**1) Recommended tech stack (mapped to requirements)**

**Core AWS (must-haves)**

* **Amazon Bedrock** — LLM hosting & agent orchestration (LLM reasoning + RAG).
  + Use Bedrock models (Anthropic/Claude, Amazon Titan, Nova) for planning, itinerary generation, and multi-turn reasoning. [AWS Documentation](https://docs.aws.amazon.com/bedrock/latest/userguide/getting-started.html?utm_source=chatgpt.com)
* **Amazon Bedrock AgentCore** — use primitives:
  + **Gateway** — wrap booking/price APIs and internal tools as agentic tools.
  + **Memory** — user preferences, past trips, conversation memory (persisted to DynamoDB/S3).
  + **Observability** — trace agent decisions for demo and audit.
  + **Identity** — optional for login-based personalization (Cognito).
* **Amazon SageMaker** (optional) — if you want to fine-tune or host custom models or evaluation pipelines.

**Data, storage & search**

* **Amazon S3** — store user uploads (passports (mock), itinerary PDFs), demo assets.
* **Amazon DynamoDB** — user profiles, preferences, trip states, booking confirmations.
* **Amazon OpenSearch Serverless** — semantic index for activity suggestions, cached destination docs, and similarity search.

**Compute & orchestration**

* **AWS Lambda** (serverless functions) — implement booking workflows, orchestration endpoints for the agent, and handle webhooks from third-party APIs.
* **Amazon API Gateway** — expose backend endpoints for your frontend and agent gateway tools.
* **Amazon EventBridge / Step Functions** — orchestrate multi-step booking flows (e.g., search → hold → confirm → notify).

**Auth & monitoring**

* **Amazon Cognito** — user sign-in and basic Identity primitive integration.
* **Amazon CloudWatch + AgentCore Observability** — logs, metrics, tracing for demo and debugging.

**Frontend & demo**

* **React + Vite** or **Next.js** (client) — interactive UI to show itinerary builder, chat widget, booking flow. (Muhammad suggested React earlier — good choice.)
* **Simple CLI demo** alternative — Node.js script that runs an autonomous planning flow for judges who prefer quick playback.

**Third-party (booking + activities) — use official APIs only (no scraping)**

Important: avoid scraping. Use official partner APIs (or demo with mocked data / sandbox keys) to avoid ToS/legal issues.

* **Flights & hotels**: **Amadeus for Developers** (flight/hotel/search, pricing), **Skyscanner / RapidAPI** (flight search), **Booking.com/HotelsCombined** affiliate APIs (hotel availability). These require API keys and may have commercial/onboarding restrictions — plan early.
* **Activities / local experiences**: **Viator** (TripAdvisor/Viator APIs), **GetYourGuide**, or **Google Places API** (for POI & activities) — but check quotas and license for reuse.
* **Payments / bookings** (demo): use sandbox modes or mock booking confirmations. DO NOT attempt real charges unless you have a verified sandbox/test merchant account.

**Dev / infra tooling**

* **Terraform / AWS CDK** — infra as code (optional, great for reproducible demo).
* **GitHub** repo (public) — required by hackathon. Include README with infra/deploy instructions and mock-data switching flags.