie'. At the bottom of the form are three buttons: 'Edit Details', 'Save Changes', and 'Change Password'.

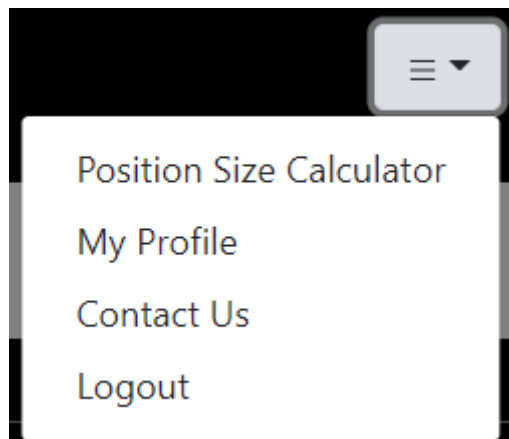


The image shows a 'Change Password' form on a dark background with a network diagram and a logo at the top. The form has a title 'Change Password' in white. Below it are three labels: 'Please enter your current password', 'Please enter your new password', and 'Confirm your new password'. Each label is followed by a text input field. The 'Please enter your current password' field has a 'Verify' button to its right. At the bottom of the form is a 'Change Password' button.

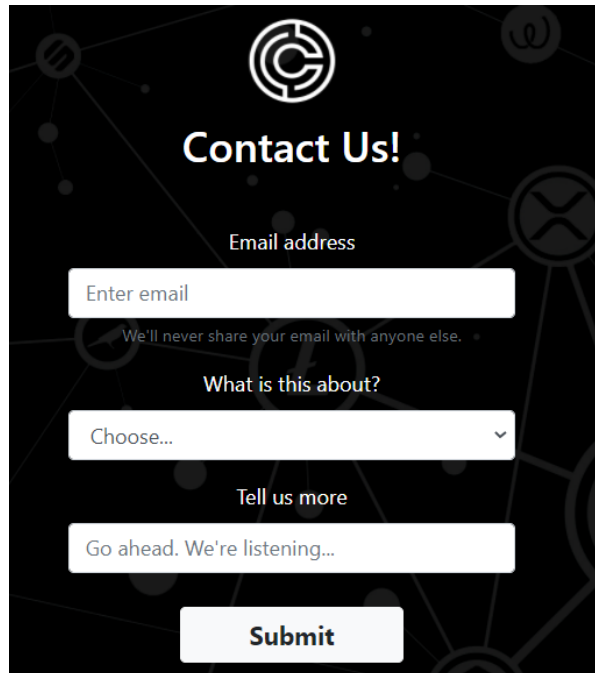
Add/Edit/Delete Cryptos: On the user's portfolio page the user can add/edit or delete cryptos from their portfolio.



Drop Down Menu: This menu allows the user to access their position size calculator, profile, to contact us with queries as well as logout.



Contact us: This page enables the user to get in touch with queries with our support email.

A contact form titled "Contact Us!" with a circular logo at the top. The form includes an "Email address" field with a placeholder "Enter email", a privacy notice "We'll never share your email with anyone else.", a "What is this about?" dropdown menu with a "Choose..." placeholder, a "Tell us more" field with a placeholder "Go ahead. We're listening...", and a "Submit" button at the bottom. The background is dark with faint geometric patterns and icons.

Contact Us!

Email address

Enter email

We'll never share your email with anyone else.

What is this about?

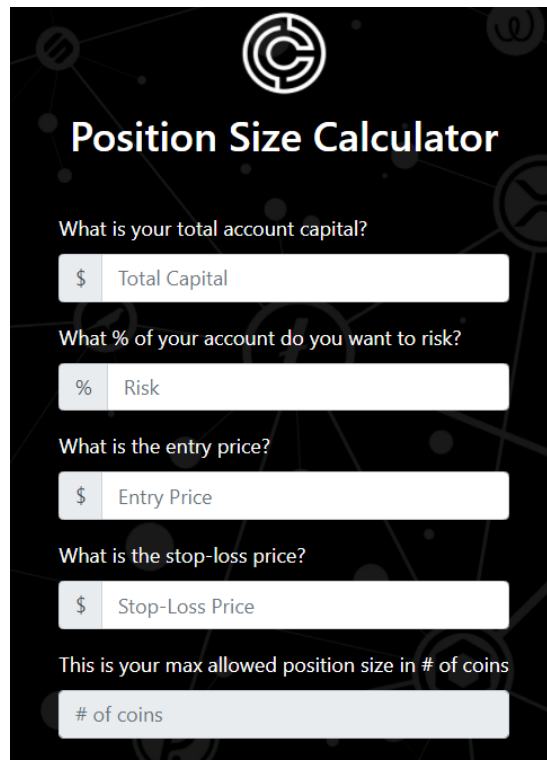
Choose...

Tell us more

Go ahead. We're listening...

Submit

Position size Calculator: This page is for the user to work out the number of coins you wish to risk depending on what you enter as entry price and stop loss price.

A "Position Size Calculator" form with a circular logo at the top. It contains five input fields: "What is your total account capital?" with a "\$" icon and placeholder "Total Capital"; "What % of your account do you want to risk?" with a "%" icon and placeholder "Risk"; "What is the entry price?" with a "\$" icon and placeholder "Entry Price"; "What is the stop-loss price?" with a "\$" icon and placeholder "Stop-Loss Price"; and "This is your max allowed position size in # of coins" with a placeholder "# of coins". The background is dark with faint geometric patterns and icons.

Position Size Calculator

What is your total account capital?

\$ Total Capital

What % of your account do you want to risk?

% Risk

What is the entry price?

\$ Entry Price

What is the stop-loss price?

\$ Stop-Loss Price

This is your max allowed position size in # of coins

of coins

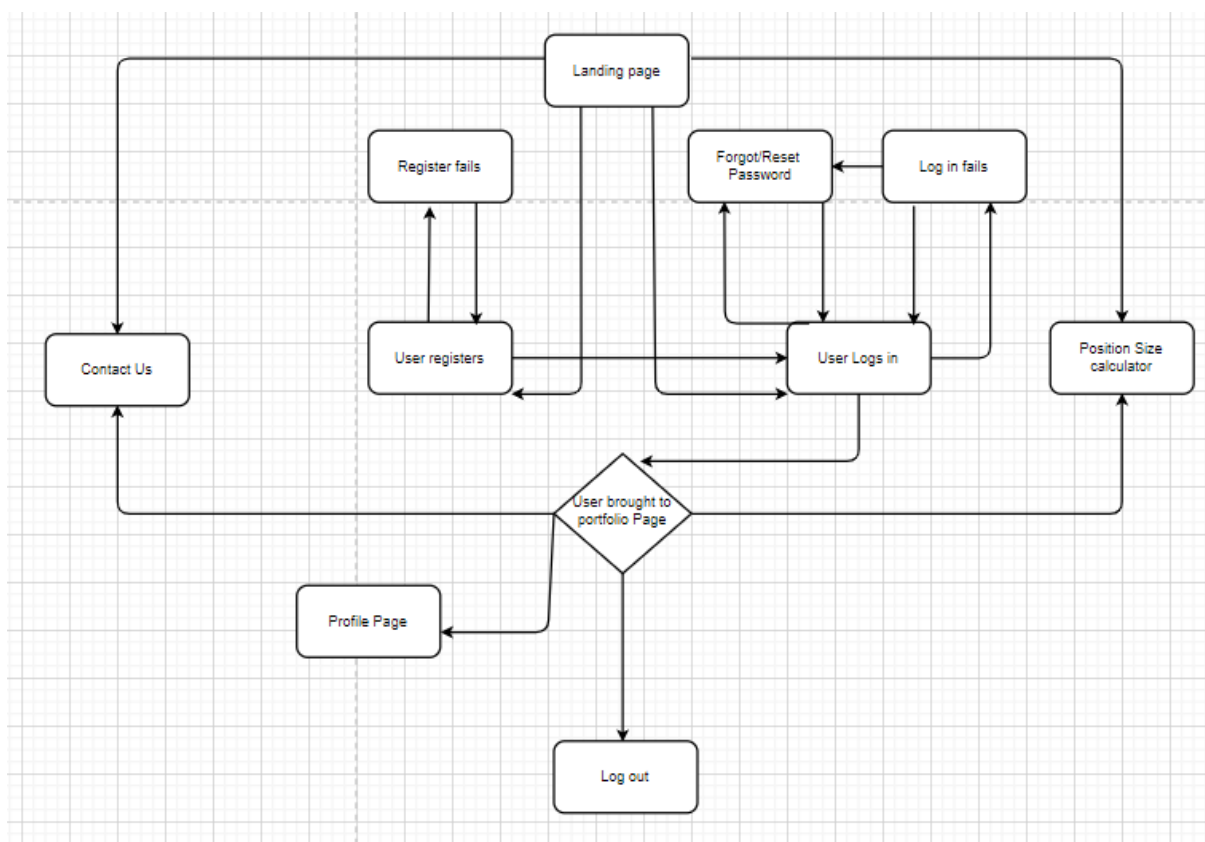
6. Testing

When it came to testing the backend of the application the only real choice was white box testing. This was because there was only two of us on the project and we were both familiar with the internals of the application. We used selenium and its automated testing for the front end.

White Box Testing: To carry out our white box testing we first identified a control graph of the application, see example (a). We soon after wrote test cases to cover every single path, see example (b) for test cases of register. We then covered statements and decision coverage within the code.

Selenium Testing: To carry out our tests on the front-end we used a software called selenium. Using automated testing for the front-end was the obvious choice because it is much faster, not prone to errors and can run multiple tests at once. See example (c) below.

Example (a)- Control Graph



Example (b) – Test cases for register

Requirement ID:	Test case Name	Description	Search parameters / Instructions	Checks	Expected Result	Actual Result	Pass/Fail
1	TC-001	Check all text fields and buttons	1.Enter characters into text fields and	Checks user can input data into text fields and click button to register	UI should all work	UI all works	Pass
2	TC-002	Checks required field by not filling any data	1.Do not enter any data into fields 2.Click register button	Checks user cannot register with invalid data	Error messages should be displayed for all invalid fields	Error messages are displayed to user	Pass
3	TC-003	Required fields	1.Enter valid values into all required fields	Checks user can successfully register	User is registered	User was registered	Pass
4	TC-004	valid password	1.Fill out all fields correctly. 2.Enter 4 characters into password 3.Click register	Checks user cannot register when password is below 5 characters	User cannot register with invalid password	user could not register with invalid password	Pass
5	TC-005	Valid Email	1.Fill out all fields correctly. 2.Enter no @ in the email 3.Click register	Checks user cannot register unless email has valid format	User cannot register with invalid email	User could not register with invalid email	Pass
6	TC-006	Valid First Name	1.Fill out all fields correctly 2.Enter numbers into first name 3.Click register	Checks only letters can be entered into first name field	User cannot register with invalid first name	user could not register with invalid first name	Pass
7	TC-007	Valid Surname	1.Fill out all fields correctly 2.Enter numbers	Checks only letters can be entered into surname field	User cannot register with invalid surname	user could not register with invalid surname	Pass

Example (c)-Selenium Testing

Project: cryptoloo*

Tests + [Run] [Stop] [Refresh] [Clock]

Search tests... Run current test Ctrl+R

Register*	Command	Target	Value
3	click	linkText=Register	
4	click	css=.form-group:nth-child(3) > .form-control	
5	type	css=.form-group:nth-child(3) > .form-control	Cory
6	click	css=.form-group:nth-child(5) > .form-control	
7	type	css=.form-group:nth-child(5) > .form-control	O'Donoghue
8	click	css=.form-group:nth-child(7) > .form-control	
9	type	css=.form-group:nth-child(7) > .form-control	g00376678@gmit.ie
10	click	css=.form-group:nth-child(9) > .form-control	
11	type	css=.form-group:nth-child(9) > .form-control	123456
12	click	css=.btn-lg	

7. Limitations and Bugs

In the duration of this project there were only minor bugs encountered. Right now, in the application there are some limitations such as our API call limit, our supported crypto currencies, and our server.

API call limit: For our crypto currencies we are using a free API called CoinMarketCap. This free API only allows up too 333 calls per day and 10,000 calls per month which at the moment is fine. However now the application has been deployed and may gradually become popular we could soon need to upgrade to a [paid](#) version.

Supported crypto currencies: As of April 2021, there are more than 9,000 cryptocurrencies in existence. As our application does not dynamically add each cryptocurrency and logo, these are added manually so our selection is limited. Therefore, if a user wants to add a currency that we do not currently support they would have to contact customer support desk and request it to be added which will cause delays.

Server: Now the application is deployed if it began to grow exponentially, we would have to upgrade our server and our virtual machine where it is being hosted to keep up with the demand.

One of our testing limitations right now would be the automated testing as they are quite brittle. The tests were successfully carried out, however they can only be running in a specific order and with specific details. If we were to keep working on the project a more thorough way would need to be used in the automated testing.

8. Improvements in the future

This application has countless possible improvements that could be made in the future if we were to keep working on it. Some of the largest improvements would be the addition of charts, percentage gains regarding your portfolio or different currency fluctuations, add separate calculators, user transaction history and add trading.

Charts: We could add two types of charts.

Chart one would display your how much your portfolio has developed over the course of using our application, it would show percentages of how much your portfolio is up or down in the last 24 hours, week, or month. This would be a line chart.

Chart two would be for each currency which would show live prices of all cryptocurrencies and access their extensive charts and analytical data. This would be a line chart when on the portfolio page but once you click the crypto it brings you to a page dedicated to that cryptocurrency with a much more detailed historical chart which would be a candlestick chart.

Calculators: Currently we only have one calculator which is our position size, but we could also add an exchange rate calculator where you could convert 100EUR to Bitcoin for example.

User Tracking: This implementation would be so the user could review their transaction history where it shows how much money they have invested in total, how much profit they are up, how much losses they have. It could also show what other currencies they have previously held and for how long.

Trading: The largest improvement would be enabling the user to buy coins from the page with your debit card, PayPal or apple pay.

Currency: Cryptolio currently only supports the USD. In the future I would like to add an option for the user to select their currency e.g EUR, CAD, CHF, JPY. This would then allow the user to see their portfolio value and cryptocurrency prices in their local currency.

Theme: This application has been created with minimalistic dark theme. It would be nice to have the ability to switch to a light theme for those who would prefer a lighter theme to a dark theme.

9. Conclusion

In conclusion we feel we achieved all the main objectives that were set out at the beginning of the project. These objectives included having a working login and register system with the ability to reset your password if it was forgotten, a portfolio page which allows the user to view their portfolio along with the ability to add/edit/delete cryptos from the portfolio, set up communication between the user and the team here at cryptolio via a contact us page, the ability to edit your credentials such as your name, email or password, a position size calculator to allow users to practice risk management, a database which stores user and portfolio details and a backend which receives requests from the client and contains the logic to send the appropriate data back to the client. The completion of this project has provided a client/server application for a cryptocurrency portfolio tracking system with a built-in position size calculator. This application helps fill a gap in the ever-expanding crypto market where users can keep track of all their cryptocurrencies in the one place while also having the ability to practice proper risk management by using the calculator built into our platform. Taking on a much larger scale project without specification was immensely beneficial as it enabled us to push our self-learning, discover new tools and technologies, while improving our teamwork and communication.