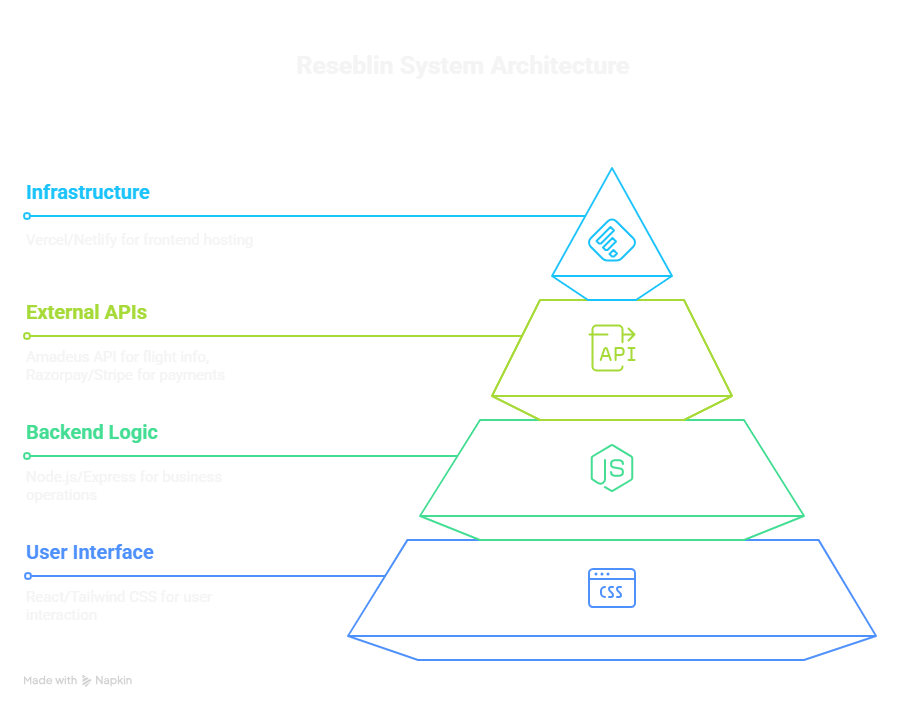
# Project Design Phase-II Technology Stack (Architecture & Stack)

Date: 24 June 2025  
Team ID: LTVIP2025TMID57557  
Project Name: Flight Booking System  
Maximum Marks: 4 Marks

## Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

Diagram: 

## Table-1: Components & Technologies:

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Component | Description | Technology |
| 1 | User Interface | How user interacts with application | HTML, Tailwind CSS, React JS, Vite |
| 2 | Application Logic-1 | Frontend Booking Logic | React + TypeScript |
| 3 | Application Logic-2 | API Integration Layer | Fetch/Axios in React |
| 4 | Application Logic-3 | Server-side Business Logic | Node.js, Express (assumed) |
| 5 | Database | Stores user and booking data (not implemented) | To be added: MongoDB or PostgreSQL |
| 6 | Cloud Database | Cloud hosted DB service (optional) | Firebase / AWS RDS (suggested) |
| 7 | File Storage | For file uploads (if any) | Local Filesystem / Cloud Storage (suggested) |
| 8 | External API-1 | Flight Information & Booking APIs | Example: Amadeus API |
| 9 | External API-2 | Payment Gateway | Example: Razorpay / Stripe (not integrated) |
| 10 | Machine Learning Model | N/A for current project | None |
| 11 | Infrastructure (Server / Cloud) | Application deployment & hosting | Vercel / Netlify for Frontend, Localhost for backend (assumed) |

## Table-2: Application Characteristics:

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Characteristics | Description | Technology |
| 1 | Open-Source Frameworks | Frameworks and libraries used | React, Tailwind CSS, Vite, Node.js |
| 2 | Security Implementations | Basic form validation and CORS handling | HTTPS, CORS Middleware (assumed) |
| 3 | Scalable Architecture | Frontend and backend modularity | Client-Server model, scalable if containerized |
| 4 | Availability | Static hosting and server reliability | Vercel/Netlify for high uptime |
| 5 | Performance | Optimized frontend with Vite and Tailwind | Vite bundling, lazy loading |

## References:

https://c4model.com/

https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/

https://www.ibm.com/cloud/architecture

https://aws.amazon.com/architecture

https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d