**Project Design Phase-II**

**Solution Requirements (Functional & Non-functional)**

|  |  |
| --- | --- |
| Date | 26-05-2025 |
| Team ID | LTVIP2025TMID43459 |
| Project Name | FlightFinder |
| Maximum Marks | 4 marks |

***Functional Requirements:***

Following are the functional requirements of the proposed solution.

Functional Requirements – FlightFinder

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR No-1 | User Authentication | Sign Up, Login, Password Reset, OAuth (Google/GitHub) |
| FR No-2 | Booking Submission | Submit booking, Upload attachments, View success confirmation |
| FR No-3 | Booking Status Tracking | View status updates, Filter by booking type/status |
| FR No-4 | Agent Assignment & Dashboard | Agents view assigned bookings, Update status |
| FR No-5 | Admin Management Panel | Assign agents, Monitor activity, Generate reports |
| FR No-6 | User-Agent Communication | |  | | --- | |  |  |  | | --- | | Real-time chat module, Complaint-threaded conversation | |
| FR No-7 | Feedback and Booking Closure | User submits feedback after resolution, Marks booking as closed |

The functional requirements of FlightFinder define the core capabilities that the system must deliver to meet user expectations. These requirements ensure that all user roles — including end users, agents, and administrators — can effectively interact with the platform.

Each functional requirement is derived from real-world user needs and structured as epics with sub-tasks or user stories. For example, the user authentication feature ensures secure access control, while booking submission and tracking enable users to register and monitor flight requests. Agents are empowered with dashboards to manage assigned tasks, and admins have tools for booking routing, analytics, and feedback analysis.

Together, these features form the foundation of a scalable, role-based platform capable of streamlining the entire booking lifecycle from submission to resolution.

***Non-functional Requirements:***

Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **NFR No** | **Non-Functional Requirement** | **Description** |
| NFR No-1 | Usability | The system should offer a responsive, clean, and intuitive UI for all users (users, agents, admins). |
| NFR No-2 | Security | Implement HTTPS encryption, token-based authentication, and role-based access controls. |
| NFR No-3 | Performance | All pages should load in under 2 seconds. Booking submission and communication should be near-instant. |
| NFR No-4 | Availability | System should offer at least **99.9% uptime**, with auto-restart on crash using process managers. |
| NFR No-5 | Scalability | Architecture should support multiple departments/organizations and high concurrency using scalable backend (e.g., microservices or load balancing). |
| NFR No-6 | Maintainability | Codebase should follow modular design with separate services for frontend, backend, and database for easier updates. |
| NFR No-7 | Compatibility | The platform should be compatible with all major modern browsers and mobile devices. |

The non-functional requirements define the quality attributes of the FlightFinder platform — focusing on performance, security, scalability, availability, and usability.

To ensure a seamless user experience, the application prioritizes fast page load times, responsive interfaces, and minimal delays in messaging or booking actions. Security is enforced through HTTPS encryption, token-based authentication (JWT), and strict role-based access control.

The system is designed to support **99.9% uptime**, with room for horizontal scaling to accommodate growing user bases. It is also engineered to be maintainable and compatible across devices and browsers, making it future-ready and adaptable to evolving requirements.

These NFRs ensure that FlightFinder is not only functional but also reliable, secure, and user-friendly in real-world environments.