Course: Database and Software Technology 2

Part: Web Technology

Roll number: 2074

ICA Project Reflective Report

Project name: LCGenome

Role: Developer

Date: 2024/5/1

LCGenome
ZJU-UOE INSTITUTION

Catalogue

LCGenome: A User-friendly Website for Lung Cancer Genetic Counseling and Drug Targeting

Developer reflective reporting

1 Abstract

Addressing the critical challenges to human health posed by lung cancer, our team has created LCGenome, a user-friendly platform designed for the storage and retrieval of lung cancer genes and targeted therapy information. The project's source code has been made publicly accessible on GitHub, with my contributions as both a developer and team leader. Throughout the development process, I have adhered to Huawei's guidelines for web development and engaged in close collaboration with team members across different phases. After continuous development and multiple iterations, we have successfully realized the website's construction.

2 Introduction

Lung cancer is the deadliest cancer in the world, posing a major threat to human health (Thai *et al.*, 2021). With the deepening of medical research, the key role of genetic factors in the occurrence and development of lung cancer has become increasingly clear. Gene mutation plays an important role in the occurrence and development of lung cancer (Thai *et al.*, 2021). Targeted therapy, as a widely used treatment, has the characteristics of precision, effectiveness, and relatively few side effects, bringing new hope to patients (Lahiri *et al.*, 2023). However, amidst the vast medical information and rapidly evolving scientific advancements, patients and their families often struggle to access accurate genetic counselling and

timely targeted drug information. Therefore, the establishment of a specialized website for lung cancer genetic counselling and targeted drug discovery is particularly necessary.

Our team has developed a website, LCGenome (project URL on GitHub: https://github.com/guobiao-ye/JavaWeb_LCGenome), a user-friendly platform for data storage and information retrieval for lung cancer research and treatment (Fig 1). The website was built using Java Web technologies to present complex medical information in a user-friendly manner, ensuring that users can easily access and understand the content. By integrating the latest lung cancer genetic research and targeted drug information, LCGenome helps patients and healthcare professionals make more informed decisions. Through this project, we hope to bridge the gap between patients and the best treatment options and contribute to the fight against lung cancer.

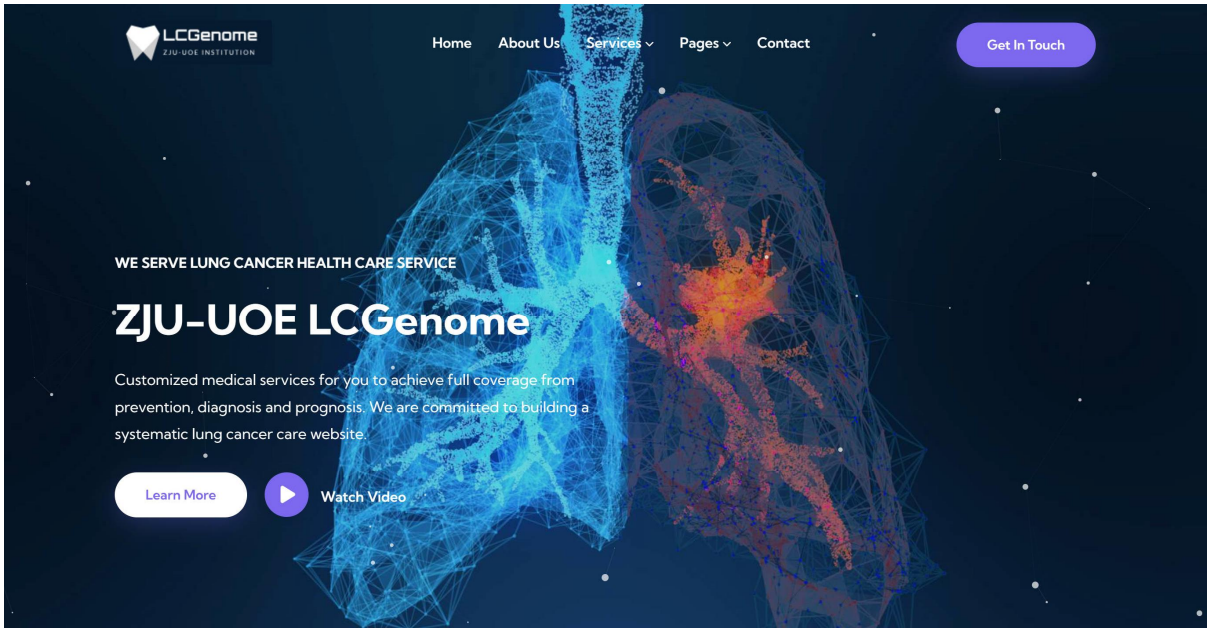


Figure 1. Home page of LCGenome. This page is created by "index.jsp "and integrates the basic information of the site.

3 Overall Team Workflow

During the development of the LCGenome project, we built a team oriented around four processes: requirements - design - development - test. As the developer of the project, I had to maintain thorough communication with the leads of each segment and always keep a clear understanding of the project's direction. Therefore, I took on the role of the team leader. Our project was phased with multiple team meetings. At the first team meeting, we found a project plan guideline from Huawei in DST2 demonstration projects, which we intend to use as a template to guide the advancement of our team's project (Huawei, n.d.). We also discussed our project goals and development model. To build a project with a broad database and clinical significance, we plan to develop a user-friendly website to store lung cancer data and provide query services. Due to our team's lack of experience in Java Web project Development, the Requirements analyst chose the Waterfall Model as the development life cycle. The model is easy to understand and use, and the content of each stage is stable, which is suitable for the first development. To achieve efficient teamwork, I created a team project on GitHub for members working together (Fig 2).

The subsequent meetings were centered on the design work, which was comprised of three distinct components: scientific design, web design, and database design. Each component was led by a single principal who was responsible for driving the work forward. As the developer, I played an integral role across all three components, interfacing with each principal during the meetings. Under the established overarching framework, the scientific designer further refined the operational logic of LCGenome for both research and clinical applications, detailing the user's

preparatory actions, the search process, and the anticipated outcomes. Notably, in terms of initial user preparation, the scientific designer meticulously documented the entire workflow for processing sequencing data to extract relevant information for querying the drug database. The database designer, responding to scientific needs, had sourced lung cancer genes and targeted drug information from the GDC and PharmGKB databases, respectively (Barbarino *et al.*, 2018; National Cancer Institute, 2021). The data crawlers were encapsulated into Java classes for my use. With this groundwork laid, I collaboratively discussed the visual composition of the website's front end with the web designer. The site encompassed several pages, with the services page identified as the central feature, providing two database modules, two search modules, one file matching module, and a module for PPI (Protein-Protein Interaction) analysis (Fig 3).

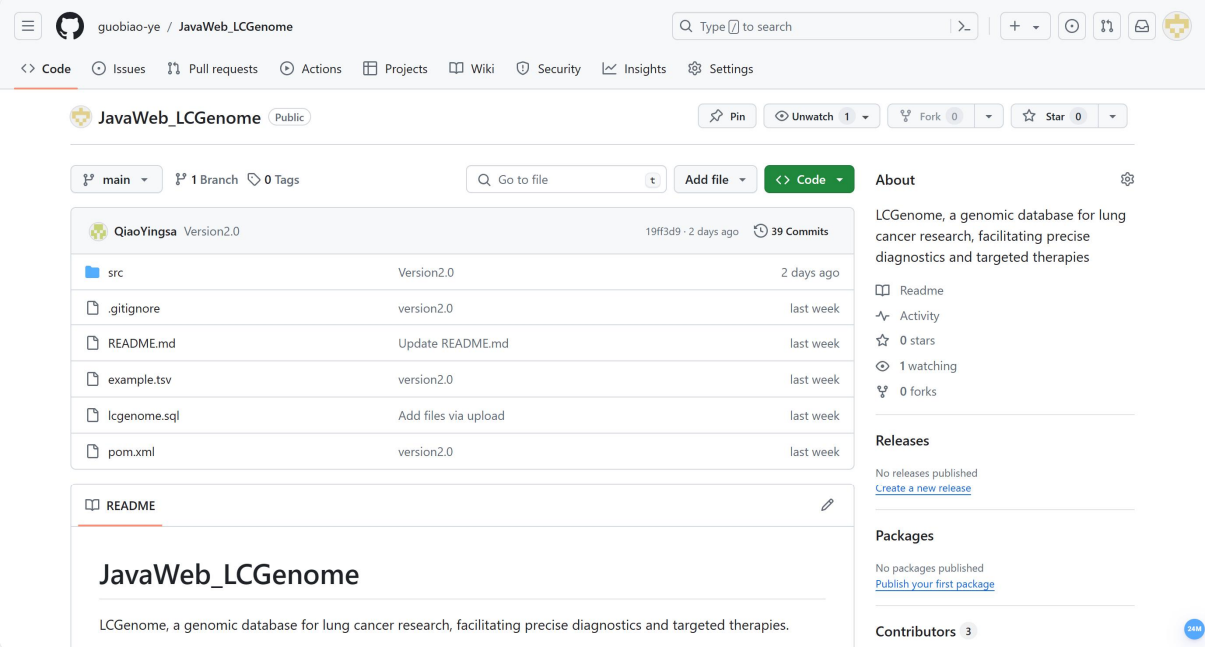


Figure 2. The LCGenome project on GitHub. Project source code and iteration information were uploaded to the GitHub project repository https://github.com/guobiao-ye/JavaWeb_LCGenome.

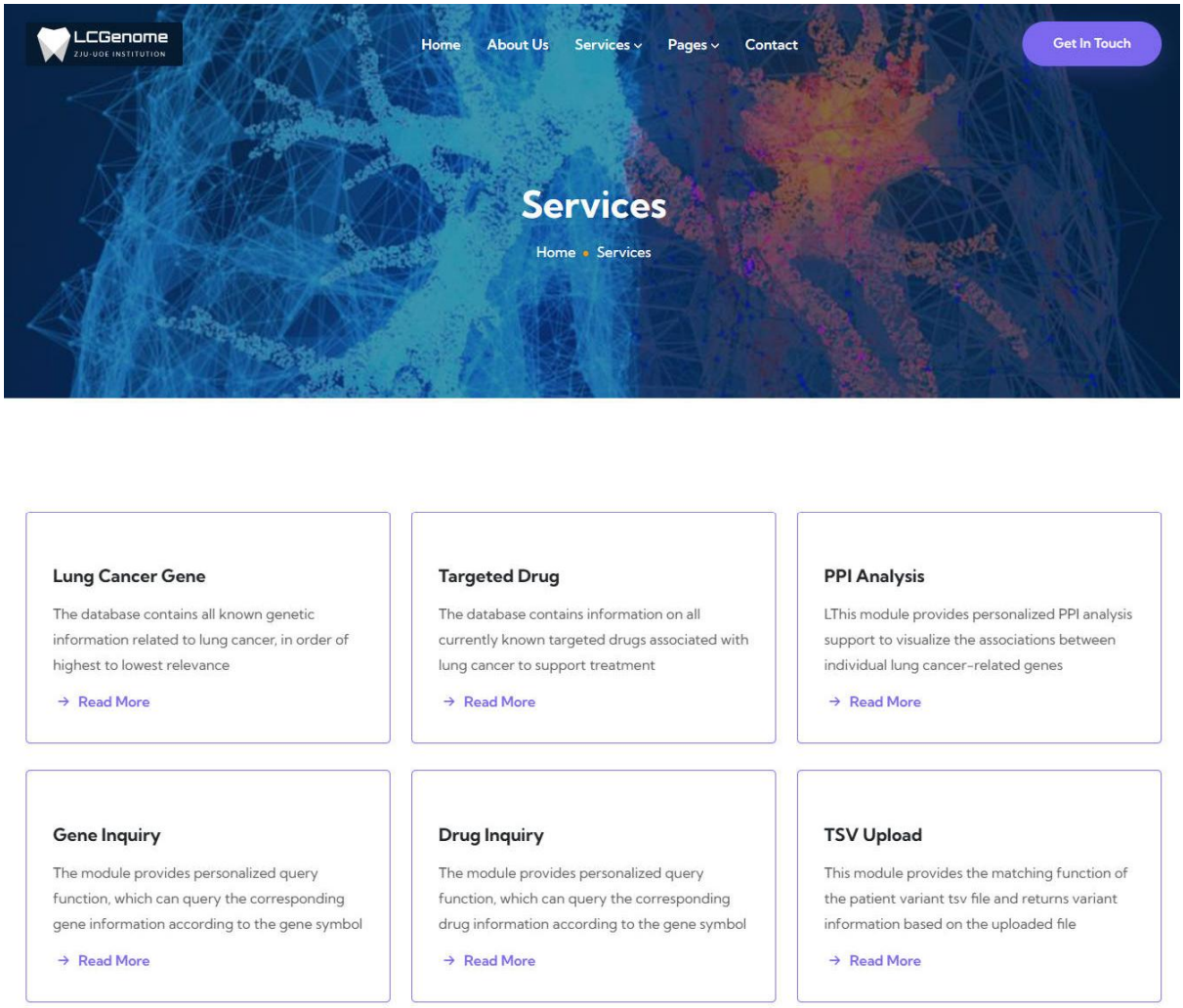


Figure 3. Services page of LCGenome. The Lung Cancer Gene module links to “gene.jsp” for lung cancer genetic information; The Targeted Drug module links to “drug.jsp” for information on lung cancer-related targeted drugs. The PPI Analysis module links to “ppi.jsp”, which provides personalized PPI network analysis and is still under development; Gene Inquiry module links to “gene-inquiry.jsp”, which retrieves the corresponding lung cancer gene information according to the gene input by the user; Gene Inquiry module links to “drug-inquiry.jsp”, which retrieves the corresponding targeted drug information according to the gene input by the user; TSV Upload module links to “tsv.jsp”, which aligns genetic information with the user-uploaded mutated gene TSV file.

As several meetings progressed and through collaborative communication with the various designers, I worked on the project architecture for the website. This process included the design of visual elements, the writing of front-end static resources, and the implementation of back-end web functionalities. I continuously uploaded and updated the source code to the team's project repository on GitHub for team members to review. After the development work was completed, the tester wrote unit testing code, conducted a comprehensive test on all aspects of the website's performance, and shared and discussed the results in the final meeting. The entire code implementation process is showcased on https://github.com/guobiao-ye/JavaWeb_LCGenome.

4 My Contribution as a Developer

In the process of building the sophisticated LCGenome platform for lung cancer medical information, as the project developer, I needed to carry out intricate coordination and partnership with team members across all stages. My contributions spanned the entire project lifecycle, from strategic planning to the ultimate deployment of the platform. Each phase was meticulously designed to ensure the platform's functionality, reliability, and user-friendliness, aligning with our goal of delivering a robust and accessible medical information resource.

4.1 Project Planning

At the onset of the project, I participated in the project planning meetings where, in collaboration with the team members, we established lung cancer as the theme for our website. After deciding on the Waterfall Model as the Development Life Cycle Model for our project, I began to formulate

my individual task plan based on the web project guideline from Huawei in course materials:

1) I need to be deeply involved in the systems requirement analysis to accurately capture the project goals and user needs, and to thoroughly understand the scientific logic behind the web pages, which will lay a solid foundation for the subsequent design work. This will require further interfacing with the requirement analysts and scientific designers. (4.2)

2) During the system architecture phase, I should focus on architectural planning and functional module design, and collaborate with the web designer to create an intuitive and user-friendly interface through front-end development. (4.3)

3) In the database integration phase, my main task will be to integrate the crawlers written by the database designer into the Java Web project, ensuring data security and smooth web page presentation by writing efficient back-end logic. (4.4)

4) During the back-end coding implementation phase, I should adhere to the requirements analysis and scientific design from before, contemplate the specific website services, and implement them through code. I will aim for a clear project structure and standardized code to ensure the correct implementation of system functionalities. (4.3, 4.4)

5) I will be involved in the initial execution of system testing to ensure the web pages operate correctly before they are submitted for further testing by the testers. (4.6)

6) System maintenance design will also be part of my responsibilities, providing the necessary technical support to ensure the long-term stability and maintainability of the system. (4.6)

Throughout the project lifecycle, I maintained close collaboration with team members to ensure that the technical realization perfectly aligned with the project goals.

4.2 Requirement Alignment and Environment Deployment

4.2.1 Website Functionality

In the prior project planning session, I meticulously documented the data requirements and anticipated user operational processes to prepare for subsequent development. Regarding data requirements, our website is designed with two storage modules dedicated to housing information on lung cancer genes and targeted therapies.

Collaboratively, the scientific designer and I delved into discussions regarding the user operational flow. The scientific designer, with a foundation in research and clinical significance, meticulously planned the website's services and functionalities. Together, we defined user stories and use cases to ensure that all features align with preset user needs.

As a developer, I am responsible for evaluating the feasibility of these preset features from a technical implementation perspective. I drew upon a demonstration project from our course materials, which showcased the implementation of a matching query feature for TSV files. Specifically, users can convert sequencing data into TSV format

files containing information on mutated genes. These files can then be uploaded to the website, where the backend analyzes the mutated gene names contained within and retrieves corresponding information from the database, presenting the results on the webpage. I planned to integrate this functionality into our website services, enabling users to search the database for gene information based on their uploaded mutated gene TSV files. Additionally, I designed two search modules that allow users to query a lung cancer gene database and a targeted drug database based on gene names. Moreover, I proposed the development of a Protein-Protein Interaction (PPI) analysis module. This module will provide users with a visualization tool to explore the biological pathways potentially affected by specific mutated genes, based on PPI networks.

After in-depth communication with team members, I outlined the development of six modules within the website's service section: two data storage modules for storing and displaying information on lung cancer genes and targeted drugs; a PPI analysis module for visualizing lung cancer gene interactions; two search modules for gene information and targeted drug queries; and a TSV file matching module for global retrieval of mutated gene information via uploaded files (Fig 2).

```
101 2  3  4  5  6  NAT2   8  9  10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
102 2  3  4  5  6  CHRNA5  8  9  10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
```

Figure 4. Demo TSV file. The TSV file only has the gene symbol column in string format. This column is used as a key for the website to extract and match against the lung cancer gene database.

4.2.2 Environmental Deployment and Tools

For the development of the LCGenome web project, I utilized IntelliJ IDEA, a powerful Integrated Development Environment (IDE) that facilitated the creation and management of our web application. Given the requirement for dynamic content, the project employed JavaServer Pages (JSP) to craft the webpages, allowing for the generation of HTML on the server side.

The application was designed to be deployed on an Apache Tomcat 9 server, a widely used Java-based web server and servlet container. Tomcat provides a pure Java HTTP web server environment for Java code to run in. It is a key component in our runtime environment, ensuring compatibility and stability for the web application.

The implementation of specific functionalities within the web pages necessitates the use of Java, hence the Java Development Kit (JDK) was an essential part of our development environment. For our project, we used JDK 11, which offers the latest features and improvements in Java language and runtime.

To establish a connection with the database, we leveraged the Java Database Connectivity (JDBC) API. JDBC is a standardized interface that provides a consistent approach to interacting with relational databases. It abstracts the database operations, allowing developers to use the same code to interact with various relational database management systems (RDBMS). This compatibility simplifies the process of switching between different databases, should the need arise.

In addition to these core technologies, I also incorporated various plugins and libraries within the IntelliJ IDEA to enhance productivity. These included plugins for Git integration, which facilitated version control and collaboration among team members, as well as libraries for JSON processing and data validation.

For the front-end development, I employed HTML5, CSS3, and JavaScript, utilizing modern frameworks and libraries such as Bootstrap for responsive design.

Lastly, to ensure the security and integrity of our application, I implemented best practices in coding and regularly updated the server and all dependencies to their latest versions, applying security patches as necessary.

Throughout the development process, I ensured that all tools and technologies were configured to work harmoniously within our project's ecosystem, providing a robust and efficient runtime environment for the LCGenome platform.

4.3 System Architecture

4.3.1 System Composition and Logical Structure

In the LCGenome project, the system composition and logical structure were carefully arranged to ensure high maintainability and scalability. The project adopted a standard Java Web application structure, which clearly delineated different functional modules and resources (Fig 5).

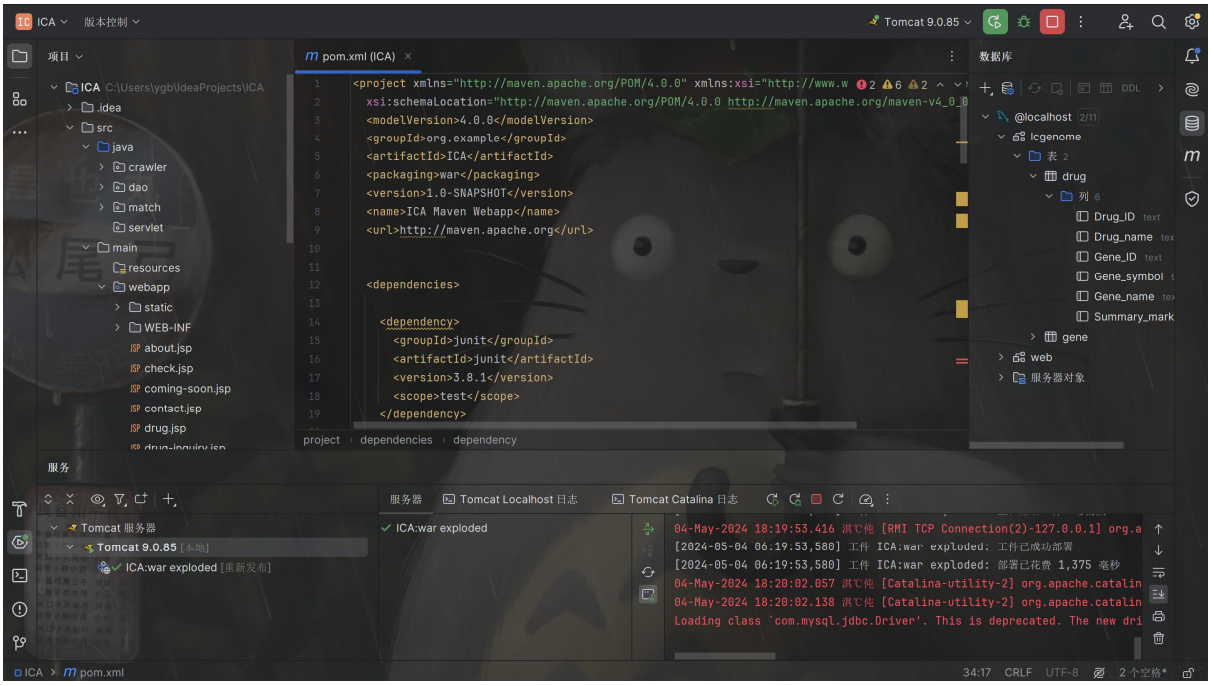


Figure 5. System Composition. The project is developed in the form of Java Web at IDEA.

At the foundation lies the ".idea" folder, housing configuration details for IntelliJ IDEA, ensuring consistency and efficiency in the development environment. The "src" directory, serving as the root of the source code, is further divided into Java and main subfolders. Within the java folder, logic was meticulously organized into subfolders such as "crawler", "dao", "match", and "servlet", each dedicated to specific functionalities like web crawling, data access, matching logic, and Servlet processing.

The main folder encompassed the primary resources and configuration files for the project. The resources subfolder within it stored configuration files and scripts, while the "webapp" folder contained the front-end resources of the web application. Inside "webapp", the "static" folder was designated for static resources like images and stylesheets; the "WEB-INF" folder held sensitive and

configuration files, ensuring security; and all JSP files were located directly under “webapp” for rendering dynamically generated web pages.

Additionally, the project included a “.gitignore” file to specify files that Git should ignore, along with “ICA.iml” for IntelliJ IDEA project configuration and pom.xml for Maven's project management and dependency configuration.

4.3.2 System Functional Modules

The system functional modules (4.2.1) of the LCGenome project revolve around the core Services page, services.jsp, which integrates the following six key modules:

1) **Gene Information Module - gene.jsp**
   Provides detailed information on lung cancer genes, helping users understand the association between specific genes and lung cancer.

2) **Targeted Drug Information Module - drug.jsp**
   Allows users to inquire about detailed information on targeted drugs, including their mechanisms of action, indications, and potential side effects.

3) **PPI Analysis Module - ppi.jsp**
   Offers a visualization tool using Protein-Protein Interaction (PPI) networks to explore the biological functions of specific genes.

4) **Gene Inquiry Module - gene-inquiry.jsp**

Users can search for gene-related information based on gene names, providing research support for researchers and medical professionals.

5) Drug Inquiry Module - drug-inquiry.jsp

Users can inquire about detailed information on specific drugs, including ingredients, methods of use, and precautions.

6) File Upload Module - tsv.jsp

Users upload TSV files containing information on mutated genes, and the system analyzes the file content to provide related gene and drug information.

The core functionality of the “tsv.jsp” module is supported by the “MatchVariants.java class”, while the functionality of other modules is embedded within their respective JSP pages. The project also includes a Login page, “login.jsp”, for user authentication before accessing the Services page (Fig 6). The authentication logic is handled by “check.jsp”, which verifies the user's account password by calling the “LoginService.java” class.

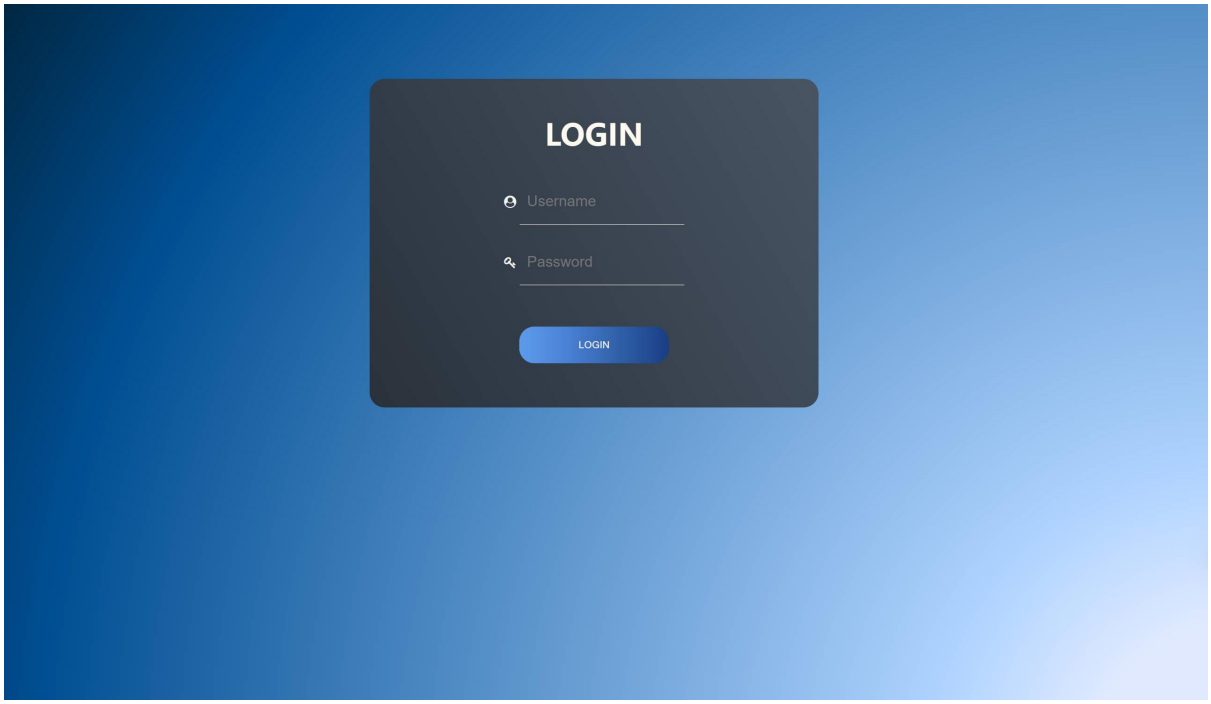


Figure 6. Login page. By connecting with the local account database and using "check.jsp" to check, the page realizes the login function.

4.3.3 Interfaces and Boundaries

The system architecture of the LCGenome project is modular, facilitated by clear interfaces and distinct boundaries, which are crucial for the application's scalability and maintainability. External interfaces are predominantly web-based, provided through JSP pages that allow user interaction via a web browser. The crawler module handles data interfaces, utilizing APIs and web services to fetch and parse data from external sources.

Internal interfaces are defined by the interactions within the “dao”, “match”, and “servlet” packages, maintaining a modular design that supports independent development and testing. The servlet module serves as an intermediary between the presentation layer (JSP pages) and the business logic layer, encapsulating APIs that separate concerns within the system.

The “dao” package establishes database access boundaries, abstracting the database structure to minimize the impact of changes on the application code. User interface boundaries are set by the design of JSP pages, each dedicated to presenting a specific module's functionality, ensuring a clear and focused user experience.

Authentication and authorization boundaries are implemented through “login.jsp” and “check.jsp”, with “LoginService.java” managing the verification of user credentials. Error handling and validation mechanisms are integrated at the user interface and data interface layers to ensure data integrity and system robustness.

The architectural design allows for the addition of new modules without disrupting existing functionalities, providing clear boundaries for future maintenance and expansion.

4.3.4 User Interface

In the LCGenome project, I took on the role of transforming the web designer's conceptual ideas and sketches into actual User Interface (UI) designs. I carefully considered how to convey information through visual elements while maintaining the clarity and aesthetic appeal of the interface. This process involved the meticulous selection of colours, fonts, layout, and graphic elements to ensure that the user interface was not only attractive but also user-friendly.

For front-end development, I utilized technologies such as HTML, CSS and JavaScript to translate the conceptual drawings into concrete web pages. I focused on creating responsive designs to ensure that the web pages would display well across different devices

and screen sizes.

To efficiently manage static resources, I created a static folder within the “webapp” directory. This folder was further subdivided into multiple subfolders, including “css” for style sheets, font for font files, “image” for larger JPG format pictures, “js” for JavaScript scripts, and “picture” for smaller PNG format images. This organizational approach made resource management more orderly and facilitated collaboration and maintenance among team members.

Additionally, within the “webapp” folder, I included all the JSP files that I had written and transformed based on the design concept (Figure 7). These JSP files are the core of the dynamic web pages, generating HTML content based on user requests and backend data.

Figure 7. Static resources and JSP files. The “webbapp” folder contains the static web resources and all JSP pages.

4.4 Database Integration

4.4.1 Establish a Connection to the Local Database

After the data designer finished his work, I was responsible for the integration of our web application with local databases. Utilizing the

crawler class developed by the database designer, I orchestrated the storage of lung cancer genes and targeted drug data tables into a local MySQL database named “lcgenome”. This process involved the systematic transfer and organization of critical genomic data, ensuring that the information was readily accessible for the application's needs.

Furthermore, I created an additional login database to securely store user account information. This database was designed with user authentication and authorization in mind, providing a dedicated repository for managing user credentials.

To bridge the application with these local databases, I employed the Java Database Connectivity (JDBC) API, which facilitated a robust connection between the application and MySQL databases. With JDBC, I was able to execute SQL queries and transactions, enabling the application to interact dynamically with the stored data (Fig 8a-e).

4.4.2 Filtering Feature

On the “gene.jsp” page, a straightforward filtering capability is implemented, enabling users to filter displayed information based on whether genes have targeted drugs (Fig 8a). This feature is facilitated through a dropdown menu embedded in the column header for "Targeted Drug" within the table.

Users can select "All," "Yes," or "No" to filter the gene information presented in the table. The dropdown menu is linked to a JavaScript script on the page, which triggers an event listener when a user selects it.

Upon selection, the script captures the value from the menu and iterates over the rows in the table. It examines the text content within the "Targeted Drug" column of each row to determine if it matches the user's selected filtering condition. Rows that align with the condition (or are left visible when "All" is selected) remain in view, while others are hidden from the user.

This filtering mechanism is entirely implemented on the front end using JavaScript, eliminating the need for server-side requests. This approach provides a quick and responsive user experience, allowing users to swiftly locate gene information of interest based on the presence of targeted drugs.

4.4.3 Search Feature

The search functionality on the LCGenome website is facilitated through form interfaces provided by JSP pages, where users enter relevant information such as lung cancer gene symbols, initiating a POST method request. Upon receiving the request, the server establishes a connection with the backend MySQL database using the JDBC API and constructs a parameterized query based on the user's input to ensure the security and accuracy of the search. The search results are retrieved by executing an SQL statement and obtaining a “ResultSet” object, which is then dynamically converted into an HTML table displayed on the page, or a prompt is shown if no matching records are found (Fig 8c, d). Additionally, to enhance user experience, the website employs CSS styling for the search form and result presentation, as well as JavaScript for client-side interactivity such as filtering and sorting.

4.4.4 File Matching Feature

On the TSV page of the LCGenome project, users can upload TSV files containing gene variant information through a clean and straightforward interface (Fig 8e). By clicking on the label adorned with an upload icon, a hidden file input field is activated, allowing users to select a TSV file from their local storage. Upon selection and clicking the submit button, the page sends a POST request with the file in multipart/form-data encoding to the server's “MatchVariants” endpoint. Upon receipt, the server endpoint processes the file, which involves reading the TSV file content, parsing the data, and executing a matching logic against the backend database. Once a match is found, the server sends back a dynamically generated HTML table of gene information to the client. This table is then displayed within the matched-info section of the page, presenting users with detailed gene information such as Gene ID, Symbol, Name, and more. Throughout the process, CSS styling is applied to enhance the interface's aesthetics, ensuring a user-friendly and visually responsive experience.

Figure 8. The function realization of each module is based on the local database. a) “gene.jsp” displays all lung cancer-related genes in the GDC database, sorted by the highest correlation coefficient. The “Targeted_Drug” column provides the screening function for users to select genes with or without currently targeted drugs. b) “drug.jsp” displays the information of all lung cancer targeted drugs in the PharmGKB database and their targeting relationship with lung cancer genes. c) "gene-inquiry.jsp" retrieves and presents the corresponding gene information according to the gene symbol entered by the user. d) "drug-inquiry.jsp" retrieves and presents the corresponding targeted drugs according to the gene symbol entered by the user. e) "tsv.jsp" extracts the gene symbols uploaded by the user to the mutant gene TSV file, retrieves the lung cancer gene database and presents the corresponding gene information.

4.5 Website Maintenance

4.5.1 Project Layering and Module Categorization

To ensure the long-term stability and ease of maintenance for the LCGenome website, I have placed a special emphasis on the clarity and rationality of the project structure. The entire website project is divided into logically distinct modules, each with a clear function and position. This not only aids in quickly locating issues but also simplifies subsequent feature expansions and iterative updates.

In the implementation of the project, I adopted a layered directory structure that effectively separates front-end pages, back-end code, database scripts, and static resources. For instance, the “java” folder is dedicated to housing back-end Java classes, while the webapp folder contains JSP pages and front-end resources. This separation allows front-end and back-end developers to work in parallel, enhancing development efficiency.

4.5.2 Comment

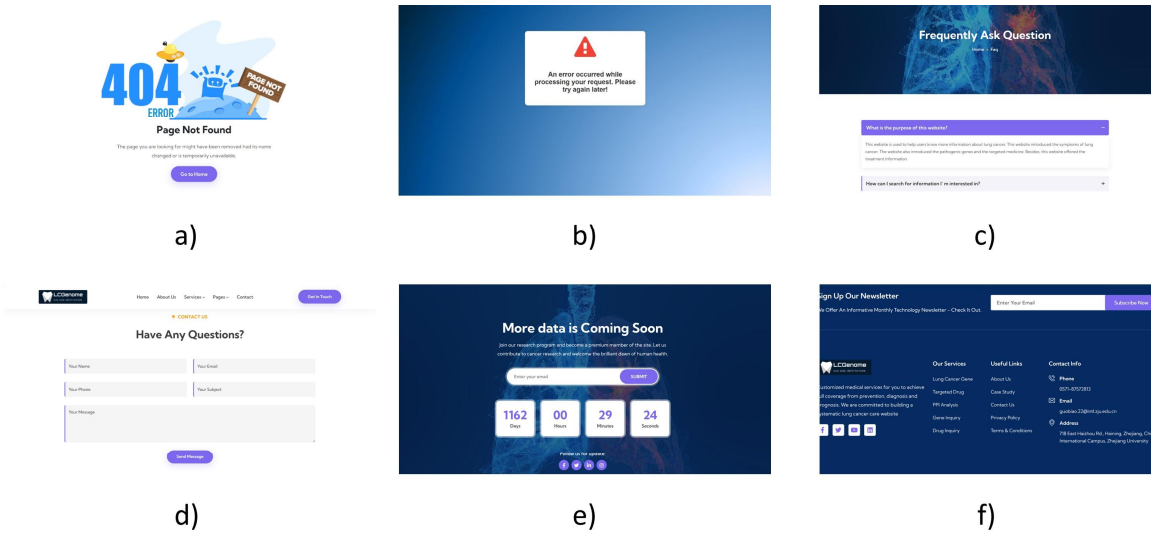
I extensively used comments and documentation throughout the project to ensure that every part of the code was clearly explained. This provides great convenience for maintenance personnel and ensures the continuity and stability of maintenance work, even when there are changes or handovers in project members.

4.5.3 Version Control Systems

I used the version control system Git to record every commit and update to the code. This not only helps us track historical changes in the project but also allows for quick fault location and rollback to a stable state in case of issues, greatly improving stability and security.

4.5.4 Related Pages to Assist Maintenance

To enhance the maintainability and user experience of the LCGenome website, we have implemented several specialized pages that address common issues and provide essential information (Fig 9). These pages not only improve the website's functionality and user satisfaction but also serve as essential tools for the maintenance team to monitor and address user concerns promptly. By providing clear communication channels and support resources, I aim to maintain a robust and user-friendly platform.



a) b) c)

d) e) f)

Figure 9. Maintenance-related pages. a) “error-404.jsp”: This page is designed to be displayed when a requested webpage cannot be found on the site, guiding users back to the main content with clear messaging and navigation options. b) “error.jsp”: In the event of a service failure, such as a mismatch in the TSV file during the variant information lookup, this page provides users with an error message and instructions on how to proceed. c) “faq.jsp”: A Frequently Asked Questions (FAQ) page was included that offers answers to common inquiries users may have about the website's features, how to navigate it, and how to interpret the data presented. d) “contact.jsp”: A

dedicated Contact page has been created to provide users with a direct means of reaching out to the support team or the website administrators. This page includes detailed contact information and is accessible from every page of the site via a consistent footer element. e) “coming-soon.jsp”: For community developers interested in contributing to the site's improvement, a Coming Soon page was developed. This page allows developers to submit their email addresses to join the development initiative and stay informed about upcoming projects and updates. f) Footer Consistency: The footer of every page consistently features the Contact information, ensuring that users can easily find a way to reach out for support or feedback, regardless of the page they are currently viewing.

4.5.5 Testing and Development Interwoven

In collaboration with testers, we adopted a comprehensive testing strategy to ensure the functional integrity and performance stability of the website. As a developer, during the development process, I conducted exhaustive functionality tests on the website by running a local Tomcat 9 server. This included but was not limited to, login authentication and the interactive use of six core service modules. This process simulated the workflow of real users to ensure that all features operate as expected and that the user interface is friendly and intuitive.

Subsequently, to further verify the stability and maintainability of the code, I uploaded the source code to the GitHub platform. This not only facilitated version control and team collaboration but also allowed the tester easy access to perform various performance tests. Through close communication with the tester, I collected valuable feedback

and made necessary improvements to the code accordingly. This iterative process of development and testing greatly enhanced the reliability of the website and laid a solid foundation for future functional expansion and maintenance. Through this rigorous testing cycle, the website can provide users with a stable and responsive online experience.

5 Discussion

Building on teamwork and through multiple project iterations, we have developed a user-friendly platform for data storage and information retrieval in lung cancer research and treatment. In this process, I utilized Java Web technology to translate the requirement and the scientific logic into the workflow of an interactive platform; I implemented the concepts of web designers using static web page technology, further refining the web page design to enhance the aesthetic appeal of the user interface; on the core service pages of the website, I implemented four main interactive functions: user login, data presentation, data retrieval, and file matching, which encompass a login module and five service modules; during the web page maintenance process, I added pages to assist with website maintenance and collaborated with the testing team for subsequent project iterations; additionally, I added a series of extension pages to assist users, aiming to achieve user-friendliness and completeness of the website (Fig 10). The entire process strictly adhered to Huawei's website design guidelines.

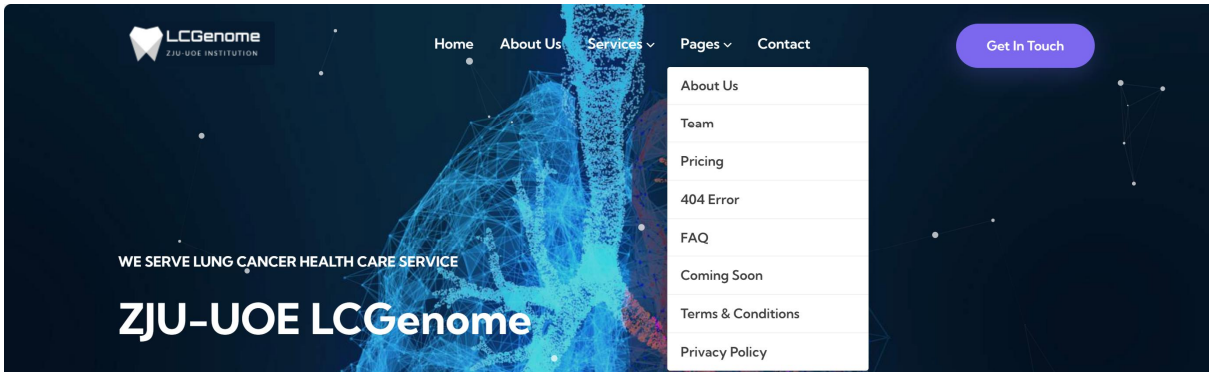


Figure 10. Top navigation bar and extension page. The user can be directed to any location on the site from the top navigation bar on every page.

The current data retrieval feature on our website is limited to presenting gene information or potential therapeutic drugs based on a single gene symbol, which can be cumbersome for users. For future enhancements, I aim to integrate data from two databases so that the website can display both gene details and corresponding targeted drug information on one page using just the gene symbol. This will require merging Java methods within the relevant JSP pages.

Regarding the TSV file matching module, users are required to have a TSV file that has been processed from sequencing data before they can utilize this feature, which can be a barrier to entry. We are exploring the possibility of extending the website's capabilities to include the preprocessing of sequencing data, which would involve learning about complex data processing workflows and web technologies.

Beyond the existing functionalities, we had initially planned to incorporate a PPI network analysis module into our website services (4.2.1). This module would use interactive graphical components to illustrate PPI interactions. During development, I have contemplated viable approaches

to implementing this feature. Given the extensive list of genes related to lung cancer, I propose focusing on the top 50 genes with the highest correlation to lung cancer for constructing the PPI network. To enhance interactivity and cater to users' individual analysis needs, I intend to include hyperlinks on each gene icon within the PPI network. This would allow users to access gene and targeted drug information from our database by clicking on the relevant genes. Furthermore, I plan to introduce buttons that would enable users to filter the PPI network and alter its display to suit their specific requirements. Addressing two main challenges will be crucial for realizing this feature: determining the appropriate filters or display modifications that align with users' analytical needs, and learning how to implement these features through coding.

Overall, the website has successfully managed to store and present information on lung cancer genes and targeted drugs. It offers users basic functionalities such as querying and file matching, with additional pages designed to enhance user experience. Although some features are still in the development phase, the website already holds significant scientific and clinical relevance in the field of lung cancer.

6 Conclusion

As the developer and project leader on the LCGenome project, I've been closely interfacing with team members at every stage and have taken on the responsibility for both front-end and back-end coding to realize the website. I've constructed all the pages of the website and implemented its core functionalities, which are based on a local database, including user login, data presentation, data retrieval, and file matching. The website has

already possessed scientific and clinical significance in the field of lung cancer. Looking ahead, I plan to refine the data presentation in the search module, add a PPI analysis module, and try to develop relevant functions to help users with pre-sequencing data processing.

7 References

Barbarino, J.M. *et al.* (2018) 'PharmGKB: A worldwide resource for pharmacogenomic information', *Wiley Interdisciplinary Reviews. Systems Biology and Medicine*, 10(4), p. e1417.

Huawei. (n.d.). Detailed Design Scheme Template. In *UOE-100235-0005159-1379: IBMS8013A: Database and Software Technology 2-1379*.

Lahiri, A. *et al.* (2023) 'Lung cancer immunotherapy: progress, pitfalls, and promises', *Molecular Cancer*, 22(1), p. 40.

National Cancer Institute (2021) Genomic Data Commons. [Online] *Available at: https://gdc.cancer.gov/*

Thai, A.A. *et al.* (2021) 'Lung cancer', *Lancet (London, England)*, 398(10299), pp. 535–554.