University of Glasgow

School of Computing Science

Internet Technology Implementation Report

2551764X Yingyu Xu

2870692Y Xulong Yang

2847999Z Yiming Zhou

2888251W Zhaoxu Wang

2752052Y Mai Yang

1. Introduction:

The BookBalance is an innovative online platform that streamlines and simplifies money management for individuals and businesses alike. BookBalance's attractive user interface and comprehensive features provide users with a set of tools for managing bookkeeping, visualizing financial data, and easily accessing relevant information.

When visitors visit the BookBalance website, they are met with a streamlined registration procedure that assures account security and validity. Users gain access to their accounts after completing email verification, where they may make use of a variety of services suited to their requirements and positions.

BookBalance's basic functions include four major areas: account management, data visualization, budgeting, and interactive support. Users may create, query, amend, and remove accounting records, resulting in a centralized repository for financial transactions and history. Furthermore, BookBalance provides powerful visualization features, allowing users to obtain insights into their spending patterns and financial trends via interactive charts and graphs. The Budgeting tool allows users to create customized budgets for various periods, allowing them to manage resources more efficiently and meet financial objectives. Furthermore, the Interactive Support function includes a ChatBox interface that allows users to receive automatic solutions to typical finance-related questions, hence increasing user engagement and accessibility.

The overarching goal of BookBalance is to empower users with the tools and insights needed to make informed financial decisions, ultimately fostering greater financial literacy and well-being. By providing a user-friendly platform for managing finances, BookBalance aims to simplify the complexities of financial management and promote financial wellness for individuals and organizations alike.

Overall, BookBalance represents a humble effort to provide users with a practical solution for navigating their financial landscape with confidence and ease.

Public URL of web app: **https://ddl123.pythonanywhere.com/**

2. Design Specifications:

2.1. Overview of Implemented Application

The integrated BookBalance programme is a complete online platform that simplifies accounting and financial management for individuals and businesses. It includes user registration and log in authentication, budget management, accounting functions, data visualisation, ChatBox interactivity, and search capabilities. BookBalance's user-friendly interface and adaptable design enable users to take greater control of their finances, make educated decisions, and achieve their financial objectives more effectively.

2.2. Requirements

1. Users must register with a valid email address and the username cannot be the same with others.
2. Upon verification, users gain access to their accounts, where they can log in securely.
3. Users can customize budgets for different time frames, enabling them to track their expenses and allocate funds effectively.
4. Users can create, query, update, and delete accounting records, providing them with a comprehensive toolset for managing financial transactions.
5. BookBalance offers visual representations of billing records, allowing users to gain insights into their financial data through charts and graphs.
6. Users can engage with a chatbot to receive automated responses to common finance-related questions. In this function, users only need to input some keywords to get the response.
7. Users can search for relevant accounting records through input different date, facilitating quick and efficient access to specific financial information.
8. The application utilizes a responsive CSS framework to ensure optimal viewing and interaction across various devices and screen sizes.
9. BookBalance features a polished and refined interface, ensuring a seamless user experience that is both intuitive and visually appealing.
10. The application implements robust security measures, including secure login procedures, to safeguard user data and privacy.

2.3. System architecture diagram

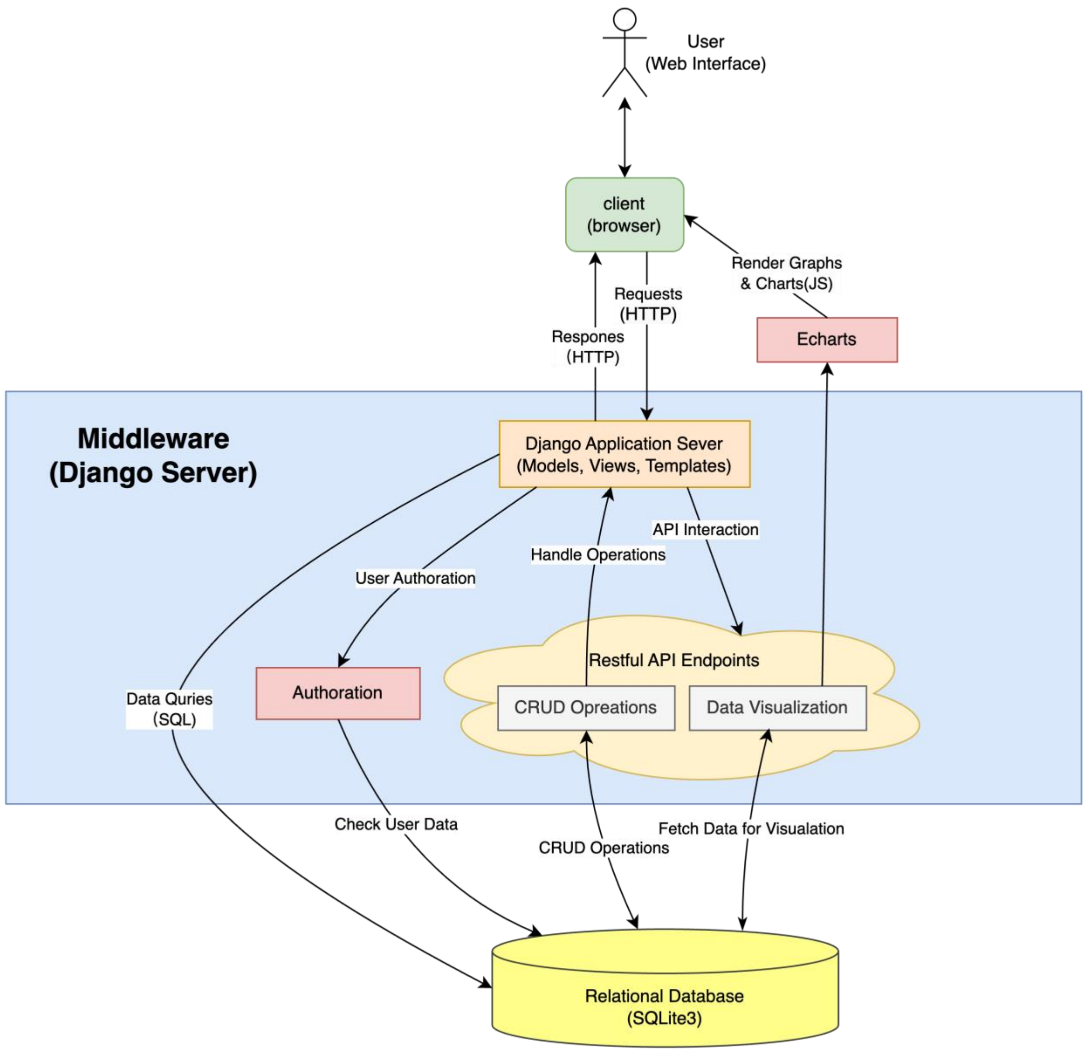


Fig 1 system architecture diagram

This system architecture outlines a sophisticated Django-based web application tailored for accounting, using echarts for enhanced data representation. The client-side, a browser, initiates communication via an HTTP request to the server-side middleware—Django Application Server—that orchestrates operations through its MVT framework, handling user authentication with database, and managing data transactions via RESTful API endpoints. These endpoints facilitate CRUD operations with an SQLite3 database and leverage JavaScript for rendering interactive financial charts and graphs in the browser, using JSON-formatted data processed through echarts.

2.4. ER diagram

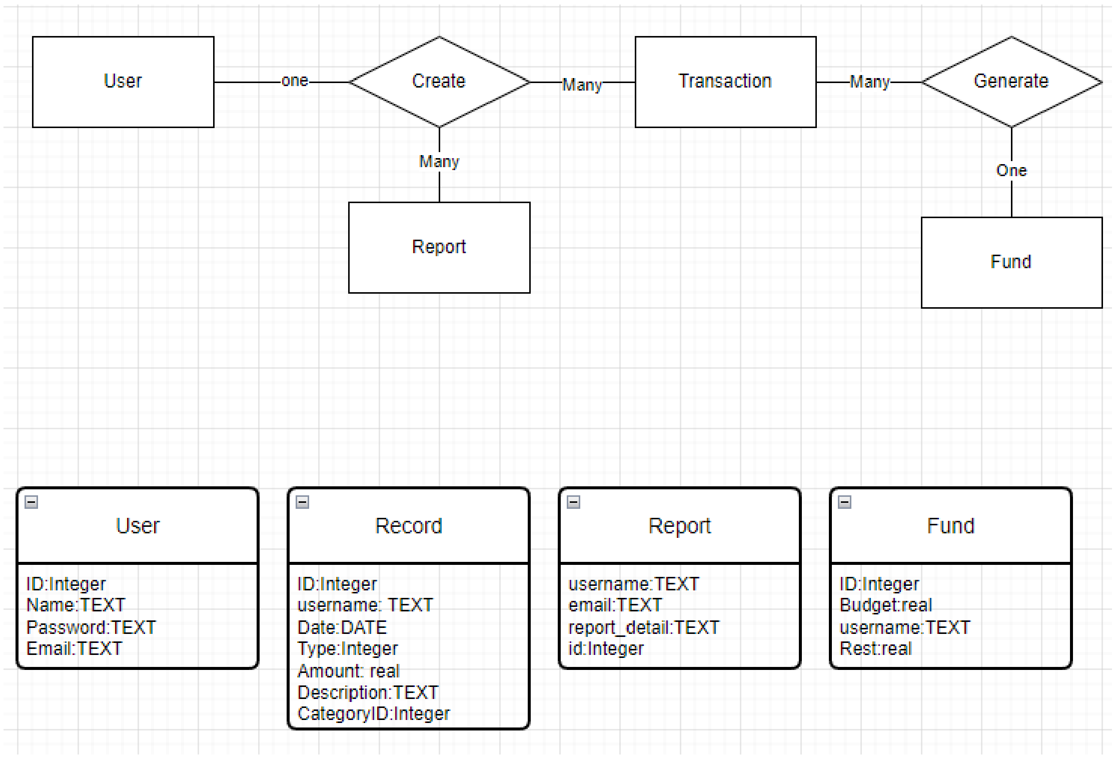


Fig 2 ER diagram

In the ER diagram, our team made minor modifications to the previous design based on the specific implementation of different functions. During the specific implementation process, we finally chose to use username to connect different database tables instead of user_id. At the same time, for the types of Record, we chose to use enumeration to classify them without designing a separate database table.

2.5. Site Map

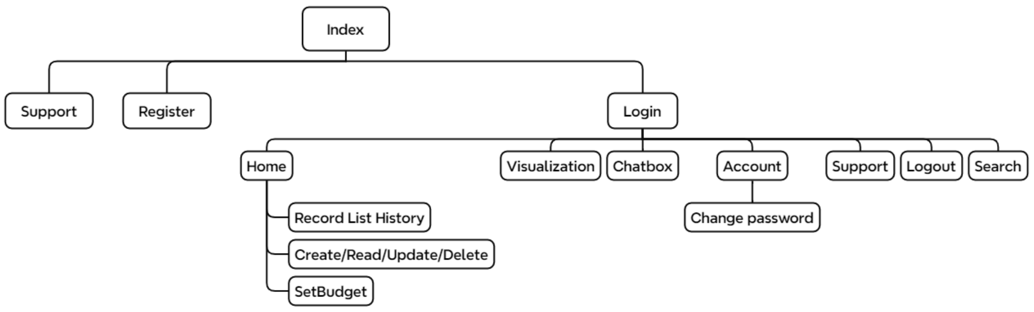


Fig 3 Site Map

The updated sitemap emphasizes the distinction between pre- and post-login access. The Home section, post-login, serves as a financial hub with tools for transaction management and budgeting. Visualization provides graphical data analysis, and the Chatbox offers an interactive chatbot service. Under Account, users can change passwords. The website also features dedicated Support and efficient Search functions, alongside a Logout option, ensuring a seamless and secure user journey within BookBalance.

2.6. Wireframes

Here from fig 4 to 17 lists several wireframes to show the basic functions and layout of our web application.

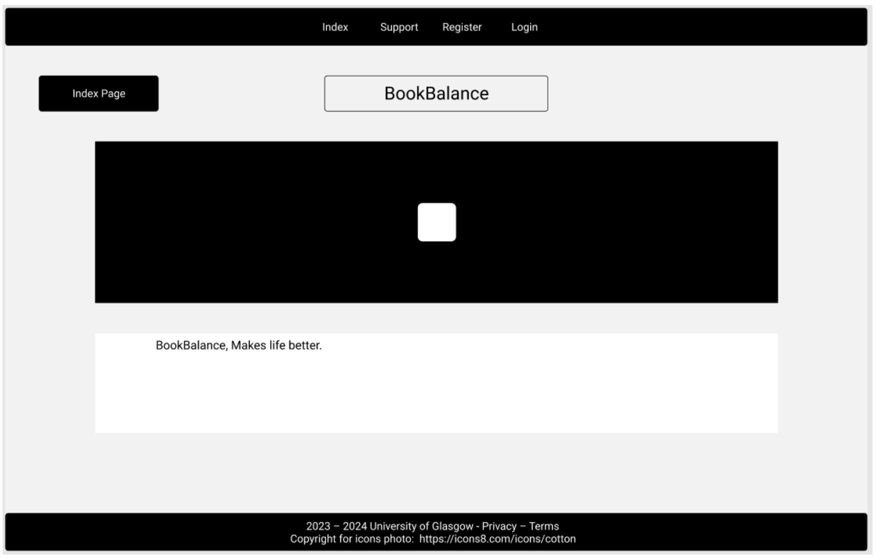

Index
Support
Register
Login
Index Page
BookBalance
BookBalance, Makes life better.
2023 – 2024 University of Glasgow - Privacy – Terms
Copyright for icons photo: https://icons8.com/icons/cotton

Fig 4 Index Landing Page

When the user does not log in, he can see a landing page. Figure 4 shows the landing page (index home page) when the user is not logging in. This page has four links to access through the navigation bar, which are 'index' (this page), 'support', 'register', and 'login'.

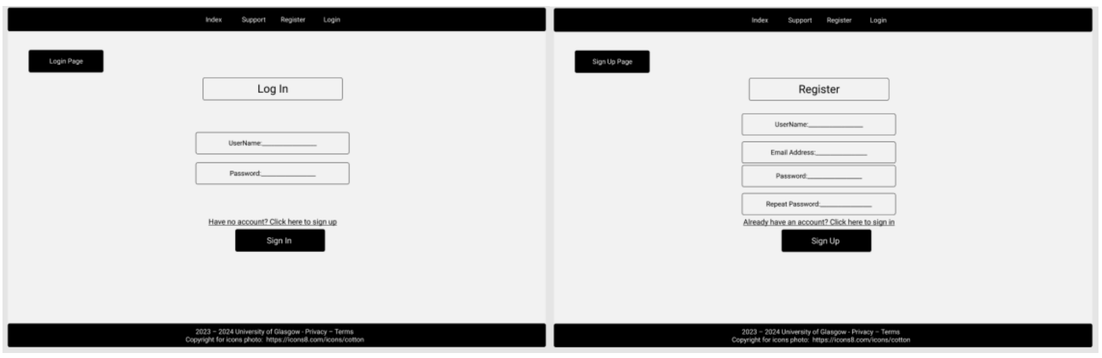

Index
Support
Register
Login
Login Page
Log In
UserName:
Password:
Have no account? Click here to sign up
Sign In
Sign Up Page
Register
UserName:
Email Address:
Password:
Repeat Password:
Already have an account? Click here to sign in
Sign Up
2023 – 2024 University of Glasgow - Privacy – Terms
Copyright for icons photo: https://icons8.com/icons/cotton

Fig 5&6 Log in and Register Page

Figure 5&6 show the login page and the register pages. These two pages can redirect to each other. Figure 7 shows the support page when user is not logged in.

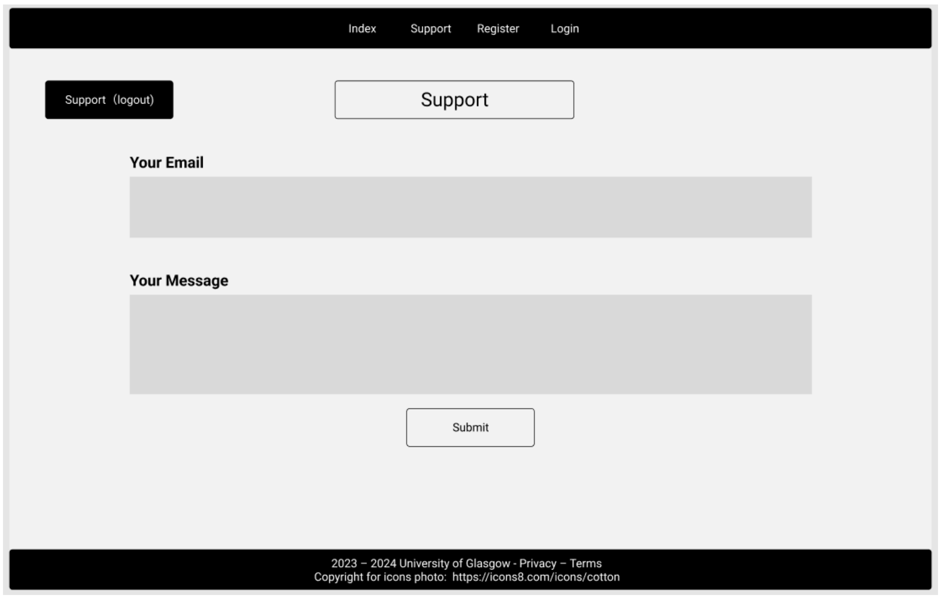

Index
Support
Register
Login
Support (logout)
Support
Your Email
Your Message
Submit
2023 – 2024 University of Glasgow - Privacy – Terms
Copyright for icons photo: https://icons8.com/icons/cotton

Fig 7 Support Page (not log in)

Figure 5&6 show the login page and the register pages. These two pages can redirect to each other.

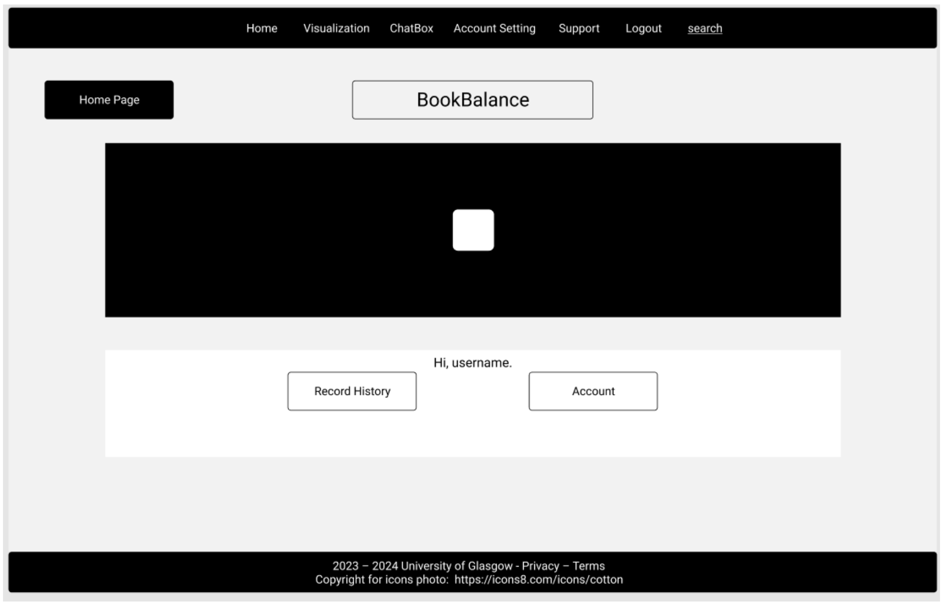
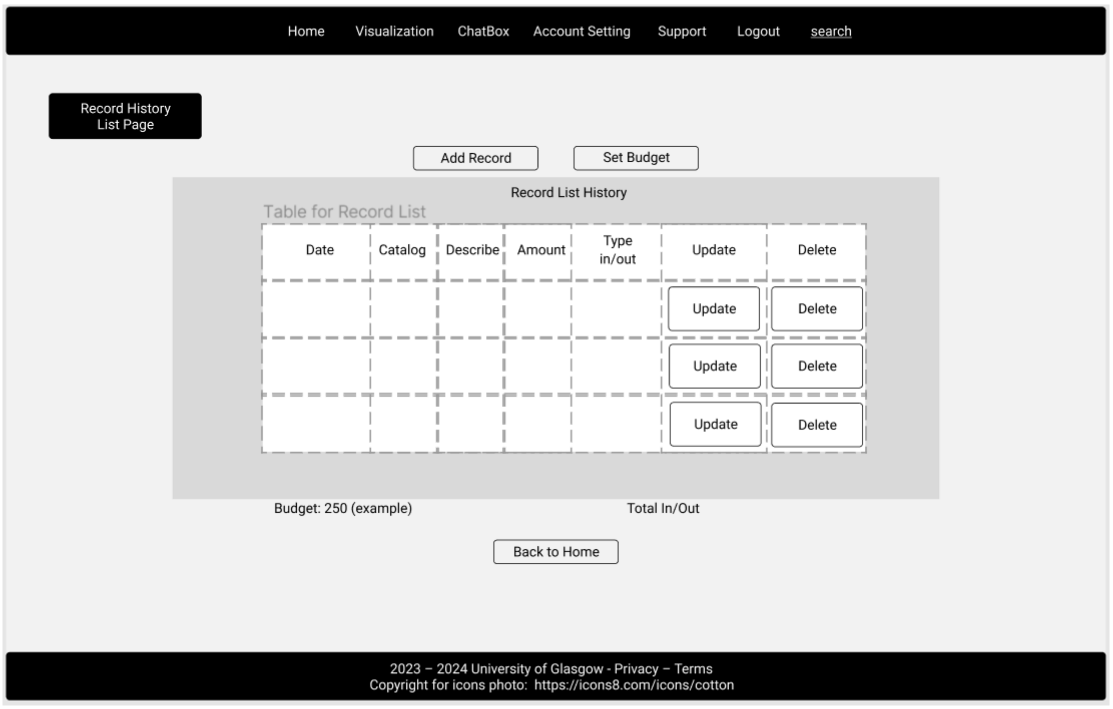


Fig 8 Home Page (log in)

When user login, they can see the home page (figure 8) insight, including their username, and two links to their record history and account settings.



Date	Catalog	Describe	Amount	Type in/out	Update	Delete
					Update	Delete
					Update	Delete
					Update	Delete

Fig 9 Record History List Page

After clicking the 'record history' button, the user can see the record history list page shown as figure 9. In this page, there are two buttons at the top, 'Add record' and 'Set Budget', which redirect to the corresponding pages. The user can see a table showing all their records. At the bottom of the table, here is listed the budget user set, the total in and out amount of the records. For each record (each line), there are 'Update' and 'Delete' buttons. 'Update' button redirects to the 'Record-edit Page', while 'Delete' button realized the function that deletes this certain record.

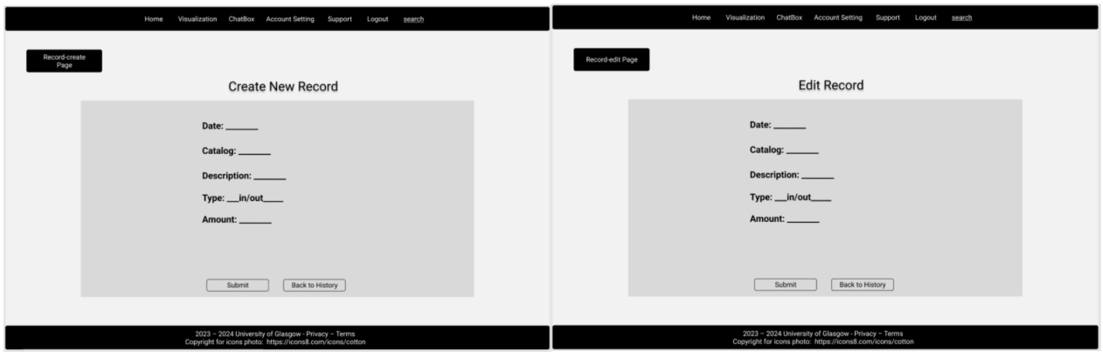


Fig 10&11 Record-create Page and Record-edit Page

Figure 10 shows the page to create a new record. Users can submit the information for a record or back to the previous history page. Figure 11 shows the page to edit an existing record. These two pages are in the same style.

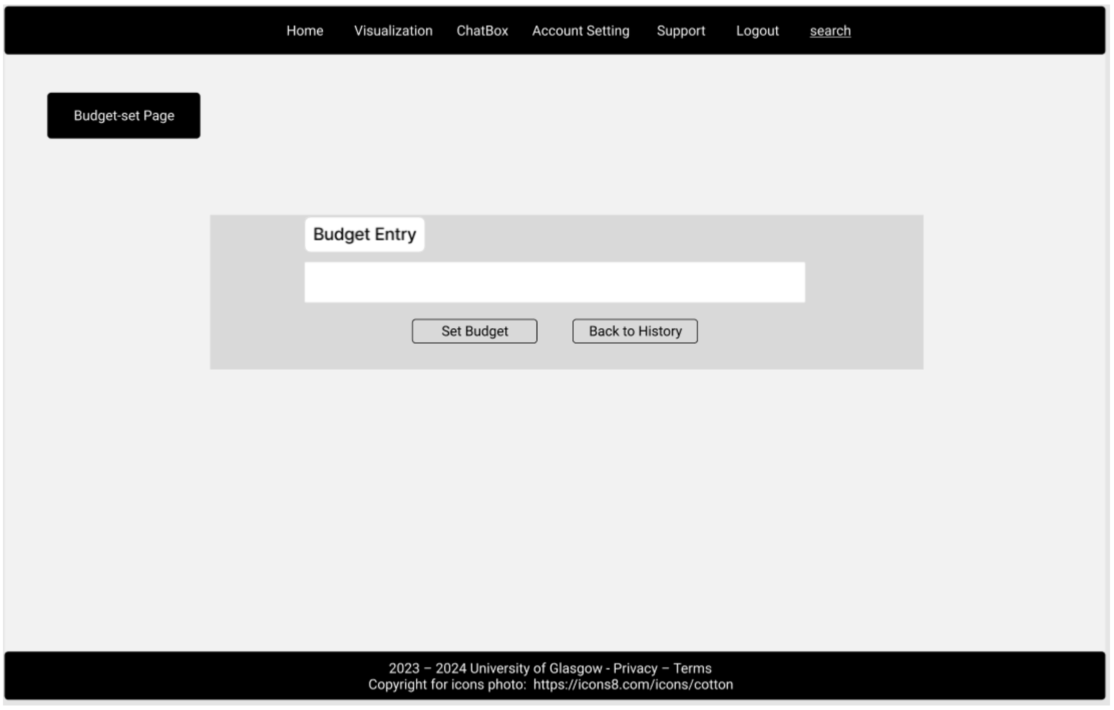


Fig 12 Budget-set Page

Figure 12 shows the page after the user clicks the 'set budget' button from the 'Record History List Page'. Here users can set their budget. They can click 'Back' button back to history page.

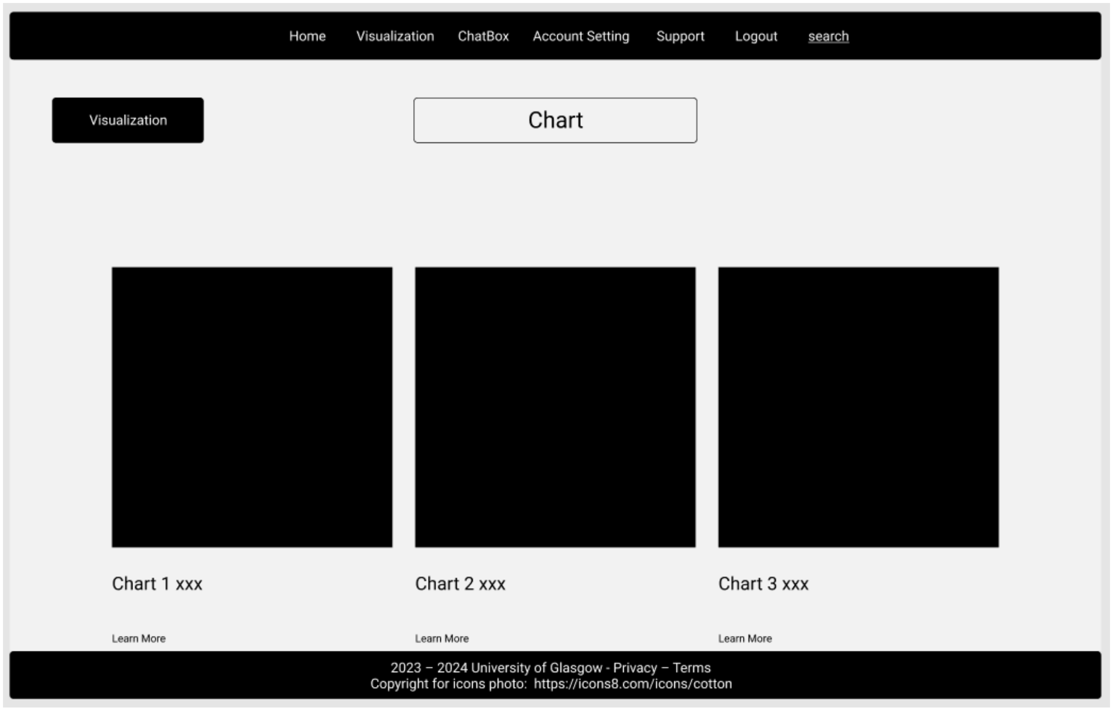


Fig 13 Visualization Page

User can click on 'Visualization' link on the navigation above to the 'Chat Box Page' shown in figure 13. This page shows the records data visually with several dynamic charts. People can choose a year and month to change the outcome charts.

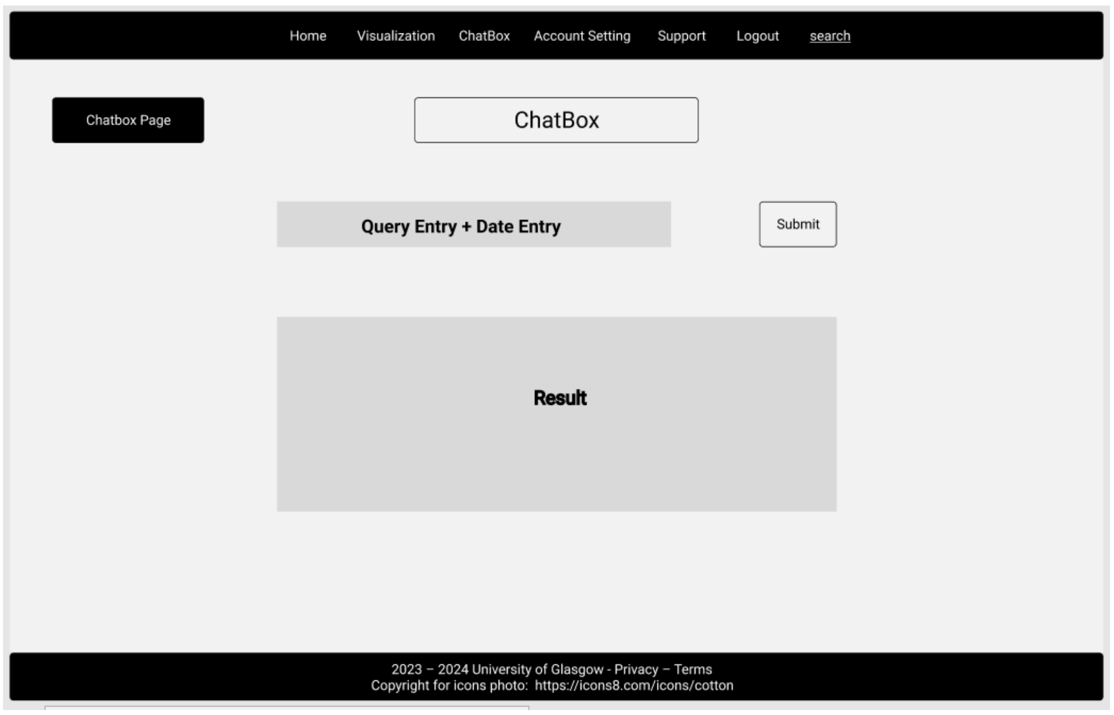


Fig 14 Chat Box Page

User can click on 'ChatBox' link on the navigation above to the 'Chat Box Page' shown in figure 14. Here by entering the query keyword and date (optional), users can get the results of relative records as a table. For example, if type 'income' as the query entry, the results will get the list of records that are all type of 'In'.

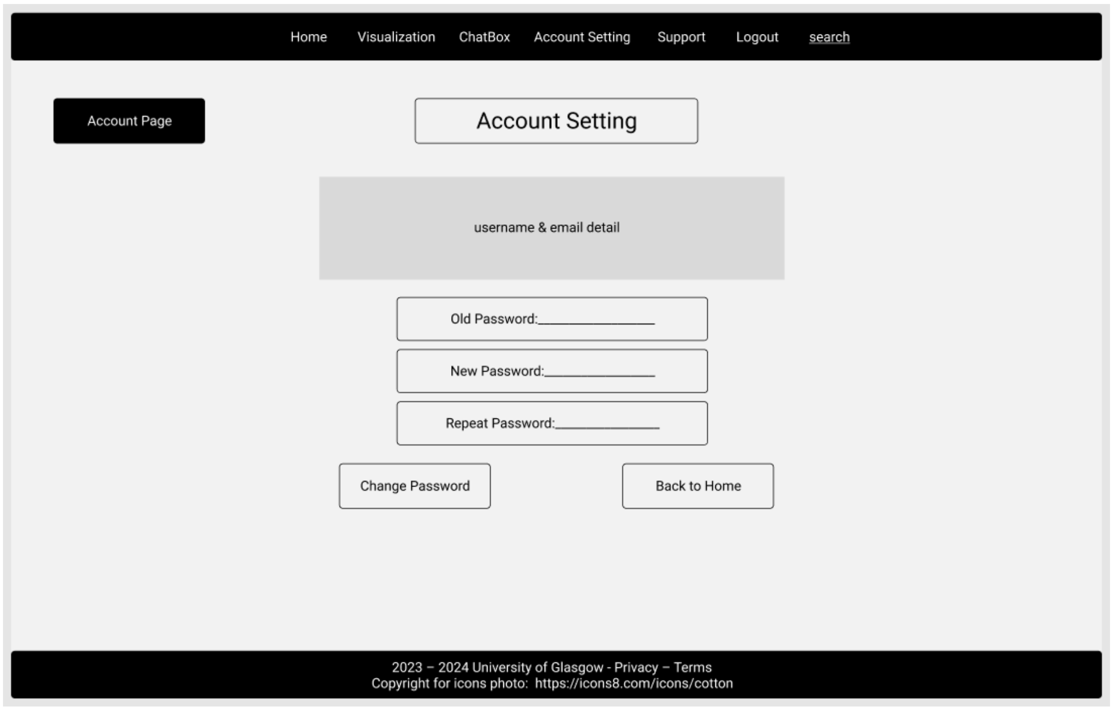

Home
Visualization
ChatBox
Account Setting
Support
Logout
search
Account Page
Account Setting
username & email detail
Old Password:_______________
New Password:_______________
Repeat Password:_______________
Change Password
Back to Home
2023 – 2024 University of Glasgow - Privacy – Terms
Copyright for icons photo: https://icons8.com/icons/cotton

Fig 15 Account Setting Page

Figure 15 shows the account setting page. On this page, users can change their password. Their username and email address information are listed on this page as well. Note that if the input old password is wrong or the repeat password is not equal to the new password, there will be an error message at the bottom of the page; while if successfully changed the password, a success message will be displayed. Also, this page can be accessed from the home page with the ‘account’ button.

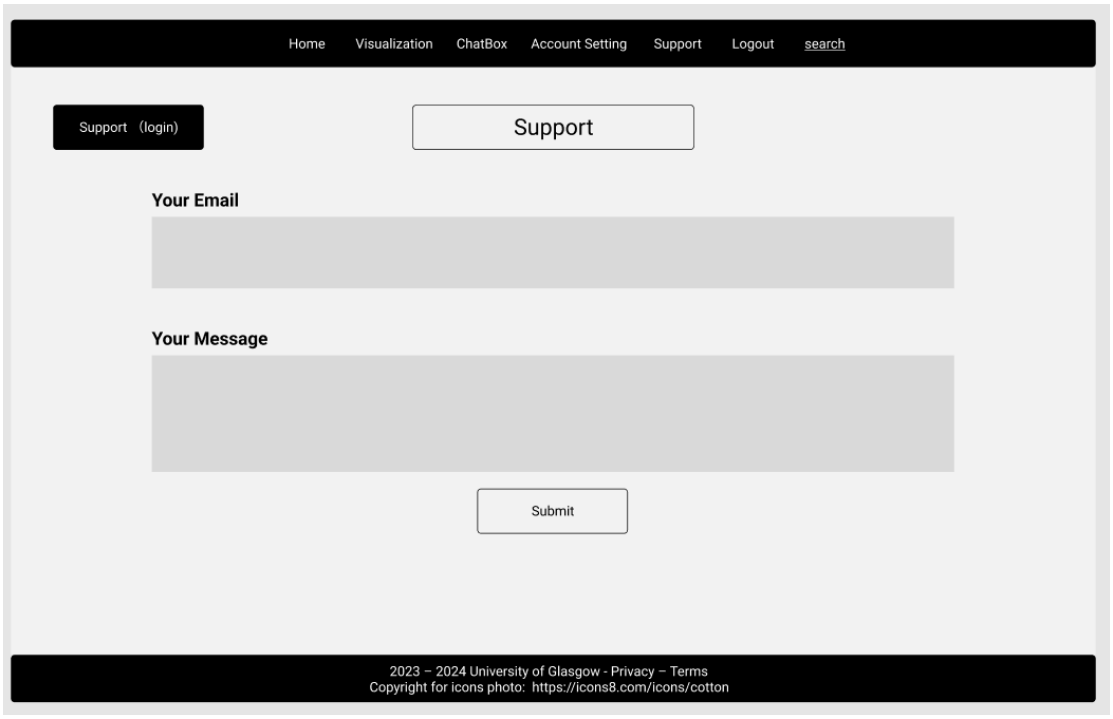

Home
Visualization
ChatBox
Account Setting
Support
Logout
search
Support (login)
Support
Your Email
Your Message
Submit
2023 – 2024 University of Glasgow - Privacy – Terms
Copyright for icons photo: https://icons8.com/icons/cotton

Fig 16 Support Page (Log in)

Figure 16 shows the support page when users are logged in. The function provided is the same as the support page when users are not logged in. Once submitted, it will redirect to the record history page.

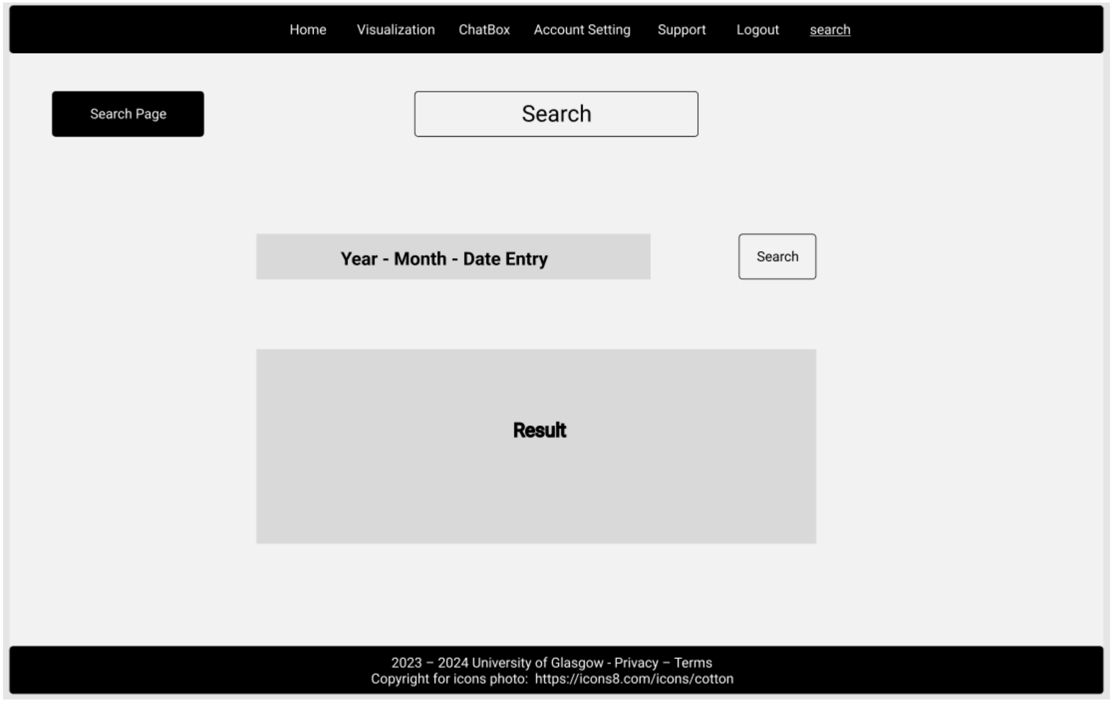

Home
Visualization
ChatBox
Account Setting
Support
Logout
search
Search Page
Search
Year - Month - Date Entry
Search
Result
2023 – 2024 University of Glasgow - Privacy – Terms
Copyright for icons photo: https://icons8.com/icons/cotton

Fig 17 Search Page

Figure 17 shows the search function for the records on the certain date of the user. Users can enter the exact date by 'Year-month-date Entry'.

3. Description of Implementation

The implementation of the BookBalance application involved the development of several key components, each contributing to its functionality, look and feel, and underlying code structure. Below is a detailed description of these components:

User Authentication and Management:
The user registration and login functionalities were developed using Django's built-in function to check the password. And the login function utilises cookies and sessions to make user login directly in the second time. Considering the security problem, passwords would not be remembered.
Upon registration, the password is encrypted by the Django built-in functions.
We use 'login_view' and 'register' to realize the user authentication. Notes that when the user logs in, "*response.set_cookie*" provides the user to remember the password and username for next login. 'logout' provprovides the function to allow the user to log out and delete any cookie using the "*response.delete_cookie*".

Budget Management:
The budget management function allows users to establish, customise, and track budgets over a variety of periods.
Django models were used to set their budget.
A straightforward interface allows users to monitor and change their budgets, giving them insight into their spending patterns and financial goals.
'set_budget' set the 'fund' object from the database by updating its 'budget' attribute through "F*und.objects.filter(username=username).update(Budget=budget, Rest=rest)*", while if the current fund object is not existing, we will create a new fund object with the provided budget value getting from the 'set_budget.html' page.

Accounting Functionality:
The CRUD procedures for accounting records were built using Django's model-view-template (MVT) architecture.
Users may easily create, query, edit, and remove accounting records using the user interface.
Input validation and error handling procedures were put in place to maintain data integrity and prevent unauthorised access.
'record_create' and 'record_update' uses the function that is provided by Django MVT, which is the function "*RecordForm(request.POST)*", to create or update a new

‘Record’ into the database from getting the information from the front-end. While ‘record_update’ and ‘record_delete’ both transfer the ‘pk’ value, i.e., recording to the current record id, into URL, so that the certain record can be operated.
‘record_list’ retrieves the Record information by filtering with username and returning the relevant data to the ‘record_list.html’.

Data Visualisation:
The data visualization page uses JavaScript and Echarts, which allows users to view detailed records bar charts and consumption category statistical pie charts for specified months and years at a custom time and makes them to properly analyse their financial data.

The backend query is implemented in views.py. Using ajax call to interact with the server asynchronously, each time the corresponding tab is opened, the backend data is requested, and the data of the chart is updated to make it displayed.

As to the specific implementation, we query data from the database in the backend code using Django, convert it to JSON format, and define the chart's location in the HTML. Some AJAX calls are made to the backend to update the data in the chart, and finally, Echarts is used to generate a visual chart for display on the page.

Regarding the functions in views.py, ‘retrieve_year_has_data’ and ‘retrieve_month_has_data’ are used to query the years for which data is available, to aid in frontend filtering. The functions ‘retrieve_current_year_income_expense’ and ‘retrieve_current_month_income_expense’ are utilized to query and consolidate the account records for different years, months, types, and categories for a user, amalgamating them into the JSON response needed by the frontend.

Chatbox Integration:
A chatbot was added to the application to give users automatic solutions to typical finance-related questions. The chatbot would analyse the user inputs and find the keywords to generate the result. Moreover, user could also input the date to find the records they want.
‘chatbox’ function using “*request.POST.get*” to get front-end ‘query’ and ‘date’ information, extract the query keyword and filter the Record by these two constrictions and username. For instance, if the input query is ‘most income record’, we use “*Record.objects.filter(Q, username=username, type=0).order_by("-Amount").first()*” to filter the corresponding results, where Q refers to the SQL query conditions provided by Django's package.

Search functionality:

Users may search for individual accounting records using date.
In this function, user need to select the date of the Record. The input would be regarded as the filter condition in the backend. The result would be shown directly in the web page.
‘search’ uses SQL filter funtion “m*odels.Record.objects.filter(username=username, Date=date).order_by("Date")*” to get the corresponding record by getting the ‘date’ information from “*request.POST.get('date')*”.

Front-end CSS and JavaScript:
We use ‘static/css/style.css’ as our whole website CSS style. However, we also use ‘bootstrap-5.3.1’ bootstrap (*https://github.com/twbs/bootstrap/archive/v5.3.3.zip*) in the meantime. For the pages when users are not logged in, we use ‘‘static/css/style-login.css’ for the styling. Also, for some pages, we use ‘static/js/base.js’ as our js implementation. The visualization part uses the technique of Ajax to provide the ‘echarts’ dynamically.

Code structure and readability:
The application adheres to Django's best practices for project organisation, with distinct modules for models, views, templates, and static files.
The code is well-commented and follows PEP 8 guidelines for readability and uniformity.
Unit tests were created using Django's testing framework to validate code quality and functioning.

Overall, the BookBalance application's development displays a seamless integration of multiple components, resulting in a user-friendly and feature-rich financial management solution. From user identification and budget management to data visualisation and chatbot integration, each component adds to the application's functionality, look and feel, and underlying codebase, giving users a simple and easy way to manage their funds.

4. System testing

The BookBalance application was rigorously tested for functionality, usability, speed, and security. The following are the important findings from the system testing:

Functional Testing:
All essential capabilities, such as user registration, budget management, accounting operations, data visualisation, chatbot interaction, and search functionality, underwent extensive testing.
Test cases were created to ensure the validity of CRUD operations, budget

customisation, data visualisation accuracy, chatbot replies, and search result relevance.
The programme passed all functionality tests, exhibiting robustness and dependability in dealing with varied user interactions and circumstances.

Usability Testing:
Usability testing evaluated the application's user interface design, navigation flow, and overall user experience.
Testers provided feedback on the clarity of instructions, intuitiveness of controls, and ease of access to various functionalities.
Users gave the programme favourable comments, praising its straightforward interface, responsive design, and user-friendly interactions.

Performance Testing:
Performance testing was done to determine the application's responsiveness, load times, and scalability under various usage circumstances.
Load testing was carried out to mimic concurrent user interactions and evaluate the application's response time and resource utilisation.
The programme performed satisfactorily, with low latency and constant response times even under heavy load situations.

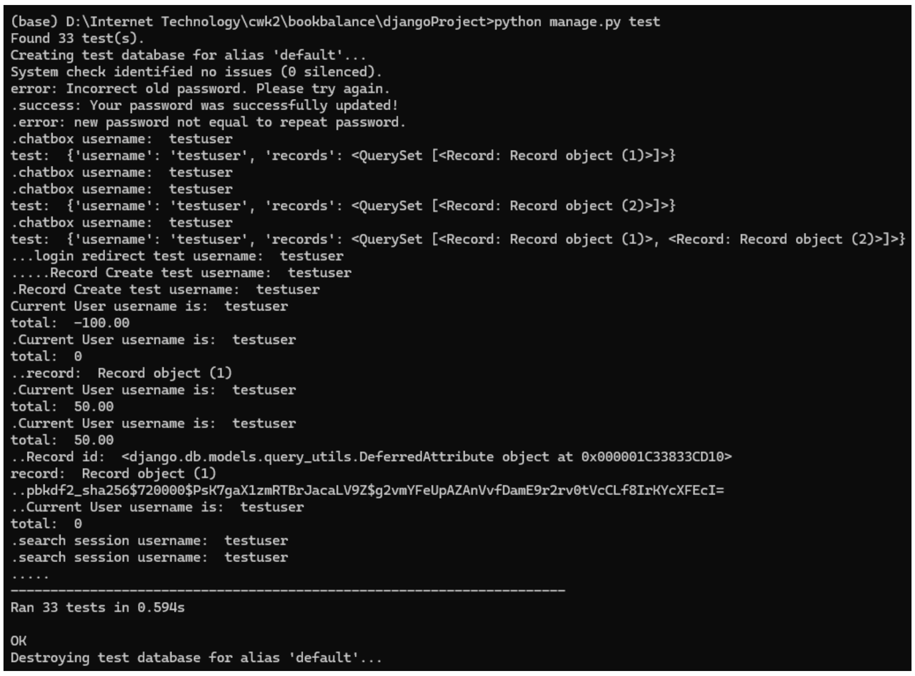
Security Testing:
Security testing was designed to detect and resolve possible vulnerabilities such as cross-site scripting (XSS), SQL injection, and session management problems.
Penetration testing tools were used to simulate attacks and determine the application's resilience to typical security risks.
The programme passed security testing, with no serious vulnerabilities found. To protect user data and prevent unauthorised access, necessary security measures were established, including input validation, data encryption, and access control.

Overall, system testing of the BookBalance application revealed excellent findings, confirming its functionality, usability, performance, and security. The application's stability and dependability make it suited for real-world deployment, giving users a safe and efficient platform for managing their cash.

System testing:
In this part,33 tests are performed. See the appendices for test results.

5. Appendices

System testing results



```
(base) D:\Internet Technology\cwk2\bookbalance\djangoProject>python manage.py test
Found 33 test(s).
Creating test database for alias 'default'...
System check identified no issues (0 silenced).
error: Incorrect old password. Please try again.
.success: Your password was successfully updated!
.error: new password not equal to repeat password.
.chatbox username:  testuser
test:  {'username': 'testuser', 'records': <QuerySet [<Record: Record object (1)>]>}
.chatbox username:  testuser
.chatbox username:  testuser
test:  {'username': 'testuser', 'records': <QuerySet [<Record: Record object (2)>]>}
.chatbox username:  testuser
test:  {'username': 'testuser', 'records': <QuerySet [<Record: Record object (1)>, <Record: Record object (2)>]>}
...login redirect test username:  testuser
.....Record Create test username:  testuser
.Record Create test username:  testuser
Current User username is:  testuser
total:  -100.00
.Current User username is:  testuser
total:  0
..record:  Record object (1)
.Current User username is:  testuser
total:  50.00
.Current User username is:  testuser
total:  50.00
..Record id:  <django.db.models.query_utils.DeferredAttribute object at 0x000001C33833CD10>
record:  Record object (1)
..pbkdf2_sha256$720000$PsK7gaX1zmRTBrJacaLV9Z$g2vmYFeUpAZAnVvfDamE9r2rv0tVcCLf8IrKYcXFEcI=
..Current User username is:  testuser
total:  0
.search session username:  testuser
.search session username:  testuser
.....
----------------------------------------------------------------------
Ran 33 tests in 0.594s

OK
Destroying test database for alias 'default'...
```

Fig 18 System testing results

Contribution

Name	GUID	Contribution (in %)	Responsible part(s)
Yingyu Xu	2551764X	20%	The foundational website setup, CSS and HTML design, CRUD functions of records and record history list implementation.
Xulong Yang	2870692Y	20%	Complete lots of the basic functions and the chatbot function. Deploy the web App.
Yiming Zhou	2847999Z	20%	Use ajax, js, and echarts to complete the data visualization part. Writing some unit tests.
Zhaoxu Wang	2888251W	20%	The final report.
Mai Yang	2752052Y	20%	The front end of the support page.

Image Copyright Acknowledgement

The icons utilized within the BookBalance platform have been sourced from Icons8 (*https://icons8.com/icons/cotton*), under the specific collection titled "Cotton." We extend our gratitude to Icons8 for providing high-quality visual assets that enhance the user interface of our platform.

We acknowledge and respect the copyright and usage terms set by Icons8. The incorporation of these icons into the BookBalance platform is in accordance with the terms provided by Icons8 and under the appropriate licenses.