**Stay Elite – Approach & Implementation**

Assessment Submission  
Author: Nitesh Goel

## 1. Introduction & Understanding

Stay Elite aims to provide CXOs and HNIs with a personalized, secure, and efficient travel management platform. The system focuses on reducing manual effort, remembering user preferences, centralizing loyalty information, and visualizing travel history in a cost-effective, intuitive manner.

## 2. Core Problem & Real-Life Relevance

High-value travelers often face fragmented travel data, making bookings repetitive and error-prone. Current tools do not adapt to their preferences or provide actionable insights. Stay Elite addresses these gaps, offering an intelligent, privacy-focused solution that enhances user experience while keeping operational costs low.

## 3. Methodology & Approach

My approach breaks the project into four key pillars:   
1. Personalized preference management with AI-assisted suggestions.  
2. Centralized repository for loyalty memberships and benefits.  
3. Static map visualization of travel history for clarity and cost-efficiency.  
4. Smart rebooking and recommendation system for repetitive trips.  
  
The focus is on building a modular, scalable system where each component can evolve independently while providing immediate value to users.

## 4. Technical Stack & Decisions

• Frontend: React.js – dynamic dashboard, preference toggles, map integration.  
• Backend: FastAPI (Python) – lightweight APIs, ML-ready for preference engine.  
• Database: PostgreSQL for structured travel data, Redis for caching.  
• Storage: AWS S3 – documents and static maps.  
• Map Rendering: Plotly/Matplotlib server-side to generate static SVG/PNG maps, avoiding recurring API costs.  
• Notifications: SendGrid/Twilio – booking alerts, membership expiries.  
• Security: OAuth2 + JWT, RBAC, TLS, AES-256 encryption, audit logs.  
  
These choices balance \*\*practicality, scalability, and security\*\*, keeping the solution realistic for an early implementation while demonstrating technical insight.

## 5. UX / Workflow Thought Process

The system is designed for minimal cognitive load:  
1. Executive assistants submit travel requests with optional auto-filled preferences.  
2. Concierges review and approve bookings, adjusting only when necessary.  
3. Users receive static map visualization and actionable recommendations like 'Rebook Similar Trip'.  
4. Membership alerts and document reminders ensure smooth operations.  
  
This workflow ensures speed, accuracy, and consistency for elite travelers.

## 6. Innovation / Preference Learning

Preference learning uses real signals such as preferred airlines, hotels, travel times, and seat classes. Initially, a rule-based system handles common patterns (80% coverage), while lightweight ML models suggest less obvious preferences. Confidence thresholds prioritize precision and provide human-overridable recommendations.

## 7. Static Map Strategy

Static maps aggregate trip locations and render pins with counts and latest visit labels. Images are generated server-side (Plotly/Matplotlib) and cached on S3, providing quick access without API costs. Filters by year and trip type allow clear insights, making the map both practical and visually appealing for executive dashboards.

## 8. Success Metrics / Real Outcomes

• 80% of new trips auto-fill with accurate preferences.  
• 30% of bookings utilize the 'Rebook Similar Trip' feature.  
• 75% of users engage with the static map monthly.  
• Manual corrections for preferences reduced by 50%.  
• High user satisfaction (target >90%) achieved through efficiency and personalization.

## 9. Timeline / MVP Plan (8 Weeks)

Weeks 1–2: Set up user profiles, preferences, trip request models, and database schemas.  
Weeks 3–4: Implement trip workflow, membership repository, document uploads.  
Weeks 5–6: Develop static map backend, SVG/PNG generation, caching, and dashboard embedding.  
Weeks 7–8: Implement preference learning (rules + ML), 'Rebook Similar Trip', notifications, and basic admin panel.  
This phased approach ensures a working MVP that delivers visible value early, allowing iterative improvement.

## 10. Reflection & Conclusion

Stay Elite combines \*\*innovation, practical engineering, and user-focused design\*\*. My approach demonstrates the ability to think end-to-end: understanding real user pain points, designing low-cost, privacy-aware solutions, and implementing a modular, scalable system. Completing this project would not only create a useful product but also showcase a student's readiness to transition into a professional engineering role.