Rationale

The navigation architecture presented in this assignment is grounded in the functional decomposition developed in Assignment 1 and refined through feature inventory and grouping analysis. Each Level 1 navigation category exists to represent a distinct user goal rather than a technical system boundary. Event Discovery was prioritized as the primary entry point due to its high frequency of use and its role as the starting point for most user journeys.

Bookings and Tickets were intentionally separated to distinguish between transactional management and time-sensitive access. This separation reduces cognitive load by preventing users from navigating through historical or administrative screens when they need immediate ticket access. Profile and Support were placed as secondary navigation destinations because they are accessed less frequently and typically outside core booking flows.

Screen groupings within each category were designed based on task sequence and state progression. Features that require focused, linear interaction, such as seat selection and payment, were embedded within contextual flows rather than exposed as persistent navigation items. Certain screens were merged or hidden to avoid redundancy, particularly where system-driven logic already enforces progression or validation.

Conditional and system-driven routes were explicitly integrated into the architecture to ensure predictable recovery paths. Error states, retries, authentication redirects, and session handling were treated as first-class navigation elements rather than exceptions. This approach ensures that the navigation structure remains resilient under real-world conditions such as failures, data unavailability, or partial user actions.

Overall, structural decisions were made to balance clarity, scalability, and flexibility while maintaining alignment with user goals and system constraints.

Key Insights and Learnings

This assignment demonstrated that effective navigation architecture cannot be designed independently of system logic. Many critical navigation paths are driven by conditions such as authentication state, availability, validation, and system enforcement rather than direct user intent.

The exercise highlighted the importance of separating interface-level screens from system-driven routes. While users interact with only a subset of the system, the navigation architecture must still account for all possible transitions, including errors, retries, and forced redirects. Ignoring these paths leads to fragile designs that fail under real-world usage.

Another key insight was the value of grounding navigation decisions in functional decomposition rather than assumptions about screens. By deriving navigation directly from system functionality, the architecture remains consistent, complete, and defensible.

Finally, designing navigation as a structured system rather than a collection of pages enables better scalability. As new features or conditions are introduced, they can be integrated into the existing structure without disrupting core user flows.