Code-Layout, Readability and Reusability:

1. **Define the Structure:**
   - Start with defining the overall structure of the application, including different modules and components like employee information, travel details, approval workflow, and reporting.
2. **Create User-Friendly Layout:**
   - Design a user-friendly layout with a clear and intuitive interface. Use a modern and clean design that allows employees to easily input their travel details and submit for approval.
3. **Implement Readability Best Practices:**
   - Follow coding conventions and standards to ensure consistent formatting, naming conventions, and comments that enhance the readability of the codebase. Use descriptive names for variables, functions, and components.
4. **Utilize Modularization:**
   - Split the application into modular components to promote reusability. Create reusable components for common functionalities such as form inputs, approval status indicators, and user profiles.
5. **Use Frameworks and Libraries:**
   - Leverage popular frameworks and libraries that support the development of user interfaces and streamline the development process. Consider using frameworks like React or Angular for the front-end and Node.js or Django for the back-end.
6. **Incorporate Responsive Design:**
   - Ensure that the application is responsive and compatible with various devices and screen sizes. Use CSS media queries and responsive design techniques to optimize the user experience on different devices.
7. **Implement Approval Workflow:**
   - Develop a robust approval workflow that includes multiple levels of authorization, notifications, and tracking capabilities. Store the approval history for future reference and auditing purposes.
8. **Include Reporting Functionality:**
   - Incorporate reporting functionality to generate comprehensive reports on employee travel requests, approvals, and expenses. Use visualization libraries like D3.js or Chart.js to create visually appealing and informative reports.
9. **Ensure Security Measures:**
   - Implement security measures such as authentication, authorization, and data encryption to protect sensitive employee information and travel data from unauthorized access or breaches.
10. **Testing and Maintenance:**
    - Conduct thorough testing to identify and fix any bugs or issues. Perform regular maintenance to keep the application up-to-date and secure, and to address any compatibility issues that may arise with updates to frameworks or libraries.