Browser-Based 3D Gallery To Showcase Student Work

Yikun Zhao

Master of Science
School of Informatics
University of Edinburgh
2024

Abstract

This report presents a browser-based 3D gallery to provide students with an immersive and interactive display platform to showcase their work. This report analyses and evaluates the technical difficulties and advantages of the 3D-gallery, the design of the user interface and optimisation of user experience, as well as accessibility and user-friendliness. This gallery has been tested and improved through user studies, including optimisation of browser compatibility and assurance of the visual quality of the student work. The results of these user studies show a general increase in user satisfaction with the improved gallery with an average rating increase of approximately 42 %. This gallery also has consistent and good cross-browser compatibility and a satisfactory accessibility score of 90. These results demonstrate that the gallery can be a useful platform for students to showcase their work.

Keywords: 3D-gallery, Browser-based, Accessibility, Artwork, User Interface, User Experience.

Ethics approval

This project obtained approval from the Informatics Research Ethics committee.
Ethics application number: 710059
Date when approval was obtained: 2024-06-13

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Yikun Zhao*)

Acknowledgements

I would like to express my heartfelt gratitude to my supervisor, Dr Brian Mitchell. The consistent support, detailed instructions, and profound knowledge provided by him have been invaluable during the whole process of writing this thesis. He provided me with much academic advice to help me overcome complex difficulties. His commitment to excellence and his patience in guiding me through each stage of this thesis have profoundly influenced my academic growth.

I would also like to thank all the professors who have taught me over the past year. Their lectures, discussions and feedback have been instrumental in developing my research skills and deepening my knowledge.

To my family, I owe my deepest gratitude. Their unconditional love and support always encourage me in my life. Their constant support has given me strength and motivation, and their encouragement and help have kept me going through the toughest of times and allowed me to continue to pursue my goals even in the face of difficulties.

Gratitude is extended to my classmates and friends who have been a constant source of support and companionship. Their understanding and patience have meant a lot to me. Special thanks to those who offered their advice, shared their experiences and provided breaks from the intensity of academic life. I feel incredibly fortunate to have such wonderful people in my life.

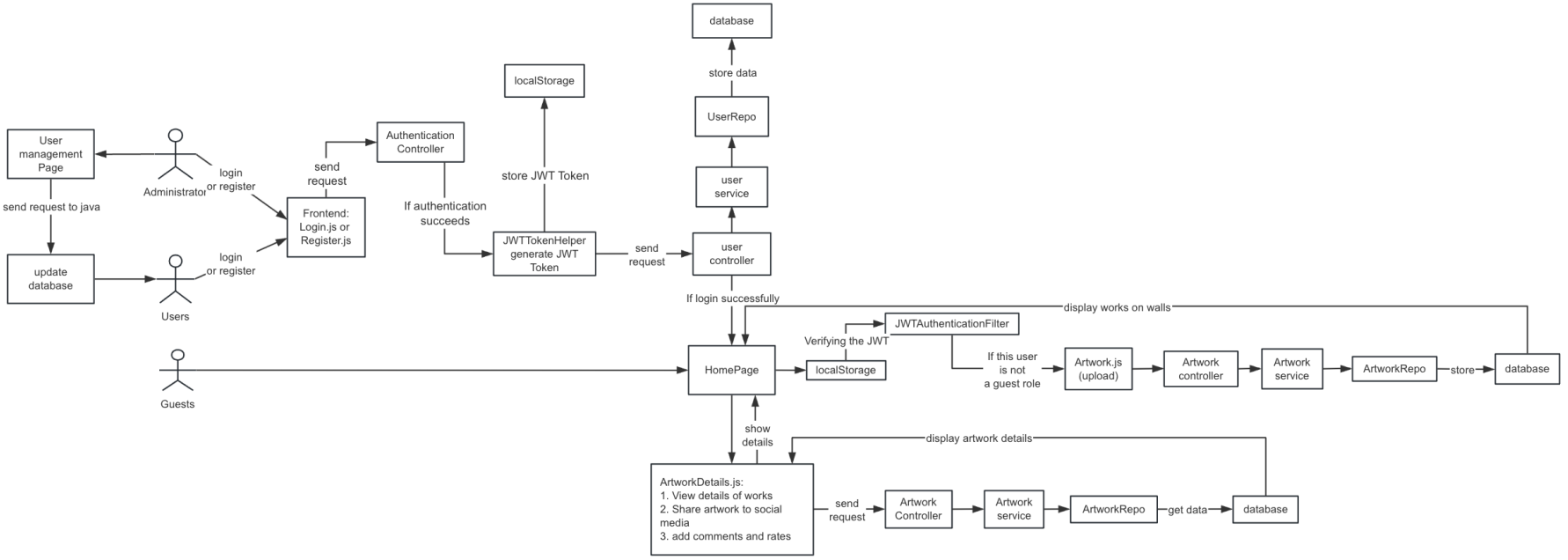


Figure 1: Project overview.

Tables of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Project motivation

This project creates a browser-based 3D-gallery to showcase student work, helping the University of Edinburgh's School of Informatics display its students' projects. A 3D-gallery, offers technological novelty, interactivity, and participatory elements that can attract audiences. The School of Informatics lacks suitable physical space for displaying student projects. The main way students display their work is by putting posters on the wall, though project posters are often limited to a temporary display one afternoon or evening per year. While this traditional form of display provides students with the opportunity to showcase their projects to some extent, it is limited by the physical space and time. Probably most people in the School have little idea of the contents of the vast majority of Honours and Masters projects, as well as PhD projects.

A 3D-gallery can overcome the constraints of time and physical space, allowing more people, to access and browse these excellent projects conveniently via the Internet. It also enhances the viewers' experience and participation due to its technological novelty, interactivity and engagement, while the School of Informatics has the technical facilities required to create a 3D-gallery. Therefore, a well-designed 3D-gallery with an intuitive and accessible interface could not only help improve the in-house situation but also help students' work gain recognition from a broader audience.

1.2 Project objectives

This project concerns creating a 3D-gallery accessed through web browsers. The gallery showcases student work, including 3D-models, digital artwork, multimedia content, and interactive simulations. This provides an immersive experience and an opportunity to engage virtually with student work such in fields such as artificial intelligence, data science, and computer science generally. The gallery also offers interactive features of click-to-interact, message pop-ups, and social sharing to enhance user engagement. Inclusivity (in terms of disabilities) is a key design factor.

Students can upload their work to the platform and use it to showcase their work, reach a wider audience, and have the opportunity to receive feedback. Researchers and educators can access more resources on the platform, such as in-depth interpretation and learning materials, promoting interdisciplinary collaboration and inspiration.

This gallery is designed to work properly on major web browsers, specifically Firefox, Chrome, Safari, and Edge. The primary focus (at the supervisor's request) is on Firefox, while still providing users with a consistent and high-quality experience across common browsers and devices. This is accomplished through the use of responsive design and progressive design enhancements. Open-source tools and resources will be utilized to create and animate models. Three.js and React.js are used for displaying 3D-content in browsers.

Scalability is also a key consideration in handling an ever-expanding library of work and managing increased traffic as the gallery's popularity grows. Scale is no problem during development but it was important to use good design that will scale. Thus detailed documentation has been created to assist future developers and users in navigating and maintaining the gallery, thus ensuring its long-term sustainability as an educational tool.

1.3 Importance of the problem

Traditional galleries have limitations in physical space and time. The serious and uninteractive atmosphere of brick-and-mortar galleries, along with obscure and boring explanations of student works, does not necessarily resonate with today's audience that prefers cutting-edge and novel content (Xanthoudaki, 1998). The

School of Informatics has the additional problem of a lack of (permanent) display space. The School currently displays student project posters only on two evenings per year, one each for Honours and Master's students. The School also lacks a more interactive and accessible platform, even though it does not lack for suitable infrastructure and technical skill.

Younger generations are highly interested in and accepting of new digital technologies (Mason and McCarthy, 2006) and use the internet and social media frequently. If a gallery can modernize by incorporating digital technology, expanding its exhibitions with more visual and auditory elements, increasing entertainment content and social interaction functions, and integrating art with entertainment and socializing, it will become more appealing to visitors (Prensky, 2001). Edinburgh's Our Dynamic Earth and London's Science Museum have successfully adopted interactive and modern participatory technologies.

This 3D-gallery overcomes the School's lack of a physical gallery, offering more convenience for students to showcase their work:

1. by breaking the limitations of physical galleries in terms of location and time, student work such as a project on data science or computer science becomes more accessible, providing publicity for exhibits and a barrier-free viewing environment for people with mobility impairments (Schweibenz, 1998). This gallery provides students with a broader space to display their work, offering an opportunity for their creativity and works to be more easily discovered and appreciated;
2. by combining multimedia and interactive displays to provide audiences with rich educational resources. The artworks are integrated with dynamic interaction, increasing interest and attracting the audience while stimulating curiosity and encouraging deeper exploration of the artworks. Prensky (2001) indicates that this interactive approach makes it easier for audiences with no professional background to appreciate the student work, diversifies the ways of display, and enhances audience understanding;
3. by facilitating social sharing, the platform allows viewers to share student work they encounter in the gallery on social media. This promotes interaction and communication among viewers (Schweibenz, 1998). The interactive sharing feature allows users to share their views and experiences through social media. This not only increases the visibility of student projects but also promotes their research and ideas to a wider audience.

1.4 Difficulties of the problem

Physical galleries face limitations in terms of space, requiring careful arrangement of student work to allow for sufficient visual space while maintaining a cohesive display (Falk and Dierking, 2016). Falk and Dierking (2016) indicate that managing lighting to prevent reflections and shadows, as well as controlling environmental conditions like humidity, temperature, and light, adds to the difficulty of preserving and caring for the exhibits.

3D-gallery faces technical and user experience design difficulties. The user experience in the virtual environment, the ease of browsing, the realism of the visual effects, as well as the network speed and server stability will affect the entire viewing experience.

The process of creating this 3D-gallery faced multiple difficulties in graphics processing, cross-platform compatibility, user interaction, security management, and system performance:

1. 3D-gallery development requires powerful graphics processing capabilities and complex technical settings to achieve consistency and smoothness. Insufficient processing power made development in a virtual machine impracticable;
2. There are difficulties in achieving cross-platform compatibility, especially with Firefox. This gallery needs to implement responsive design and employ progressive enhancement techniques to optimise the user experience to provide a consistent, high-quality experience across different browsers;
3. Developing and debugging interactive functions is complex and time-consuming. Intricate operations such as commenting require precise detection and processing to offer users a natural and seamless experience;
4. Protecting user-uploaded content and maintaining user data privacy through secure storage and management of user information remains a difficult aspect. Strong user identity verification and authorization mechanisms should be implemented and reasonable file upload limits need to be set.
5. Optimizing the gallery effectively to provide smooth system performance under high load is difficult. Constant attention to performance optimization strategies is required during the development process.

1.5 Target audience

The target audiences for this project are primarily:

Students This gallery offers students from various disciplines a platform to showcase their projects in an interactive and visually appealing manner. This allows students to present their work more visually, thereby improving the presentation and understanding of the audience.

Educational institutions Educational institutions of all types can make use of this 3D-gallery to improve their digital infrastructure. It can be utilized for school open days, exhibitions, and academic exchange events. By showcasing their students' outstanding work in this manner, educational establishments can boost the reputation and appeal of the school.

Teachers and mentors Teachers and mentors can use this platform to streamline the process of reviewing and assessing student projects. This 3D-gallery has functions such as viewing student work from different angles, zooming in on details and experiencing dynamic demonstrations, which provide a richer and more interactive experience than traditional formats, helping teachers better understand students' design thinking and project details.

Companies Students and graduates can showcase their skills and creativity in a 3D-gallery to stand out in the job search. Employers and recruiters can browse through the gallery to visualize candidates' capabilities and creativity, helping them make more informed hiring decisions.

Chapter 2

Background

2.1 Background explanations

2.1.1 Front-end

To provide a clear and easily maintainable code structure, as well as a high-performance and immersive user experience, this project combines React.js and Three.js to develop an efficient interactive 3D-gallery (Evans et al., 2014). React.js is a JavaScript library that is open-source and focuses on creating user interfaces, especially for single-page applications (SPAs) (Kumar and Singh, 2016). It divides the user interface into reusable components using componentization. Each component manages its state and lifecycle independently. It improves rendering performance and development efficiency by utilizing virtual DOM technology, which involves creating a memory-based copy of the DOM tree and updating only the changed parts when the state changes (Aggarwal et al., 2018). Its key features include declarative programming, unidirectional data flow, and a Hooks system. These features assist developers in effectively managing and updating the state and views of their applications, thereby enhancing development efficiency and code maintainability (Fedosejev, 2015). React.js is primarily used to build the component-based architecture of the user interface, utilizing a componentized approach for functions such as `'Navbar'`, `'Artwork'`, and `'ArtworkDetails'`. It manages component states through hooks like `'useState'` and `'useEffect'`, controlling elements like the upload modal box, click position, and user login status. It facilitates user interaction and dynamic component behaviour through lifecycle methods and event-handling mechanisms, such as the onClick event.

Three.js is a cross-platform JavaScript library based on WebGL for creating and displaying complex 3D-graphics in web browsers (Parisi, 2012). This library simplifies the direct use of WebGL and enables developers to easily create 3D-scenes, models, and animations (Dirksen et al., 2013). It offers a rich set of features and tools, including geometry, materials, lights, shadows, animations, and physics effects for game development, data visualization, Virtual Reality (VR), and Augmented Reality (AR) applications (Danchilla and Danchilla, 2012). Its high-performance and powerful 3D-rendering capabilities allow developers to achieve realistic 3D-effects and complex interactions in the browser (Dirksen et al., 2014). Three.js is employed to create and manage complex 3D-scenes, including geometry, materials, lights, and cameras. It is responsible for creating elements like gallery walls, floors, ceilings, and artwork, and also handles the loading and display of image textures using the `'TextureLoader'`. Its animation loops and event listeners enable smooth animation and user interaction with the 3D-objects.

This project uses HTML5 canvas to create interactive components. The `'index.html'` file is the entry point for the application, bringing in external resources, setting up viewports for responsive design, providing the root node for mounting React components, and launching the main JavaScript file to render the React application. This approach is commonly used for Single Page Applications (SPA) setup (Mikowski and Powell, 2013). The setup involves creating the base HTML file, setting up viewports, defining a root node for mounting JavaScript framework components, and introducing the main JavaScript file 3. This clear separation of code structure improves maintainability and scalability (Jadhav, Sawant and Deshmukh, 2015). HTML creates the web page structure, CSS controls visual style and layout, and JavaScript manipulates the Document Object Model (DOM) for dynamic effects and user interaction (Robbins, 2012).

2.1.2 Back-end

The back-end uses the SpringBoot framework and adopts a layered architecture, including configuration, controller, model, warehouse, and services. Each has clear responsibilities and collaborates to achieve the functions of the gallery management system. The configuration module includes security configuration such as JSON Web Token authentication and password encoder configuration, file upload settings and Web configuration; controller module to deal with user authentication, art

management, file operations and sharing functions of the HTTP request; model module defines the user, art and permissions, including the definition of the entity class and its data transfer object; warehouse module using Spring Data JPA and database interaction, to provide access to art and services. The Warehouse module interacts with the database using Spring Data JPA to provide access to artwork and user data; the Service module implements the main business logic, such as artwork management, user services, and custom user services.

Spring Boot is an application development framework built on the Spring Framework. It simplifies the configuration and development of Spring applications. With Spring Boot, developers can efficiently build, test, and deploy applications using autoconfiguration, embedded servers, dependency management, and production readiness features. Gajewski and Zabierowski (2019) mentioned that autoconfiguration sets up Spring components based on the project's dependencies, and the embedded Tomcat or Jetty servers support standalone application deployment, streamlining the traditional deployment process.

JSON Web Token (JWT) is an open standard (RFC 7519) for securely transmitting information between parties. It is mainly used for authentication and information exchange. According to Ahmed and Mahmood (2019), JWT consists of three parts: the header, the payload, and the signature. The header contains the token type and signing algorithm, the payload contains the user's information and declarations, and the signature is used to verify the authenticity and integrity of the token. After the user logs in, the server generates the JWT, and the client carries the token in subsequent requests. The server ensures the request's legitimacy and the data's integrity by verifying the signature.

This project utilizes PostgreSQL as the database management system, along with pgAdmin4 for visual management of the database. PostgreSQL is a high-performance, stable, and scalable open-source object-relational database system that supports complex queries, transaction processing, and a rich set of data types to ensure data integrity and security (Obe and Hsu, 2017). PgAdmin4 provides an easy-to-use graphical user interface (GUI) that supports database creation, modification, deletion, user management, rights management, and real-time performance monitoring (Worsley and Drake, 2002). This combination not only enhances the efficiency of database management but also provides reliable technical support for data processing and project management.

Four major browsers were selected to ensure the compatibility and user experience of the project: Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari. Google Chrome is the preferred browser for development and testing due to its excellent performance, wide range of extensions, and developer tools. It holds more than half of the browser market share (Nelson, Shukla and Smith, 2020). Mozilla Firefox is known for its emphasis on user privacy and security. This project requires it to perform well in Firefox to reach a larger user base. Microsoft Edge, as the default browser on Windows, is optimized for performance and security— important in the Windows ecosystem. Safari, the default browser on Apple devices, is widely used among Mac and iOS users, ensuring compatibility and a smooth project experience on Apple devices. These browser choices not only cater to the mainstream user base but also ensure the consistent and efficient functioning of the project across different platforms and operating systems.

The project will adhere to Web accessibility best practices to ensure that this 3D-gallery is accessible and usable by all users, including those with visual, hearing, motor, and cognitive impairments. Web accessibility is not only a legal and ethical requirement but also helps to expand the user base and improve user experience and satisfaction (Chisholm and Henry, 2005). Based on the Web Content Accessibility Guidelines (WCAG) version 2.1 standards developed by the World Wide Web Consortium (W3C), all content and functionality in this project conforms to these standards (W3C, 2018). Specifically, this 3D-gallery will focus on providing alternative text, using ARIA attributes, ensuring form accessibility, ensuring colour contrast, and testing via screen readers. These measures will not only improve the project's accessibility but also create a more inclusive and user-friendly web application to ensure that all users can fully experience and enjoy this 3D-gallery.

2.2 Resources

2.2.1 Frontend languages

HTML + CSS + JavaScript

Advantages Supported by all browsers and easy to learn and use for basic web development. Large community with extensive resources and libraries.

Disadvantages Basic JavaScript might not efficiently handle intensive 3D-rendering. Requires additional libraries for complex 3D-graphics.

Notes This project needs to handle many 3D-models, so an additional framework is needed.

Three.js

Advantages Specialized library for creating and displaying 3D-graphics in the browser, perfect for the project's needs (Dirksen et al., 2013). Easily integrates with other JavaScript frameworks like React.js (Mardan, 2017). Well-documented with a supportive community.

Disadvantages Intensive 3D-scenes require significant performance optimization. The steeper learning curve for those unfamiliar with 3D-graphics concepts.

Notes This one is useful for this project due to its ability to handle 3D-graphics.

Vue.js

Advtanges It is simple and intuitive syntax, suitable for beginners and experienced developers alike and it is lightweight and efficient (Saks, 2019).

Disadvantages Relative to React and Angular, Vue.js has a slightly smaller ecosystem, and fewer resources and support.

Notes Vue.js is good but React.js and Angular.js are better.

React.js

Advantages Declarative and component-based architecture makes it easier to build interactive UIs, with a large ecosystem and strong community support (Saks, 2019). This makes it excellent for managing complex state and dynamic content.

Disadvantaes React's flexibility allows for multiple approaches to achieve the same goal; Slow rendering compared with Vue and Angular (Diniz-Junior et al., 2022).

Notes The framework was chosen as the best choice for this project due to its ability to efficiently create dynamic, interactive user interfaces and its seamless integration with Three.js for 3D-rendering.

Angular.js

Advantages Comprehensive framework with integrated tools for routing, state management, and form validation; enhanced code quality and maintainability with strong typing using TypeScript (Diniz-Junior et al., 2022).

Disadvantages Learning Angular can be more difficult compared with React and Vue, and it may be overkill for smaller projects (Saks, 2019).

Notes This is an official "major framework," but it might be too extensive compared with the lighter combination of React.js and Three.js.

2.2.2 Backend languages

Python

Advantages Python's syntax is clear and concise, making it easy to learn and write. It has a vast array of libraries and frameworks that can significantly speed up development (Khoirom et al., 2020).

Disadvantages Python is generally slower than compiled languages like Java or C++. Dynamic typing can lead to runtime errors that are harder to debug and maintain in a complex project (Khoirom et al., 2020). Python's GIL limits its efficiency in handling concurrent tasks, which is a drawback for real-time interactive applications.

Notes Python is good for rapid development and prototyping but might not be ideal for high-performance 3D-applications.

Java

Advantages Enables this 3D-gallery to be accessible on any operating system, making it more widely available. Its high performance is essential for real-time 3D-rendering and interactions. Static typing and compile-time checking improve code reliability and maintainability (Arnold, Gosling and Holmes, 2005). Java integrates well with WebGL for 3D-graphics and includes built-in security features to ensure application security (Anyuru, 2012).

Disadvantages Java's verbose syntax may slow down development. Despite having automatic garbage collection, mishandling can still cause memory problems in resource-intensive 3D-applications.

Notes This one is well-suited for this high-performance 3D-gallery project requiring robustness and security.

PHP

Advtanges Designed for web development, allowing seamless integration with HTML, CSS, and JavaScript (Tatroe and MacIntyre, 2020). Its

simple syntax enables quick learning and development. It also has a strong community and popular frameworks like Laravel and Symfony.

Disadvantages It is generally slower than Java and C++ and is often susceptible to security problems if not properly coded, which could compromise the safety of user data in your gallery. Dynamic typing and the potential for messy code structures can complicate maintenance.

Notes PHP is suitable for simple web applications but may struggle with the high-performance demands of a 3D-gallery.

C/C++

Advantages C and C++ excel in performance and memory control, making them ideal for complex 3D-rendering. C++ supports multiple programming paradigms, providing design flexibility (Stroustrup, 1986). These languages are widely used in game development and high-performance applications.

Disadvantaes The complexity of C and C++ makes development slower and more error-prone (Prechelt, 2000). They lack built-in web development frameworks, requiring developers to build everything from scratch.

Notes They are suitable for performance-critical components but require extensive additional work for web development.

2.2.3 Database

Mysql

Advantages Widely utilized with extensive community support and documentation; high performance, especially for read-heavy applications; simple to use with low maintenance costs (Rautmare and Bhalerao, 2016).

Disadvantages Limited support for complex queries and large-scale data processing (Filip and Čegan, 2020); lacks some advanced features such as full-text search and geospatial data types.

Notes Suitable for simple data needs, but it may not be robust enough for the complex requirements of a 3D-gallery project.

Postgresql

Advantages Powerful and highly extensible; supports complex queries, transactions, and concurrent processing (Makris et al., 2021). Offers rich support for various data types, including JSON, arrays, and geospatial data types

and it is highly compliant with standard SQL, offering great flexibility (Jung et al., 2015).

Disadvantages More complex to configure and optimize compared with MySQL. It may require more resources, time, and effort for management and maintenance (Makris et al., 2021).

Notes Due to its powerful features and flexibility, it is highly suitable for a 3D-gallery project that requires complex data processing and advanced features.

MongoDB

Advtanges A high-performance NoSQL database for unstructured data, designed for handling large volumes of data and high read/write operations (Palanisamy and SuvithaVani, 2020). Easy to scale horizontally.

Disadvantages Complex transactions not supported; query language can be complex, and handling structured data is less efficient than relational databases. Also requires more memory and storage space (Rautmare and Bhalerao, 2016).

Notes This solution is optimal for handling unstructured data and high concurrency but may not be suitable for a 3D-gallery project with complex queries and transactions.

2.2.4 Security methods

JWT

Advantages Self-contained; includes all necessary information within the token. Stateless; no need for server-side session storage. Easy to use with a variety of frameworks and libraries. Supports strong cryptographic signatures for data integrity and authenticity (Ahmed and Mahmood, 2019).

Disadvantages Payload size can become excessively large if too much information is included. It does not provide built-in mechanisms for token expiration and revocation (Ahmed and Mahmood, 2019).

Notes Ideal for stateless authentication and scalable web applications like a 3D-gallery; It is easy to use with various frameworks and libraries in Java (Adam, Moedjahedy and Maramis, 2020).

API Keys

Advantages Simple to implement and use; widely supported and easy to pass in request headers; suitable for simple access control scenarios (Farrell, 2009).

Disadvantages Not inherently secure, as they can be easily exposed and misused; lacks mechanisms for user-specific authentication and fine-grained access control (Farrell, 2009).

Notes Less secure for complex applications requiring user-specific authentication.

OAuth Tokens

Advtanges Provides strong security with user authentication and precise access control; supports token expiration and revocation; widely adopted standard for securing API access (Fett, Küsters and Schmitz, 2016).

Disadvantages More complex to implement compared with JWT and API Keys; requires maintaining OAuth servers and managing user sessions (Fett, Küsters and Schmitz, 2016).

Notes OAuth is too complex for this project.

Chapter 3

Design

3.1 User interfaces

When creating a 3D-gallery, it is important to carefully consider user needs, functional implementation, visual design, and user experience. This project has been developed using a combination of user experience (UX) and user interface (UI) design best practices to create an intuitive, easy-to-use, and accessible interface. Jakob Nielsen is a widely known and respected authority on usability. His 1994 publication Nielsen (1994) formed the basis for this project's interface design. Following this guideline, this gallery is designed to focus on the visibility of the system state, alignment with the real world, user control and freedom, and consistency and standards.

The gallery in this project has been designed by following the recommendations with a focus on aesthetics and minimalist design (Babich, 2019). These guidelines are good quality as they emphasize simplicity and user experience, which are crucial for creating a visually appealing and accessible gallery. Neutral tones such as white and grey have been chosen for the walls and floor to highlight the artwork and reduce visual distractions, allowing users to focus more on the artwork. The simple and clean background provided by the neutral tones makes the colours and details of the artwork stand out and be more appealing. To enhance the accessibility of this gallery, the project is designed according to the W3C's WCAG 2.1 standard (W3C, 2018). This provides sufficient contrast for colour-blind users and avoids inconvenience for dyslexic users through appropriate colour contrast and text size (Colblindor, 2016; WebAIM, 2019). The navigation bar uses a high-contrast design with a white background and dark text and icons

to enhance the visual effect, making it easier for users to identify and understand all the information on the interface. In terms of spatial layout, the design includes a spacious exhibition area and an open spatial layout to help users feel the fluidity and openness of the space during browsing. This open layout avoids a sense of visual crowding and allows users to move freely and explore all corners of the gallery. The use of smooth grey flooring adds a modern feel and reflections to create a sense of depth, making the gallery appear more spacious and brighter. Additionally, the use of high-quality virtual materials and detailed 3D-modelling enhances the overall sense of realism and visual experience.

Visual and functional consistency also plays a key role in aesthetic and minimalist design. Visual consistency is central to ensuring a coherent user interface (Yablonski, 2020). Therefore, the gallery uses a uniform colour and font style to make the entire gallery interface visually coherent and sets all buttons and icons in the navigation bar to the same colour and shape. To reduce the learning cost for users and enable them to quickly familiarise themselves with and use the functions of the gallery, the gallery uses the same effects for all actions, including clicking, sliding, and zooming, throughout the gallery to maintain consistency in all interactions. In addition, to make the gallery interface more modern and aesthetically pleasing, all pop-ups and forms in the gallery use elements of Glassmorphism. This design style mimics the look of glass by using blurring effects and transparency to give the interface a sleeker and more layered look (Brown, 2024) without completely obscuring the other work above the walls.

This gallery uses the `'PointLocker'` controller for user movement to enhance user control and increase flexibility and efficiency. This controller supports both mouse and trackpad control to manipulate the viewpoint and control the pointer, allowing users to choose the most convenient way to interact based on their operating habits. Users can use the keyboard to interact with the gallery as well. The displayed work is shown through the wall above the set of 3D display positions. If the user uploads the work in a video or PDF format, automatically generated thumbnails of the work will be displayed in the designated positions. Users can view all the details of the work, resize the work to view the details, and perform editing or deleting operations as needed. Responsive design has been implemented to enhance the flexibility of the gallery further, ensuring that it looks great on different devices and screen sizes. This allows users to access the gallery on the most suitable device based on their preferences and needs. Immediate visual and

audible feedback is provided when a user takes an action, making it clear that the action has been recognized and performed.

This gallery provides options for accessing help information on the welcome screen and the navigation bar. Users can learn how to use the gallery such as how to control the camera view, navigate the gallery, and view the works. It offers a multi-language option, allowing users to select different languages to increase international reach. There are easy-to-use feedback forms for users to submit questions and suggestions. The forms should include user contact information and a question description to enable the development team to respond and resolve problems promptly.

Chapter 4

Implementation

4.1 Implementing the back-end

The back-end of this project supports the front-end by managing data and handling user authentication including the administrator, user and visitor identities. The code structure can be seen in Appendix A. The Java frameworks used included Spring Boot, Spring Security, and JSON Web Token (JWT). Spring Boot is utilized to simplify the development of RESTful APIs and enhance application deployment efficiency. Spring Security managed user authentication and authorization and utilized JWT tokens to protect user sessions. The main difficulty in the development process is configuring security measures, including setting up `'BCrypt'` password encryption, generating and validating JWT tokens, and carefully configuring the security policy to maintain user authentication accuracy and data transmission security. Multiple security configurations are implemented to safeguard the gallery including encrypting user passwords, defining the project's security policy and specifying which URLs require authentication. The processing of the JWT token involved extracting the token from the request header, verifying its validity, extracting user information, and storing it for subsequent requests. A strong encryption algorithm is used to encrypt the token content and prevent unauthorized access and tampering during JWT generation. The token's validity period is strictly controlled to 3600 seconds to reduce the risk of token theft. This project also focuses on maintaining security in different browsers through strict CORS settings, considering the other support for security policies in browsers, especially for the differences in Same-Origin Policy (SOP) and Cross-Domain Resource Sharing (CORS) configurations. The operations not

only guaranteed system security but also provided an efficient user management mechanism by handling basic user information such as registration, login, and password reset, and implementing role-based access control. Different user roles had different permission ranges, for instance, administrators could access all users' information and perform editing and deletion operations, while regular users could only view and edit their information. This feature not only bolstered security but also enhanced the flexibility and scalability of this gallery in multi-user scenarios.

This project utilizes PostgreSQL as the primary database management system to store and manage data for the 3D-gallery. Using the Spring Data JPA and Hibernate ORM framework, database entities can be mapped to relational database tables making data reliable and consistent. The database architecture was designed to provide efficient data manipulation and flexible scalability, enabling the project to quickly adapt to changing business needs. Key resources stored in the database include artworks, users, authorities, and ratings. An overview of the main attributes is shown in Figure 4.1.

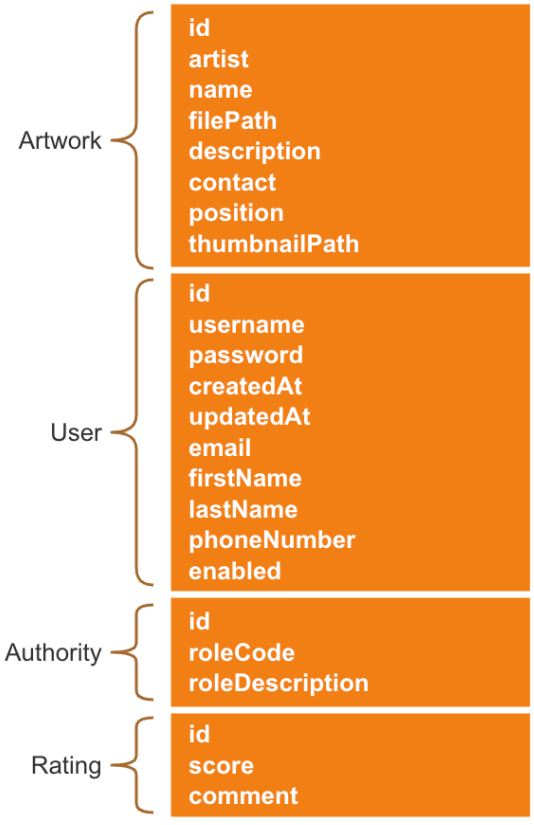


Figure 4.1 Attributes of the database model.

4.2 Implementing the interface

This project's interface is created to provide an immersive 3D-gallery experience. It is developed using the Vite build tool and manages its state through Redux. React.js provides powerful state management and efficient rendering capabilities, while Three.js is used to create and display the 3D-scenes at the heart of the project's immersive gallery experience. Vite is a modern front-end builder and development server designed to enhance the development experience and build efficiency, especially for modern JavaScript frameworks like React.js and Vue.js. It eliminates the need for traditional packaging tools to package the entire project during the development process by using native ES module imports. This results in a speedy startup time and compiles only the currently used modules, greatly improving compilation speed and efficiency, particularly in scenarios involving large-scale projects with multiple dependencies (Wieruch, 2017). You and Vite Team (2024) mentioned that Vite pre-bundles dependencies using `'esbuild'`, a tool written in Go that can pre-bundle dependencies 10-100 times faster than traditional JavaScript-based bundlers. This improvement enables developers to focus more on feature development and implementation, rather than being slowed down by lengthy build processes and complex build tool configurations, particularly in environments where rapid iteration is essential.

Redux is a JavaScript state management library for applications. It offers a predictable state container for centralized application state management, making state changes foreseeable and traceable. This makes the maintainability and scalability of state management in large applications (Mardan, 2017). Redux is used to store and update the user's authentication state and information by communicating with the back-end API. The back-end then returns a JWT token and user information after validating the user details, which are captured by Redux's action and stored in Redux's store. During subsequent operations, such as uploading works, the front-end retrieves the token from Redux and adds it to the request header before sending it to the back-end, thereby ensuring that the user's operation is authenticated.

To provide a smooth user experience, it is important to achieve consistent 3D-rendering and response performance across different browsers and devices with varying capabilities. This project heavily relies on Three.js for creating and displaying 3D-scenes, which, while powerful, demands high rendering performance.

For complex 3D-models and interactive animations, low-performance devices may experience frame rate drops and lag, directly impacting user experience. To address this problem, several optimization strategies are employed. Low polygon models are used whenever possible, and material textures are compressed and optimized to reduce rendering overhead, easing pressure on the GPU and improving overall scene loading speed. The lazy loading technique in Three.js dynamically loads resources in the scene based on user perspective and interaction, preventing performance problems caused by loading all resources at once. Rendering efficiency is optimized by minimizing unnecessary repaint operations and implementing frame-based animation control. For instance, shadow quality and lighting effects are automatically reduced on low-performance devices to maintain smooth rendering. This hierarchical optimization strategy is concerned with delivering a consistent user experience across devices with varying performance. Differences in WebGL and Three.js support across browsers may lead to problems or performance degradation when rendering complex 3D-scenes in some browsers. Compatibility testing is conducted on major browsers including Chrome, Firefox, Safari, and Edge. Configuration parameters of Three.js are adjusted to optimize rendering effects across different browsers and devices. For example, shadow resolution and lighting complexity are appropriately reduced based on browser performance to balance visual effects and rendering speed.

4.3 User studies

This project had two rounds of user studies ($n = 6$, 4 with technical backgrounds), each time using the feedback to optimize the gallery. These studies have contributed to the continuous enhancement of the 3D gallery's functionality, user experience, and accessibility which have led to a stable and satisfactory outcome. Participants' subjective experiences and feedback were gathered through interviews focusing on system performance, interface design, interaction experience, and accessibility. The user studies centred around three main goals:

1. **Identifying usability problems** within the gallery, such as the complexity of interaction processes, the intuitiveness of the user interface, and the effectiveness of user feedback mechanisms;

2. **Evaluating the gallery's browser compatibility** to achieve consistent 3D rendering and responsive performance across different browsers such as Chrome, Firefox, Safari, and Edge;
3. **Optimizing the design of user roles and permissions** so that different user roles, including guest, registered user, and administrator, receive clear and distinct functional guidance when interacting with the gallery.

To fully evaluate the usability of this project, the two user studies assign three user roles: visitor, registered user and administrator:

`Visitor:` An ordinary user who is not registered or logged in and can only use some of the public functions such as browsing the gallery, viewing artwork details.

`Registered User:` A participating user who, after registering and logging in to the gallery, can upload, edit and manage their own artwork, plus browse, comment and rate others' artwork;

`Administrator:` A privileged user who can manage the account information of all users, assign permissions, review uploaded artwork, and maintain the overall operation of the gallery.

These roles have different functional requirements and operating habits in actual use, so they are each designed with five targeted tasks:

1. Guest users login to the gallery and browse the homepage, to evaluate the gallery's login process and the intuitiveness of the homepage navigation;
2. Registered users upload a piece of artwork, including filling in information about the artwork and uploading a picture or video file, to evaluate the usability and interactive complexity of the upload page;
3. A registered user views and edits an uploaded artwork, to evaluate the functionality of the artwork details page and the ease of the editing process;
4. Administrator users manage the information of all registered users, the purpose of which is to evaluate the ease of use of the user management interface and the clarity of administrator permissions;
5. Guest users accessed the gallery and browsed the 3D gallery using different browsers including Chrome, Safari, Firefox and Edge to test the gallery's compatibility across these four browsers.

Each user rated their satisfaction with the system after each test on a scale of 1 to 5, with 1 representing very dissatisfied and 5 representing very satisfied.

Chapter 5

Results

5.1 First user study

The first user study revealed key problems in the initial design of the 3D-gallery, which centres on the following four areas:

1. Operational Complexity:

During initial testing, users generally reported that the process of uploading and editing artwork was too complicated. It involved multiple steps and lacked clear guidance. Although this gallery version was relatively comprehensive, the lack of guidance resulted in a high learning curve for first-time users. When browsing the 3D gallery, some users pointed out that the layout of the navigation bar was not intuitive enough. They also found that the location of some of the function buttons was not easy to see, particularly when it came to managing artworks and commenting on them. Users expressed a desire for a clearer navigation structure and guidance information.

2. Role ambiguity:

There was ambiguity regarding users' permissions for different roles, such as registered users and administrators. This lack of clarity was particularly evident when it came to managing and uploading artwork. Users were unable to discern the extent of their permissions, potentially leading to misuse or failure to carry out expected operations.

3. Browser compatibility problems:

Users experienced performance problems while using Safari, including long loading times and distorted images in 3D galleries, which had a significant impact on user experience. In older versions of Safari, some interactive func-

tions behaved irregularly, resulting in unresponsive clicks or incorrect function execution. In Firefox, a user reported abnormal behaviour of interactive elements such as buttons and sliders, making it difficult to perform certain operations, especially when making detailed adjustments and uploading files.

4. **Insufficient user feedback:**
   After performing key operations such as uploading or editing artwork, the gallery did not provide clear feedback. Users were unsure whether the operation was successful, affecting their confidence in the gallery's operation.

* * *

Four key improvements were made to the gallery based on user feedback:

1. The process of **uploading and editing artwork** has been improved to make it more efficient. Unnecessary steps have been reduced, and instructions have been added to help users complete tasks more easily and quickly. Tooltips are now available at important points to explain the meaning and results of each action. A "Help" button and instructions have been added to the navigation bar, allowing users to quickly understand how the gallery works. New users can also see an orientation page when they log in for the first time to introduce them to the main functions and operations of the gallery.
2. The user interface now clearly defines the **permissions for different roles** and provides hints and explanations on relevant pages. For instance, if a registered user attempts to access the administrator function, they will receive a permission prompt to prevent misuse.
3. By optimizing the code and adjusting the configuration parameters of Three.js, especially for the Safari browser, which brings **the loading speed and rendering effects on Safari** up to the same level as other browsers. The problem of anomalous interactive functions in some older versions of Safari has been addressed, allowing users to smoothly utilize all functions. A problem in Firefox where interactive elements were behaving abnormally, particularly during file uploads and detail adjustments, has been fixed.
4. After a user performs a critical operation such as uploading or editing, they receive **instant visual and textual feedback** confirming that the operation has been completed successfully.

5.2 Second user study

The second user study focused on validating that the upgraded gallery effectively resolved the problems identified in the initial study and made further refinements to improve overall user satisfaction. The findings indicate a significant enhancement in the overall user experience of the gallery, along with improved browser compatibility:

1. Users have generally provided feedback that the overall experience of the gallery has improved. The operational flow has become more concise and smooth, requiring less time for users to understand the gallery functions. Most users have a clearer understanding of roles and permissions and can easily distinguish and operate the functions of different roles. The changes in the satisfaction ratings of the six users between the two tests are shown in Table 5.1.
2. With the enhancements, the gallery's performance in Safari and other major browsers has become consistent. Most users report that the gallery performs uniformly across browsers, with only a few instances of minor rendering problems. The accessibility of the gallery is tested using the automated tool Lighthouse and the result can be seen in Figure 5.1.
3. Users are generally happy with how quickly they receive feedback from the gallery, especially when it comes to confirmation messages after important actions. The only suggestion for improvement is that the feedback messages for certain operations such as incorrect ones could be more detailed to assist users in quickly identifying the problem.
4. Two users suggest further optimizing the performance of the gallery for slow network conditions or large file sizes, particularly for file uploads and downloads. Users would like to see progress displays and smarter upload mechanisms, such as uploading files in chunks or loading data asynchronously, added in future iterations to improve the gallery's responsiveness and stability.

5.3 Main implementation results

The workflow for the main features of this 3D-gallery is:

Enter the gallery (Figure 5.2a) Users can enter the gallery by clicking the `'Enter Exhibition'` button or clicking the `'Help'` button to get help information about how to use this gallery. Upon entering the main display area of the

3D gallery (Figure 5.2b), the user can navigate through the entire gallery using the mouse and keyboard. Users can also view detailed information about a particular artwork and interact with it, including sharing on social media, rating and leaving a comment on the artwork, Figure 5.2c. There are five extended functions such as return to the homepage in the menu bar, Figure 5.2d.

Login and Register (Figure 5.2e) Users can access other features of the gallery after login. Unregistered users can sign up using the `'Create Account'` button. Once logged in, users can upload their work by selecting a blank display area in the gallery, Figure 5.2f. Users can also edit their work directly in the work details form, Figure 5.2c.

Manage artwork (Figure 5.2g) Users can view and manage all their uploaded works in the interface.

User management (Figure 5.2h) After logging into the gallery as an administrator, all registered users of this gallery can be managed.

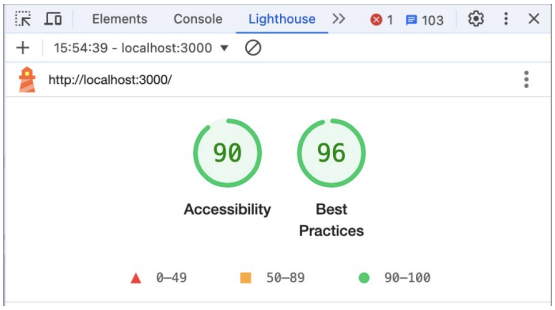
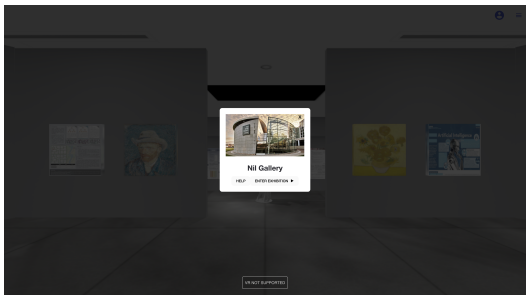


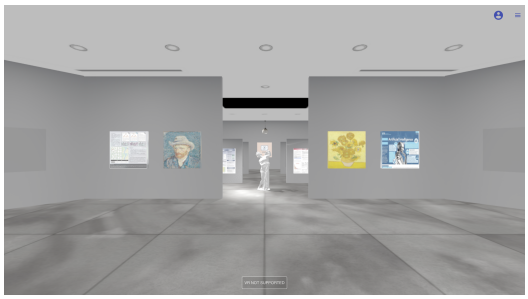
Figure 5.1 Homepage accessibility score.

	Rating		
User ID	**First Test**	**Second Test**	**Change**
User 1	3	4	+1
User 2	3	5	+2
User 3	4	5	+1
User 4	2	4	+2
User 5	3	4	+1
User 6	4	5	+1

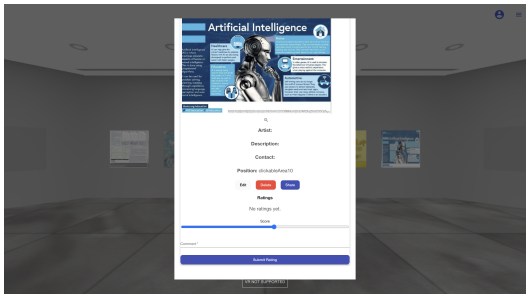
Table 5.1 User satisfaction rating.



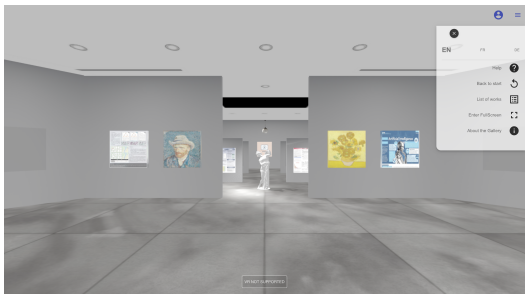
(a) Loading the gallery.



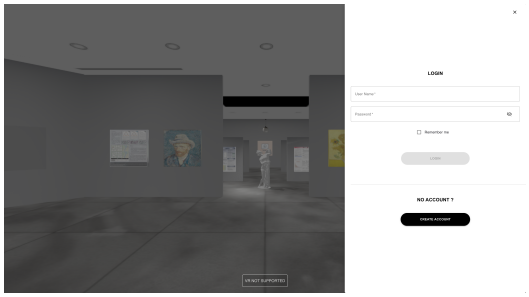
(b) Homepage.



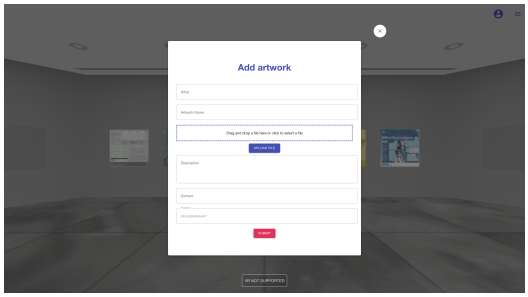
(c) Viewing artwork details.



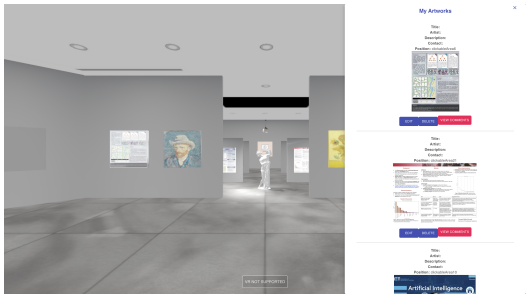
(d) Getting help.



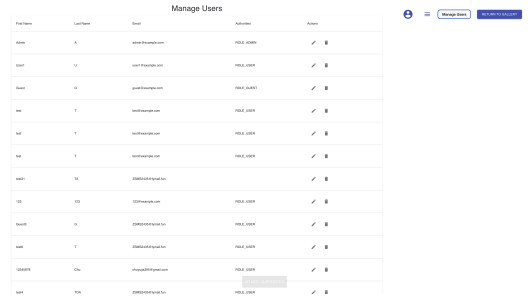
(e) Login or register.



(f) Uploading artwork.



(g) Users managing their artwork.



(h) Administrator managing users.

Figure 5.2 Screenshots.

Chapter 6

Analysis and evaluation

6.1 User study

The gallery has been improved based on two rounds of user studies. These enhancements have enhanced user experience and increased gallery stability and cross-platform consistency. User operations have become more convenient and intuitive by simplifying operational processes and clarifying role permissions. The first user study revealed problems with complex interactions, indicating that the gallery design lacked adequate user guidance and operational fluidity. This necessitated the simplification of processes and the enhancement of user operation guidance. It became clear that the user interface needed streamlining so that users with less technical expertise can navigate and use the gallery more effectively. The improved gallery interface is now more intuitive which significantly enhances the user experience, reduces the user's operational burden, and increases the gallery's usability. Users may still face confusion or errors in certain complex scenarios, such as handling large amounts of data or performing specific operations. It may be beneficial to introduce more guided UI designs, such as adding tooltips or stepwise instructions during key actions, to help users complete tasks more easily. Although the optimized gallery generally performs well, user research revealed a new demand for personalized user experiences. Some users expressed a desire for the gallery to offer more personalized operation suggestions or shortcuts based on their usage habits and preferences. This suggests that future optimizations could focus on adding intelligent and adaptive features based on user behaviour analysis, further enhancing the overall user experience.

Users reviewed the gallery's performance positively in terms of browser compatibility optimization. This is especially true for Safari which previously had problems. User feedback indicates that most of the Safari problems have been addressed. The current compatibility test will need updating as browser versions are continuously updated and new technologies are introduced. Maintaining the stability of the gallery requires constant monitoring and adjusting of the gallery's performance in different browsers as they evolve and technical standards change. Some users have reported performance problems in extreme cases, such as slow loading in low internet speed environments and lag during large file uploads. Gallery performance optimization is not limited just to compatibility, but also requires more comprehensive optimization in terms of resource management and data transfer efficiency. Future optimization efforts could include introducing techniques like progressive loading, asynchronous data processing, advanced file compression, or chunk uploading to enhance the gallery's stability and responsiveness in different network environments and improve the response speed.

The users' satisfaction ratings in the second test have a significant improvement with an average rating increase of approximately 42 % based on the user satisfaction rating scale analysis. This change suggests that the gallery's optimization has been widely acknowledged by users, particularly in terms of interface design, simplification of the operation process, and refinement of functions, leading to a notably enhanced user experience. The overall increase in user ratings demonstrates the effectiveness of the gallery's improvement efforts in various key functional areas, especially for users who had lower scores in the first test and experienced the most improvement in their ratings. However, despite the substantial overall improvement in ratings, there is still a small group of users who have experienced a less noticeable increase in satisfaction, indicating the need for greater attention to the specific needs and usage scenarios of these users in future optimizations to further enhance their experience.

6.2 Accessibility

The needs of a wide range of user groups are carefully considered during creating this 3D-gallery, including visually impaired, and colour-blind users. The Web Content Accessibility Guidelines (WCAG) 2.1 standard is strictly followed (W3C, 2018) to make a gallery accessible and usable to these users. Compared to

previous WCAG versions, WCAG 2.1 is an improved version that better addresses the accessibility needs of modern web applications, especially support for touch devices and complex interactive content. This standard sets a high bar for accessibility in web development and makes certain that this 3D-gallery meets the needs of modern users. These measures not only bring the gallery in line with the technical requirements for accessibility but also enhance the overall user experience, making the 3D-gallery accessible and usable by a wider range of users. The technical implementation provides full support for screen readers, including adding accessibility labels to all text content, 3D-elements, and interactive controls to correctly recognise and read these elements by commonly used screen readers such as NVDA and JAWS. For colour-blind users, the design process pays particular attention to the contrast of the colour scheme. All text-to-background contrast in the gallery is rigorously tested to confirm it meets the requirements of the WCAG 2.1 AA standard (text-to-background contrast of at least 4.5:1, and at least 3:1 for large text). Multiple types of colour blindness such as red-green and blue-yellow blindness are simulated for testing to confirm that users could identify the page content regardless of their visual conditions.

The test result of the accessibility has achieved a score of 90 5.1, which is a satisfactory score. However, in complex 3D-environments, certain accessibility requirements may be difficult to achieve perfectly. For example, the dynamic elements and interactive content in the 3D-gallery are still difficult for screen readers to support. Even with accessibility labelling, the ability of screen readers to effectively communicate this information still needs to be further verified. Lighthouse is effective in detecting common accessibility problems, but it cannot comprehensively capture all the accessibility problems, especially in complex 3D-content and interactions. The accessibility problems identified by the tool are often superficial, and deeper problems that affect the user experience still need to be discovered through actual user testing.

Visual perception by colour-blind users is affected by contrast, colour scheme, and texture selection. Relying solely on tools and simulation tests may result in designs that are still problematic in practice. User study is undoubtedly one of the most effective ways to assess accessibility. However, due to the limited range of users available during testing, hearing-impaired and colour-blind users could not be found to test this gallery. If the gallery is to be utilized in the community

in the future, colour-blind users would need to be invited to conduct the test to confirm the validity of these designs.

6.3 Design layout and visual effects

User feedback during user studies has shown that this project has achieved positive results regarding design layout and visual effects, especially in spatial organization, display of works, and visual consistency. The gallery's design layout mimics that of a real gallery, displaying different types or themes of works in different zones. This layout is intended to naturally guide users to explore the gallery content through rational spatial distribution. The exhibits are arranged based on eye level, spacing, and contrast principles to create an optimal display. This design strategy mirrors the traditional visual presentation of a gallery, preventing users from feeling overwhelmed and allowing them to focus easily on the artwork. This approach also proves effective in virtual environments, particularly for displaying delicate artworks and maximizing visual impact. The project underscores the importance of colour and lighting in a 3D-gallery, using a simple background design to direct the user's attention to the artwork. Maintaining graphical consistency is also a focus during the development of this gallery. However, the pursuit of consistency can lead to difficulties in user experience across different devices. While accommodating screen sizes, responsive design may compromise certain visual details or ease of interaction on smaller screens.

6.4 Technology selection

This project utilizes HTML, CSS, and JavaScript as the foundational technologies, and we relied on Three.js and React.js for performance. Three.js, a library dedicated to creating and displaying 3D-graphics, played a crucial role in the project. While it is a mature and widely used open-source library that integrates well with the React.js framework, optimizing performance for complex 3D-scenes posed difficulties. The learning curve for Three.js was steep initially, adding pressure to complete the project. However, Three.js ultimately performed well, meeting all project requirements and proving to be the best choice after carefully considering its advantages and disadvantages.

React's declarative and component-based architecture is well suited for building dynamically interacting user interfaces. The modular approach to development enhances code extensibility and maintainability, especially when used alongside Three.js. By combining React.js and Three.js, developers can leverage React's component-based architecture and state management, along with Three.js's robust 3D rendering capabilities, to build feature-rich and interactive modern web applications (Mardan, 2017). For instance, React.js can handle user input and interface updates, as well as manage and organize different parts of an application, while Three.js can draw and render 3D-scenes, allowing for complex 3D-animations and interactive effects 3. This synergy enables developers to create high-performance, highly interactive applications while ensuring code clarity and maintainability. However, when high-performance requirements are a priority, React's flexibility and rendering speed may not be as robust as Vue.js and Angular.js. In such cases, opting for a lighter framework like Vue.js may improve rendering speed and reduce development complexity. It's important to note that using these open-source libraries comes with the responsibility of regularly checking and updating dependencies to mitigate potential security vulnerabilities and performance problems during subsequent use and refinement of this project.

Java was chosen for the back-end. Separating the back-end and front-end development clarified the logic of the entire development process, reduced code maintenance complexity, and resulted in stable performance when handling large amounts of data. Managing library and framework versions in the Java ecosystem does increase the development time overhead. When development cycles are tight and flexibility is a top priority, consider Python to expedite development. However, Java is more capable of detecting errors at compile time, which is the best time to find and fix errors as early as possible in the development process. Comprehensive evaluation based on the specific needs and priorities of the project is required when choosing a back-end language.

JWT provides a convenient authentication solution but its lack of built-in expiration and revocation mechanisms increases administrative complexity. Nevertheless, JWT met the security requirements of this project, and it has performed well in managing moderate amounts of information. Overall, the chosen technologies have proven generally effective in this project, with no encountered incompatibility problems, successfully meeting the project's requirements.

6.5 Error analysis

This project initially aimed to develop in a virtual machine environment. However, the system response was too slow, especially when dealing with complex 3D graphics. It became evident that the interface was lagging and delayed, significantly impacting development efficiency. Virtual machines usually allocate GPUs through virtualization, which may cause performance loss and affect 3D rendering calculations and graphics processing when the virtual machine runs high-load graphics applications. Graphics hardware acceleration in virtual machines typically lags behind that of host machines, resulting in lower than desired graphics rendering and processing speeds. Despite attempts to increase the memory and number of CPU cores in the virtual machine to provide sufficient resources for handling graphics-intensive tasks, the results were unsatisfactory. The development process on the host machine improved significantly in terms of smoothness.

Users using the Safari browser reported encountering long loading times, distorted images, and unusual interactive features. It was determined that these problems might be caused by Safari's graphics processing engine not fully supporting certain WebGL features, leading to distorted image rendering and performance problems by analyzing the error alerts in the console. There were some delays in Safari's handling of event triggers and DOM updates, resulting in interaction anomalies. To address these problems, specific optimizations were implemented for Safari to avoid using WebGL features that are not fully supported or have poor performance on the browser.

Rendering a 3D scene requires careful consideration of the GPU's limit on the number of texture units that can be used simultaneously in each shader stage, which is typically around 16. This gallery attempted to bind more than 16 texture units at the same time in a single shader will result in WebGL throwing a warning or error. In a project using Three.js, this problem occurred when rendering walls with too many materials or textures, leading to the GPU's texture unit limit being exceeded. This project has reduced texture usage and optimized materials. To extend the gallery further, it is recommended to use the browser's built-in developer tools or Three.js' debugging tools to analyze texture usage and improve rendering performance and user experience.

User role rights management with JWT requires careful attention to the implications of updating the JWT library. An update to the library may introduce

improved security or functionality, along with changes in how certain functions are used or the removal of old APIs, which may make functions or methods that were previously relied upon no longer work properly.

Chapter 7

Conclusions

7.1 Project summary

The primary goal of this project is to create a 3D-gallery accessible via a web browser, which is used to showcase student work including 3D-models, digital work, multimedia content, and interactive simulations, helping the University display its own students' projects. Users can explore the gallery conveniently and immerse themselves in student work, interacting with different features. This project utilizes modern web technologies such as React.js and Three.js to build an interactive virtual environment. The front-end uses React.js for the user interface, Three.js for managing 3D-scenes, and the back-end is built using the Spring Boot framework, incorporating Spring Security for user authentication and authorization, JWT for secure user authentication and session management, and PostgreSQL for database management. Compatibility with major browsers such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari is considered, and this project is concerned with delivering consistency and smooth operation across different platforms and devices. The development process requires advanced graphics processing power and technical setup to optimise performance. This project integrates interactive features and secure storage for user-uploaded content to enhance user engagement and educational value. This is achieved using JavaScript libraries, frameworks like React.js and Three.js, and responsive design techniques. Security measures including user authentication using JWT are implemented to safeguard user data and privacy. This 3D-gallery improves the accessibility of student work, and provides educational organizations with a platform to exhibit outstanding student work, thereby enhancing the reputation and appeal of the University.

7.2 Future work

The gallery's interactive features can be enhanced in the future. For instance, a virtual tour function can be added, allowing users to explore different exhibition areas through predefined routes. A real-time multiplayer interactive function can also be implemented, enabling multiple users to enter the gallery simultaneously and interact and communicate in real-time. The gallery needs to handle high loads as the gallery content and user visits increase. Although performance optimization strategies like efficient database management and caching mechanisms have been considered in the current design, there is still room for improvement. In the future, more advanced performance monitoring tools can be introduced to analyze gallery bottlenecks in real-time. Technologies like WebAssembly can also be considered to improve further the efficiency and responsiveness of 3D-rendering (Haas et al., 2017). AI-driven tools can be integrated, such as using machine learning algorithms to recommend relevant student work or exhibition content based on users' browsing behaviour and interests. AI can also be employed to automatically analyze works uploaded by users, providing intelligent labelling, classification, or evaluation functions to assist users in displaying and managing their works, making the gallery more intelligent and personalized.

7.3 Skills and knowledge gained

I learned about the componentized architecture of React.js, which helped me manage state and data flow more effectively. I also mastered using React Hooks to manage component lifecycle and state changes, improving my skills in single-page application (SPA) development. In 3D-rendering, I learned to create and manage complex 3D-scenes in the browser using Three.js, including working with geometry, materials, lighting, and camera settings. I gained in-depth knowledge of the Spring Boot framework in Java development, building RESTful APIs and implementing user authentication and authorization using JSON Web Token (JWT) technology. I learned to use Spring Data JPA to interact with a PostgreSQL database. I also learned performance optimization techniques, such as handling high loads and complex rendering scenarios, and explored using WebAssembly technology to improve 3D-rendering efficiency.

7.4 Reflections

The most enjoyable part of the project for me is designing and building the 3D-gallery, especially when implementing the user interface and 3D-rendering using React.js and Three.js. It is rewarding to see my ideas gradually becoming a reality and providing users with an immersive platform to display their work. However, this project had some regrets, mainly due to underestimating the complexity of cross-platform compatibility, performance optimization, and Java version incompatibility in the early stages. Dealing with compatibility problems across devices and browsers is much more complicated than expected, especially in high-load 3D-rendering scenarios. Compatibility problems caused by Java version inconsistencies also led to a lot of debugging and adaptation work.

References

Adam, S.I., Moedjahedy, J.H. and Maramis, J., 2020. Restful web service implementation on unklab information system using json web token (jwt). *2020 2nd international conference on cybernetics and intelligent system (icoris)*. IEEE, pp.1–6 (cit. on p.13).

Aggarwal, S. et al., 2018. Modern web-development using reactjs. *International journal of recent research aspects*, 5(1), pp.133–137 (cit. on p.6).

Ahmed, S. and Mahmood, Q., 2019. An authentication based scheme for applications using json web token. *2019 22nd international multitopic conference (inmic)*. IEEE, pp.1–6 (cit. on pp.8, 13).

Anyuru, A., 2012. *Professional webgl programming: developing 3d graphics for the web*. John Wiley & Sons (cit. on p.11).

Arnold, K., Gosling, J. and Holmes, D., 2005. *The java programming language*. Addison Wesley Professional (cit. on p.11).

Babich, N., 2019. *The 12 do's and don'ts of web design | adobe xd ideas* [Online]. Ideas. Available from: https://xd.adobe.com/ideas/principles/web-design/12-dos-donts-web-design-2/ (cit. on p.15).

Brown, M., 2024. Glassmorphism: definition and best practices. *Nielsen norman group* [Online], June. Accessed: 2024-06-07. Available from: https://www.nngroup.com/articles/glassmorphism/ (cit. on p.16).

Chisholm, W.A. and Henry, S.L., 2005. Interdependent components of web accessibility. *Proceedings of the 2005 international cross-disciplinary workshop on web accessibility (w4a)*, pp.31–37 (cit. on p.9).

Colblindor, 2016. *Coblis — color blindness simulator – colblindor* [Online]. Color-blindness.com. Available from: https://www.color-blindness.com/coblis-color-blindness-simulator/ (cit. on p.15).

Danchilla, B. and Danchilla, B., 2012. Three. js framework. *Beginning webgl for html5*, pp.173–203 (cit. on p.7).

Diniz-Junior, R.N., Figueiredo, C.C.L., Russo, G.D.S., Bahiense-Junior, M.R.G., Arbex, M.V., Dos Santos, L.M., Da Rocha, R.F., Bezerra, R.R. and Giuntini, F.T., 2022. Evaluating the performance of web rendering technologies based on javascript: angular, react, and vue. *2022 xvliii latin american computer conference (clei)*. IEEE, pp.1–9 (cit. on p.10).

Dirksen, J. et al., 2013. *Learning three. js: the javascript 3d library for webgl*. Vol. 3. Packt Publishing Livery Place, UK (cit. on pp.7, 10).

Dirksen, J. et al., 2014. *Three. js essentials*. Packt Publishing (cit. on p.7).

Evans, A., Romeo, M., Bahrehmand, A., Agenjo, J. and Blat, J., 2014. 3d graphics on the web: a survey. *Computers & graphics*, 41, pp.43–61 (cit. on p.6).

Falk, J.H. and Dierking, L.D., 2016. *The museum experience revisited*. Routledge (cit. on p.4).

Farrell, S., 2009. Api keys to the kingdom. *Ieee internet computing*, 13(5), pp.91–93 (cit. on p.14).

Fedosejev, A., 2015. *React. js essentials*. Packt Publishing Ltd (cit. on p.6).

Fett, D., Küsters, R. and Schmitz, G., 2016. A comprehensive formal security analysis of oauth 2.0. *Proceedings of the 2016 acm sigsac conference on computer and communications security*, pp.1204–1215 (cit. on p.14).

Filip, P. and Čegan, L., 2020. Comparison of mysql and mongodb with focus on performance. *2020 international conference on informatics, multimedia, cyber and information system (icimcis)*. IEEE, pp.184–187 (cit. on p.12).

Gajewski, M. and Zabierowski, W., 2019. Analysis and comparison of the spring framework and play framework performance, used to create web applications in java. *2019 ieee xvth international conference on the perspective technologies and methods in mems design (memstech)*. IEEE, pp.170–173 (cit. on p.8).

Haas, A., Rossberg, A., Schuff, D.L., Titzer, B.L., Holman, M., Gohman, D., Wagner, L., Zakai, A. and Bastien, J., 2017. Bringing the web up to speed with webassembly. *Proceedings of the 38th acm sigplan conference on programming language design and implementation*, pp.185–200 (cit. on p.36).

Jadhav, M.A., Sawant, B.R. and Deshmukh, A., 2015. Single page application using angularjs. *International journal of computer science and information technologies*, 6(3), pp.2876–2879 (cit. on p.7).

Jung, M.-G., Youn, S.-A., Bae, J. and Choi, Y.-L., 2015. A study on data input and output performance comparison of mongodb and postgresql in the big data environment. *2015 8th international conference on database theory and application (dta)*. IEEE, pp.14–17 (cit. on p.13).

Khoirom, S., Sonia, M., Laikhuram, B., Laishram, J. and Singh, T.D., 2020. Comparative analysis of python and java for beginners. *Int. res. j. eng. technol*, 7(8), pp.4384–4407 (cit. on p.11).

Kumar, A. and Singh, R.K., 2016. Comparative analysis of angularjs and reactjs. *International journal of latest trends in engineering and technology*, 7(4), pp.225–227 (cit. on p.6).

Makris, A., Tserpes, K., Spiliopoulos, G., Zissis, D. and Anagnostopoulos, D., 2021. Mongodb vs postgresql: a comparative study on performance aspects. *Geoinformatica*, 25, pp.243–268 (cit. on pp.12, 13).

Mardan, A., 2017. *React quickly: painless web apps with react, jsx, redux, and graphql*. Simon and Schuster (cit. on pp.10, 20, 32).

Mason, D.D. and McCarthy, C., 2006. 'the feeling of exclusion': young peoples' perceptions of art galleries. *Museum management and curatorship*, 21(1), pp.20–31 (cit. on p.3).

Mikowski, M. and Powell, J., 2013. *Single page web applications: javascript end-to-end*. Simon and Schuster (cit. on p.7).

Nelson, R., Shukla, A. and Smith, C., 2020. Web browser forensics in google chrome, mozilla firefox, and the tor browser bundle. *Digital forensic education: an experiential learning approach*, pp.219–241 (cit. on p.9).

Nielsen, J., 1994. *10 usability heuristics for user interface design* [Online]. Nielsen Norman Group, April. Available from: https://www.nngroup.com/articles/ten-usability-heuristics/ [Accessed 11 April 2023] (cit. on p.15).

Obe, R.O. and Hsu, L.S., 2017. *Postgresql: up and running: a practical guide to the advanced open source database*. " O'Reilly Media, Inc." (cit. on p.8).

Palanisamy, S. and SuvithaVani, P., 2020. A survey on rdbms and nosql databases mysql vs mongodb. *2020 international conference on computer communication and informatics (iccci)*. IEEE, pp.1–7 (cit. on p.13).

Parisi, T., 2012. *Webgl: up and running*. " O'Reilly Media, Inc." (cit. on p.7).

Prechelt, L., 2000. An empirical comparison of c, c++, java, perl, python, rexx and tcl. *Ieee computer*, 33(10), pp.23–29 (cit. on p.12).

Prensky, M., 2001. Digital natives, digital immigrants part 2: do they really think differently? *On the horizon*, 9(6), pp.1–6 (cit. on p.3).

Rautmare, S. and Bhalerao, D.M., 2016. Mysql and nosql database comparison for iot application. *2016 ieee international conference on advances in computer applications (icaca)*. IEEE, pp.235–238 (cit. on pp.12, 13).

Robbins, J.N., 2012. *Learning web design: a beginner's guide to html, css, javascript, and web graphics*. " O'Reilly Media, Inc." (cit. on p.7).

Saks, E., 2019. Javascript frameworks: angular vs react vs vue. (cit. on pp.10, 11).

Schweibenz, W., 1998. The" virtual museum": new perspectives for museums to present objects and information using the internet as a knowledge base and communication system. *Isi*, 34, pp.185–200 (cit. on p.3).

Stroustrup, B., 1986. An overview of c++. *Proceedings of the 1986 sigplan workshop on object-oriented programming*, pp.7–18 (cit. on p.12).

Tatroe, K. and MacIntyre, P., 2020. *Programming php: creating dynamic web pages*. O'Reilly Media (cit. on p.11).

W3C, 2018. *Web content accessibility guidelines (wcag) 2.1* [Online]. W3.org, June. Available from: https://www.w3.org/TR/WCAG21/ (cit. on pp.9, 15, 29).

WebAIM, 2019. *Webaim: contrast checker* [Online]. Webaim.org. Available from: https://webaim.org/resources/contrastchecker/ (cit. on p.15).

Wieruch, R., 2017. *The road to react: your journey to master plain yet pragmatic react. js*. Robin Wieruch (cit. on p.20).

Worsley, J. and Drake, J.D., 2002. *Practical postgresql*. " O'Reilly Media, Inc." (cit. on p.8).

Xanthoudaki, M., 1998. Educational provision for young people as independent visitors to art museums and galleries: issues of learning and training. *Museum management and curatorship*, 17(2), pp.159–172 (cit. on p.2).

Yablonski, J., 2020. *Laws of ux* [Online]. O'Reilly, December. Available from: https://lawsofux.com/en/ [Accessed 31 July 2022] (cit. on p.16).

You, E. and Vite Team, the, 2024. *Why vite*. https://vitejs.dev/guide/why.html. Accessed: 2024-08-07 (cit. on p.20).

Appendix A

Code structure and features:

The back-end of this project supports the front-end by managing data and handling user authentication including the administrator, user and visitor identities.

1. Java Frameworks and Libraries:

`Spring Boot:` Used to create the back-end RESTful API. It simplifies the development and deployment of the application.

`Spring Security:` Manages user authentication and authorization.

`JSON Web Token (JWT):` Secure user authentication and session management.

2. Structure and features:

`config package:`

`'EncoderConfiguration.java':` Configures the password encoder to securely encrypt user passwords. The configuration class defines a `'PasswordEncoder'` Bean that uses the BCrypt hashing algorithm to encrypt and verify passwords.

`'FileUploadConfig.java':` Configures settings related to file uploads to maintain safe and secure storage. A `'MultipartConfigElement'` Bean sets the maximum file size and the maximum request size to prevent oversized file uploads.

`'JWTAuthenticationFilter.java':` Configures the JWT authentication filter to intercept and validate JWT tokens in user requests. The filter extracts the JWT token from the request header, checks its validity, and if it's valid, stores the user information in the `'SecurityContextHolder'` for future requests.

'JWTTokenHelper.java': A helper class responsible for generating and validating JWT tokens for user authentication and authorization. It includes functions to generate tokens, extract username and role information from tokens, validate token integrity, and check for token expiration.

'RestAuthenticationEntryPoint.java': Configures the handling logic for unauthenticated users attempting to access protected resources, returning appropriate HTTP status codes and error messages. This component sends a 401 Unauthorized response when a user tries to access protected resources without authentication.

'SecurityConfiguration.java': This component sets up the security policy for the entire application, with detailed settings for user authentication and authorization. The `'SecurityFilterChain'` is configured to define which URLs require authentication, and which ones are publicly accessible, and it also sets up the JWT authentication filter and password encryption.

'WebConfig.java': This component sets up web-related settings, including Cross-Origin Resource Sharing (CORS) configurations, to facilitate proper communication between the front-end and back-end. It also involves configuring resource handlers to map specific URL paths to the file upload directory and thumbnail directory.

controller package:

'AdminController.java': Handles requests related to admin operations, including user management tasks such as retrieving all users, updating user information, and deleting users.

'ArtworkController.java': Manages requests related to artwork:

- Adds artwork by receiving parameters like artist name, artwork name, file, description, contact information, and location, and then saving the new artwork.
- Retrieves all artwork and returning their information.
- Retrieves artwork of the currently authenticated user.
- Retrieves artwork thumbnails based on the file name.
- Retrieves artwork based on their location.
- Retrieves artwork based on their ID.
- Generates and returning share links for artwork.

- Allows users to add ratings and comments to artwork and retrieve all relevant ratings and comments based on the artwork ID.
- Updates specified artwork information, including re-uploading files.
- Deletes specified artwork, which can only be done by admins and artwork owners.

`'AuthenticationController.java':` Handles requests related to user authentication and authorization:

- Registers a user by receiving user information, saving the new user, and encrypting their password.
- Logs a user in by verifying their credentials and generating a JWT token.
- Retrieves detailed information about the currently authenticated user.
- Updates information of the currently authenticated user.
- Deletes specified users (administrators only).

`'FileController.java':` Manages requests for uploading and downloading files, providing secure storage and access. It retrieves file content by name and supports video and PDF files.

`'ThumbnailController.java':` Handles requests for retrieving thumbnails, fetching thumbnail content based on the file name, and maintaining secure storage and access for thumbnails.

`model package:`

`'Artwork.java':` Defines the artwork entity and its attributes, including artwork ID, artist name, artwork name, file path, description, contact information, location, and thumbnail path. This class also includes a many-to-one relationship mapping with the user.

`'Authority.java':` Defines the entity for user authority, including role code and role description. It implements the `'GrantedAuthority'` interface, utilized for role-based access control in Spring Security.

`'User.java':` Defines the user entity and its attributes, including user ID, username, password, creation time, update time, first name, last name, email, phone number, enabled status, and authorities. This class implements the `'UserDetails'` interface for Spring Security user authentication.

'Rating.java': Represents the user's ratings and comments on the artwork, establishing an association between the `'Artwork'` and `'User'` entities.

repository package:

'ArtworkRepository.java': This defines the database operations for the Artwork entity. It extends JpaRepository, providing create, read, update and delete operations for the database. Specific functions include finding a list of artwork by their location and finding a list of artwork by user ID.

'UserDetailsRepository.java': This defines the database operations for the User entity. It also extends JpaRepository, providing create, read, update and delete operations for the database. Specific functions include finding user information by username.

'RatingRepository.java': It is used to manipulate the Rating entity class in the database, providing basic functionality and retrieving all relevant ratings and reviews based on the artwork's ID.

'AuthenticationRequest.java': This defines the structure for authentication requests. It contains fields for the username and password, used to transmit authentication information during user login.

responses package:

'LoginResponse.java': Defines the structure for login responses, containing fields for the JWT token, username, email, user ID, and role information, which are returned to the frontend for user session management.

'UserInfo.java': Defines the structure for user information, containing the user's first name, last name, username, email, and role information, and is used to return detailed user information.

service package:

'artworkervice.java': Defines the business logic interface related to artwork. Specific functionalities include:

- Saves artwork information and files, generates unique file names, and stores files. It also includes generating thumbnails and file paths.
- Fetches all artwork information from the database.
- Finds artwork based on location.

- Finds all artwork created by a specific user based on user ID.
- Deletes artwork based on artwork ID.
- Updates artwork information, including re-uploading files and updating fields.

'artworkerviceImpl.java': Implements the 'artworkervice' interface, handling the logic for saving, retrieving, updating, and deleting artwork, including file storage, thumbnail generation, and handling of video and PDF files. It provides functionality for associating ratings with a user's artwork and saving them to the database. It also supports retrieving and returning all ratings associated with a specific artwork.

'CustomUserService.java': Implements interface for Spring Security user authentication, providing functions for loading user information, deleting users, and updating user information.

'Gallery3dApplication.java': This is the main entry point of the project, containing the main method used to launch the application. This class implements the 'CommandLineRunner' interface and creates necessary directories upon application startup such as the file upload directory.

'application.properties': Contains key configuration parameters for the application, including:

- Database Connection Information: Configures the PostgreSQL database connection with URL, username, password, and driver class name.
- JPA Configuration: Sets Hibernate database dialect, enables SQL statement logging and automatically updates the database schema.
- JWT Configuration: Manages JWT settings such as the application name, secret key, and token expiration time
- File Upload Configuration: Defines maximum file upload and request sizes to support large file uploads.
- Directory Configuration: Specifies directories for file uploads and thumbnails, confirming these directories exist and are writable.

A.1 Implementing the interface

1. Java Frameworks and Libraries:

`React.js`: Utilized for building the user interface components to achieve efficient rendering and state management.

`Three.js`: A JavaScript library used to create and display animated 3D-graphics in the web browser. It is integral to the development of the 3D-gallery scenes and animations.

2. Structure and features:

`The main files in the root directory:`

`'index.html'`: The main page of the application is responsible for loading and initializing the React application. It includes a div container with the id `'root'` where the React application is mounted. The page header also links to `'FontAwesome'` and a custom CSS file.

`'package.json'`: Manages this project's dependencies and scripts; defines the project name, version, type, dependencies, and common script commands, such as starting the development server, building the project, running ESLint, and previewing the build results.

`'vite.config.js'`: Configuration file for Vite. Contains plugin configurations, such as `'@vitejs/plugin-react'`, as well as configurations for the development server (port set to 3000, and proxy configurations for cross-domain problems).

`modules folder:` contains several JavaScript files, each responsible for a different aspect of the gallery's functionality:

`'Artwork.jsx'`: This component is used to display and manage individual artwork, handling the artwork display and interaction. This component utilizes React's `'AuseState'` and `'useEffect'` hooks to manage the state of the component, including the artist's name, artwork name, description, and contact information. Users can upload images, videos, and PDF files, and a preview is displayed after the upload. The artwork information and files are sent to the server through form submission, and the server response is processed.

`'ArtworkDetails.jsx'`: Users can use this component to view and manage details of individual artwork, edit artwork information, delete artwork, share artwork, and view ratings and comments. It also offers

a file preview function, supporting different formats including images, videos, and PDFs, and provides zoom functions. Users can also rate and comment on artwork and generate sharing links for social media platforms.

`'ArtworkPage.jsx':` This component is used to display the details of a single artwork. It retrieves the artwork details from the server using the artwork ID and then displays an image or video based on the file type. It also shows the artist's name, description, contact information, and location.

`'createCeiling.js':` This component creates and oversees ceiling objects within a 3D-scene. It loads multiple textures including color, displacement, ambient light, luminosity, metallicity, normals, and roughness. Additionally, it allows for setting how the texture repeats and adjusting other parameters. The tool creates a planar geometry and applies a texture material, before adding the ceiling object to the 3D-scene.

`'loadCeilingLamp.js':` This component is used to load and manage the ceiling lamp model in the 3D-scene, use `'GLTFLoader'` to load the 3D-model, set the initial position and scaling of the lamp, add the lamp model to the 3D-scene, and utilize a GUI library to create a control interface for real-time position adjustments of the lamp.

`'setupEventListeners.js':` This component is used to set up event listeners in the scene such as mouse and keyboard events. It adds keyboard and mouse event listeners to manage user input and interaction. It also uses ray casting on mouse clicks to detect where the user clicked and perform the appropriate action.

`'setupLighting.js':` This component is used to set up lighting in a 3D-scene using the THREE.js library. It adds ambient and spotlights and provides a graphical user interface (GUI) for adjusting the properties of these lights. The main functions include initializing the GUI interface, adding ambient lights and their properties for dynamic adjustment, creating multiple spotlights with customizable properties, and adding these properties to the GUI for dynamic adjustment. It also allows for adding spotlights at specific locations to simulate ceiling lights and statue lights.

'ManageUsers.jsx': This component is responsible for creating a user management interface. It enables administrators to view, edit, and delete user information. The main functions include retrieving and presenting all user information in a table, offering forms for editing user details, allowing user deletion, and facilitating interaction with the back-end API through the Axios library.

'movement.js': It outlines the game state, keyboard input handling, and methods for updating camera movement and detecting collisions. Key functions involve tracking pressed keys, updating camera movement, moving the camera based on keyboard input, detecting camera collisions with walls, and resetting the camera position if a collision occurs.

'Navbar.jsx': The navigation bar is designed to offer different functional buttons for user interaction, such as uploading works, login, registration, account information, user management, and other related functions. It adjusts the display of buttons based on the user's identity, manages button click events to show relevant modal boxes, communicates with the global state using the redux library, and provides debugging information upon component mounting and updates.

'scene.js': It is used to set up the infrastructure for a 3D-scene, which includes configuring cameras, renderers, and controllers, as well as providing animation loops. The main functions involve initializing the camera and renderer, handling window size changes, defining animation loops to continuously render the scene, managing mouse events to control viewpoint movement, enabling and disabling mouse events through overlays, and controlling the interactive state of the scene.

'Sidebar.jsx': This component creates a sidebar that offers navigation and 'help' information. Its main functionality includes showing and hiding the sidebar, displaying help dialogues when users click, providing navigation buttons for going back to the home page and entering full-screen mode, showing detailed help information through dialogues, and defining click event handlers for each function button.

'createWalls.js': This component is used to create walls and clickable areas in a 3D-scene. Its main functions include loading wall textures

and creating multiple wall meshes, adding these walls to the scene, defining clickable areas and adding interaction event handling, detecting mouse clicks via raycaster, and displaying information on the walls or performing other interactions.

'APP.jsx':

This component serves as the main application component. It is responsible for initializing the 3D-scene and managing different views and states, including user authentication, artwork uploading, editing, and deleting functions.

It contains several state and effect hooks to manage the initialization of the 3D-scene, user authentication state restoration, and auto-logout functionality.

It uses the `'Navbar.jsx'` component for navigation and the `'Artwork.jsx'`, `'ArtworkDetails.jsx'`, `'LoginPage.jsx'`, `'RegisterPage.jsx'`, and `'AccountPage.jsx'` components to manage different functions.

This component is used to initialize and manage 3D-scenes in Three.js, including cameras, renderers, lights, and objects.

It manages user interaction events, such as clicking on artwork to display details, uploading new artwork, and editing and deleting artwork.

It is also used for managing user login and registration status, and for displaying the appropriate modal windows for actions.

'main.jsx': This is the application's entry file. It renders the `'App.jsx'` components to the root element using ReactDOM and wraps Redux's Provider for global state management.

pages folder:

'LoginPage.jsx' and 'RegisterPage.jsx': These handle the user login and registration pages, respectively. They use form components and input validation logic to check for valid user input. Upon successful login or registration, they update the global state and close the respective modal windows.

'AccountPage.jsx': The component allows users to view and edit their personal information, manage uploaded artwork, and view ratings and comments on their works. Users can modify their data, preview and

edit artwork, or delete them. It interacts with the back-end via an API to dynamically update user data.

`Redux folder:`

`'authenticationService.jsx':` This component defines API requests related to user authentication including user login, registration, fetching, and updating user information. It contains key functions such as `'userLogin'` for handling user login operations, `'addArtwork'` for managing the addition of new artwork by the user, `'userRegister'` for processing user registration, `'fetchUserInfo'` for fetching user information, and `'updateUserInfo'` for updating user information.

`'authActions.js':` This component defines Redux actions related to user authentication. Important actions include `'authenticate'`, which initiates the authentication process, `'authSuccess'`, which handles successful authentication and stores user data in 'localStorage' for persistence, `'authFailure'`, which manages authentication failure scenarios, and `'logout'`, which handles user logout.

`'store.js':` This component configures the Redux store, combining middleware such as `'redux-thunk'` for handling asynchronous operations. It serves as the central repository for the application's state.

`'types.js':` This defines constant types for Redux actions to maintain consistency and avoid typos in action types.

`'auth.js':` It contains the Redux reducer function for authentication. It updates the authentication-related state based on different action types, including `'AUTH_REQ'` for authentication requests, `'AUTH_SUCCESS'` for successful authentication, `'AUTH_FAILURE'` for authentication failures, and `'LOGOUT'` for user logout.

`'index.js':` It combines multiple reducers functions into a single root reducer using `'combineReducers'`. This allows all pieces of the application's state to be managed correctly.

`public folder:` This is where public resource files such as images, 3D-models, and audio files are stored.

Appendix B

Ethics information

B.1 Instructions given to participants

Participants in the user studies are informed about the purpose of the research, which is to evaluate and optimize the gallery's user experience, functionality, and cross-platform compatibility. Participants are assigned different user roles including visitor, registered user, and administrator and complete specific tasks based on their roles. These tasks are designed to test the gallery's navigation smoothness, interaction complexity, role permissions clarity, and consistent performance across different browsers. All participants are informed that their participation is voluntary and that they can withdraw from the study at any time. Their data would be kept strictly confidential and used solely for the analysis of this research.

B.2 Notes from the user study

B.2.1 Pilot Study

The pilot study involved 6 participants (n=6), including both general users and those with technical backgrounds. The goal of the pilot study is to identify and address any design problems before moving on to the main study. Participants carry out tasks based on their roles during the pilot study and provide feedback on system performance, interface design, interaction experience, and accessibility. Based on this feedback, adjustments are made to this gallery interface and task flow to improve user experience and operational smoothness.

B.2.2 Second Study

After making changes to this gallery, another round of user studies is conducted with the same group of participants from the first study (n=6). The focus of this study is to assess the effectiveness of the improvements based on the findings from the first round. Participants once again perform tasks related to their assigned roles, such as browsing the gallery homepage, uploading and editing artwork, managing user information, and testing the gallery's compatibility across different browsers. The study results indicate significant improvements in intuitiveness and user satisfaction. However, the study also identifies areas where further improvements can be made, such as adding more user guidance and prompts during critical steps.

B.3 Participants' information sheet

Participant Information Sheet

Project title:	Browser-based 3D gallery to showcase student work
Principal investigator:	Brian Mitchell
Researcher collecting data:	Yikun Zhao

This study was certified according to the Informatics Research Ethics Process, reference number 710059. Please take time to read the following information carefully. You should keep this page for your records.

Who are the researchers?

Researcher: Yikun Zhao

Supervisor: Brian Mitchell

What is the purpose of the study?

The primary purpose of this study is to develop a browser-based 3D gallery to showcase student project work.

Why have I been asked to take part?

You have been asked to take part because this study targets students, particularly those involved in projects such as creating a browser-based 3D-gallery to showcase student work. This project is suitable for students of any mode, including full-time, part-time, and distance learning, as indicated in the project description.

Do I have to take part?

No – participation in this study is entirely up to you. You can withdraw from the study at any time, without giving a reason. Your rights will not be affected. If you wish to withdraw, contact the PI. We will stop using your data in any publications or presentations submitted after you have withdrawn consent. However, we will keep copies of your original consent, and of your withdrawal request.

What will happen if I decide to take part?

Specify:

- Kinds of data being collected: Feedback on the creation and showcasing of the browser-based 3D-gallery, your project experience, technical challenges, and issues related to accessibility and intuitiveness in the design process.
- Means of collection: Through questionnaires, interviews, and focus groups.
- Duration of session: Each session will last approximately 1 hour.
- If participant audio/video is being recorded: Participant audio and video will be recorded for further analysis.
- How often, where, when: Sessions will be held bi-weekly, conducted online, with specific times scheduled based on participants' availability.

Are there any risks associated with taking part?

There are no significant risks associated with participation.

Are there any benefits associated with taking part?

No, there are not any benefits associated with taking part.

What will happen to the results of this study?

The results of this study may be summarised in published articles, reports and presentations. Quotes or key findings will be anonymized: We will remove any information that could, in our assessment, allow anyone to identify you. With your consent, information can also be used for future research. Your data may be archived for a minimum of two years.

Data protection and confidentiality.

Your data will be processed in accordance with Data Protection Law. All information collected about you will be kept strictly confidential. Your data will be referred to by a unique participant number rather than by name. Your data will only be viewed by the researcher/research team: Yikun Zhao and Brian Mitchell.

All electronic data will be stored on a password-protected encrypted computer, on the School of Informatics' secure file servers, or on the University's secure encrypted cloud storage services (DataShare, ownCloud, or Sharepoint) and all paper records will be stored in a locked filing cabinet in the PI's office. Your consent information will be kept separately from your responses in order to minimise risk.

What are my data protection rights?

The University of Edinburgh is a Data Controller for the information you provide. You have the right to access information held about you. Your right of access can be exercised in accordance Data Protection Law. You also have other rights including rights of correction, erasure and objection. For more details, including the right to lodge a complaint with the Information Commissioner's Office, please visit www.ico.org.uk. Questions, comments and requests about your personal data can also be sent to the University Data Protection Officer at dpo@ed.ac.uk.
For general information about how we use your data, go to: edin.ac/privacy-research

Who can I contact?

If you have any further questions about the study, please contact the lead researcher, [name: Yikun Zhao, contact details: s2497078@ed.ac.uk].
If you wish to make a complaint about the study, please contact inf-ethics@inf.ed.ac.uk. When you contact us, please provide the study title and detail the nature of your complaint.

Updated information.

If the research project changes in any way, an updated Participant Information Sheet will be made available on http://web.inf.ed.ac.uk/infweb/research/study-updates.
[**NB:** the PI should notify the Ethics panel on inf-ethics@ed.ac.uk to upload any update PIS to the website]

Consent

By proceeding with the study, I agree to all of the following statements:

- I have read and understood the above information.
- I understand that my participation is voluntary, and I can withdraw at any time.
- I consent to my anonymised data being used in academic publications and presentations.
- I allow my data to be used in future ethically approved research.

[Button here named "I agree" or "take me to the survey"]

Participant number:_____

B.4 Participants' consent form

Participant Consent Form

Project title:	Browser-based 3D gallery to showcase student work
Principal investigator (PI):	Brian Mitchell
Researcher:	Yikun Zhao
PI contact details:	brian.x.mitchell@ed.ac.uk

By participating in the study you agree that: [Required elements of the study should go in this section, and optional ones should go under the "tick yes or no" section below.]

- I have read and understood the Participant Information Sheet for the above study, that I have had the opportunity to ask questions, and that any questions I had were answered to my satisfaction.
- My participation is voluntary, and that I can withdraw at any time without giving a reason. Withdrawing will not affect any of my rights.
- I consent to my anonymised data being used in academic publications and presentations.
- I understand that my anonymised data will be stored for the duration outlined in the Participant Information Sheet.

Please tick yes or no for each of these statements.

1.	I agree to being audio recorded. [delete as appropriate].	**Yes**	**No**
2.	I agree to being video recorded. [delete as appropriate].	**Yes**	**No**
3.	I allow my data to be used in future ethically approved research.	**Yes**	**No**
4.	I agree to take part in this study.	**Yes**	**No**

Name of person giving consent	Date dd/mm/yy	Signature

Name of person taking consent	Date dd/mm/yy	Signature