### Presentation Script

**Cover Page (Richard, 1 min)**

Good [morning/afternoon], everyone. Thank you for joining us today. My name is Richard Wang, and on behalf of my team, I’m excited to present our financial portfolio management application. In the next 20 minutes, we’ll provide an in-depth overview of our project, including its objectives, technical implementations, and future plans.

**Team Members (Richard, 1 min)**

Before diving in, let me introduce our team: - Sharon Shang Guan handled the database design and implementation. - Kelvin Chang managed the system architecture and ensured the smooth integration of different system layers. - Anthony Wang developed the frontend interface and conducted thorough testing. - And myself, Richard Wang—I managed backend development and API integration.

Together, we’ve worked collaboratively to ensure the application’s robustness and usability.

**Project Introduction (Kelvin/Sharon, 3 mins)**

Our project’s primary objective was to build an efficient, user-friendly full-stack financial portfolio management application. We aimed to provide individual users with a convenient platform to track and manage their investments, focusing primarily on stocks and bonds.

Our solution integrates a responsive web frontend and a RESTful backend service, ensuring fast and reliable communication between user interactions and the underlying database. Key functionalities include: - Real-time asset tracking, allowing users to instantly view the performance of their holdings. - Simulation of trading operations, giving users the ability to assess potential investment strategies. - Comprehensive data visualization through interactive charts, making portfolio management intuitive and insightful.

**System Flow (Richard, 4 mins)**

Now, let’s look at our detailed system flow, explaining precisely how the application operates from the user’s action to data retrieval and presentation:

1. **Frontend Interaction:** Users engage with our intuitive web interface developed using HTML, CSS, and JavaScript. This layer captures user input and provides visual feedback in real-time.
2. **API Requests:** Upon interaction, our frontend utilizes JavaScript’s Fetch API to make asynchronous HTTP requests. These requests communicate directly with our backend API endpoints, passing necessary data such as asset identifiers, transaction types, and user portfolio details.
3. **Backend Processing:** On receiving requests, our backend, powered by Node.js and Express.js, handles various tasks including request validation, business logic execution, authentication, and authorization. Each API endpoint is securely defined and tested to ensure data integrity and system security.
4. **Database Operations:** Our backend interacts with a MySQL database to perform essential CRUD (Create, Read, Update, Delete) operations. For instance, when users simulate trades or manage their portfolio, these operations ensure consistent and accurate data updates.
5. **External Data Integration:** For dynamic and accurate financial data, our backend employs Axios to fetch real-time and historical price data from the Yahoo Finance API. This integration enables users to see current market conditions instantly.
6. **Response Handling:** The backend compiles all relevant data into structured JSON responses sent back to the frontend. The frontend then dynamically renders this data, updating the user interface without page reloads.

**Screenshots Overview (Anthony, 2 mins)**

Let’s look at some key screens from our application: - **Dashboard Overview:** Here, users get immediate insights into their portfolio performance, displaying asset allocation, recent activity, and overall performance metrics. - **Stock Market Page:** This provides comprehensive real-time market data, allowing users to explore detailed asset information such as price trends and historical performance. - **Portfolio Detail View:** Users can manage individual assets, monitor gains and losses, and perform simulated trading operations directly from this view. - **Historical Chart:** An interactive tool that visually presents historical asset performance, assisting users in making informed investment decisions based on past trends and analysis.

**Live Demonstration (Team, 4 mins)**

I’ll now provide a live demonstration of our application. During this demo, we’ll explore the user experience in real-time, including: - Logging into the dashboard and exploring overall portfolio metrics. - Performing a trade simulation and observing instant feedback and database updates. - Viewing real-time market data fetched from Yahoo Finance API. - Utilizing interactive charts to analyze historical data.

[Conduct the live demonstration thoroughly, clearly explaining each action and backend process.]

**System Architecture (Kelvin, 2 mins)**

To ensure scalability and maintainability, our system is built around a robust three-tier architecture: - **Presentation Layer:** Built using HTML, CSS, and JavaScript. It efficiently manages user interactions and renders dynamic content, ensuring responsiveness across devices. - **Application Layer:** Utilizes Node.js and Express.js to handle business logic, secure API routing, and external integrations. This layer also manages authentication processes and ensures secure data flow. - **Data Layer:** Implemented with MySQL, this layer handles persistent storage of user data, portfolio transactions, and asset details. It employs transaction-based queries to ensure data integrity and accuracy.

**Technology Stack (Sharon, 2 mins)**

Here’s a deeper look into our chosen technologies: - **Backend:** We used Node.js for its non-blocking I/O and event-driven architecture, supported by Express.js for efficient request handling. MySQL2 was selected for database connectivity due to its performance and robust transaction handling. Axios facilitated seamless integration with external APIs. - **Frontend:** Built with HTML5 and CSS3 for a responsive and visually appealing design, enhanced by Vanilla JavaScript for straightforward functionality and optimal performance. We leveraged Chart.js for powerful and interactive data visualizations.

**Conclusion & Future Plans (Richard, 3 mins)**

In conclusion, we successfully delivered a fully functional, secure, and scalable portfolio management solution. Our application effectively demonstrates modern web technologies and robust backend practices.

Looking forward, our future plans include: - Implementing advanced user authentication and supporting multi-user capabilities. - Expanding asset types to encompass cryptocurrencies and ETFs, broadening investment opportunities. - Enhancing analytical capabilities, including predictive analysis and detailed reporting tools. - Upgrading the frontend with modern frameworks such as React or Vue.js, to further improve user experience and performance.

Thank you for your attention and interest in our project. We welcome any questions or feedback you might have.