# Draft Fare Ferry Processing Specification

## Versioning

| **Author** | **Date** | **Details** |
| --- | --- | --- |
| Andrew Grealy | October, 18th, 2023 | Original |

## Introduction

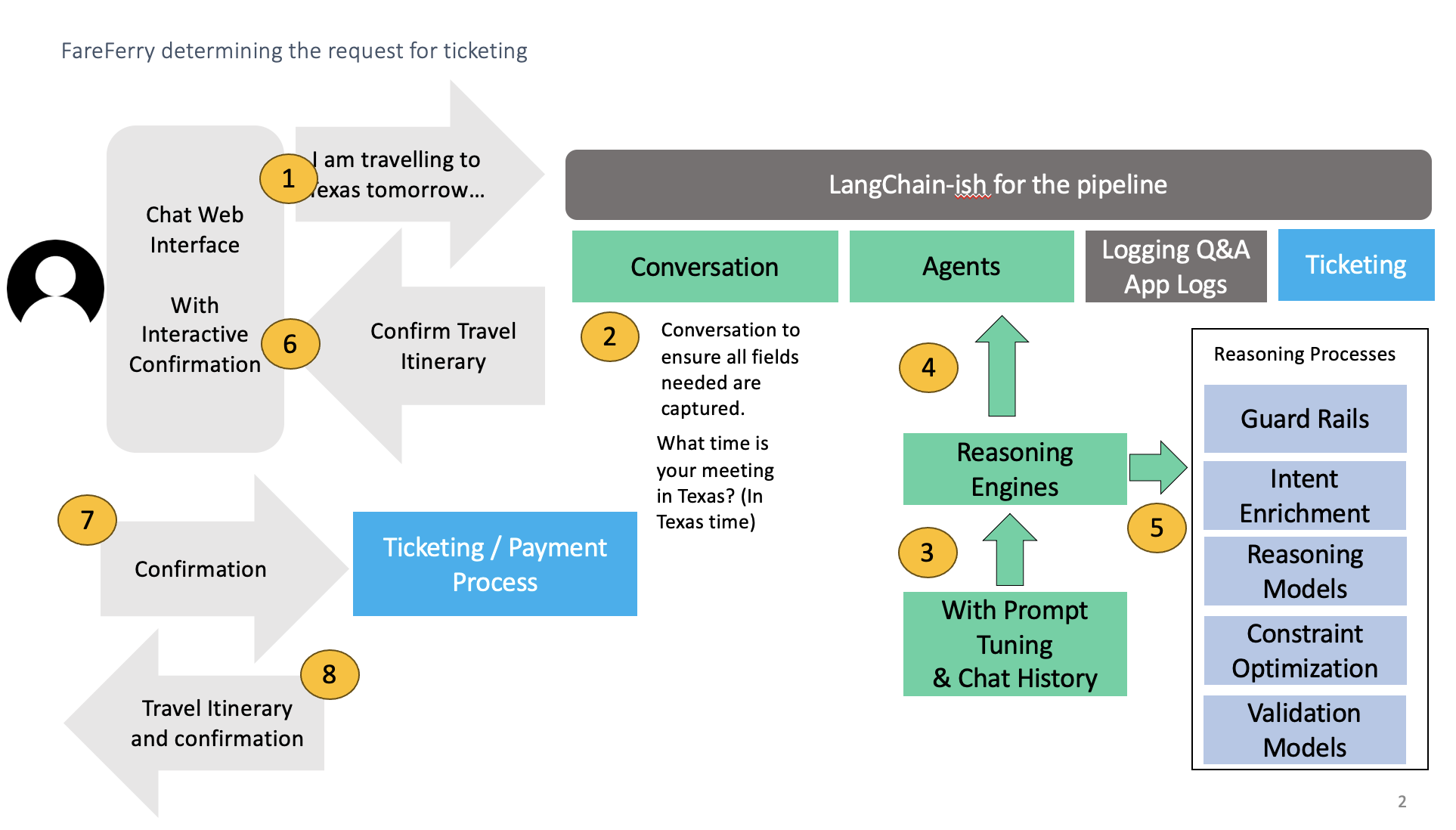
Fare Ferry is a new one to interact with, creating business travel using the latest in conversation chat, LLMs, and reasoning engines.

The idea is to take the user’s preferred preferences and the prompt for travel and convert this to a travel request that can be sent to Amadeus to fulfill the request. If all the information is not provided, then a conversation is started to ensure the needed data is collected.

The UI and chat interaction is what will set this product apart from previous travel companies. The rise of ChatGPT to the quickest 100 million users has made this chat conversational interface allow natural language for querying. Nearly everyone can quickly learn how to use this interface with virtually no experience.

Not everything can be determined easily from Chat, so we have a 3-column layout - menu, chat, then interactive UI of the generation of the itinerary.

## Simple Process Flow



1. **User Request**: A user indicates their travel plans, likely through a chat interface. An example statement is given: "I am traveling to Texas tomorrow..."

2. **System Components**:

**Chat Web Interface**: The system likely has a chatbot or live chat feature where the user initiates their request.

**Logging**: Activities within the chat are logged for monitoring or analytics.

Q&A Set: The system has a set of predefined questions and answers used for interaction.

**App Logs**: Application logs are maintained for technical monitoring.

3. **Interactive Confirmation**: The system engages with the user to confirm the travel itinerary. It ensures all required fields or information are captured.

4. Conversation Reasoning Processes:

The system uses reasoning processes to understand the user's needs better. It might ask follow-up questions like, "What time is your meeting in Texas?"

**Guard Rails**: in place to prevent errors or guide the user through the process.

**Intent Enrichment**: The system seems to enrich or interpret the user's intent through additional data or context

**Reasoning Models**: Utilizes underlying models to process and reason about the user's requests and data.

**Constraint Optimization**: It might involve finding the best solutions or outcomes within certain constraints (e.g., time, cost).

5. **Travel Itinerary Validation and Confirmation**

7. **Ticketing and payment**

After the interactive process, the system validates the travel itinerary. It ensures that all details are accurate and confirms the travel arrangements with the user. Then, calls perform ticketing and payments.

—

Neurosymbolic makes confidence judgements and can expire and lower values. \*\*\* Put some of that data in there \*\*\*

## Technologies Used

| **Technology** | **Reason** |
| --- | --- |
| LLMs | Used for natural language to capture fields needed |
| React | Front-end UI |
| Python Flask/Fast API | For backend |
| Vector Store | For common embeddings to problems |
| Database | For historical data such as payments |
| NVidia Guard Rails | Ensure someone isn’t hacking our LLM processing |
| Neurosymbolic AI | Rule processing, validation, confirmation of correct data passed, etc. |
| Okta | For corporate authentication |
| WAF | For protection against attacks |
| Hugging Face Models | For solving various problems |
| Langchain | Or something similar |
|  |  |

The UI to be created in React and the back-end in Python (Fast API or Flask). React developers are easy to hire and find and used in a lot of mission-critical applications. The Python backend makes it easy to integrate easily with AI. Over time, it would make sense to move this to Mojo for performance, speed, and reliability.

## Personalization

This is a stored JSON of the user’s personalization. (It could be serialized into columns in a database or a straight JSON, which is to be determined.)

{

"userProfile": {

"userId": "uniqueUserId",

"preferences": {

"flightPreferences": {

"airlines": ["One World Partners", "American Airlines", "United", "Lufthansa"],

"class": ["business", "first"],

"seatPreferences": ["aisle", "window"],

"layoverPreferences": {

"maxLayoverDuration": 120,

"preferredAirports": ["JFK", "LAX", "ORD"]

},

"airlineMemberships": {

"AmericanAirlines": {

"membershipLevel": "Platinum",

"frequentFlyerNumber": "12345"

},

"OneWorld": {

"membershipLevel": "Emerald",

"frequentFlyerNumber": "67890"

},

"United": {

"membershipLevel": "Gold",

"frequentFlyerNumber": "11223"

}

}

},

"hotelPreferences": {

"starRating": [3, 4],

"proximity": ["airport", "downtown"],

"amenities": ["pool", "gym", "freeWifi"],

"type": ["petFriendly", "adultsOnly"],

"roomType": ["single", "double"],

"chain": ["Hilton", "Marriott"],

"extendableStay": true,

"location": ["market", "conferenceVenue"],

"hotelMemberships": {

"HiltonHonors": {

"membershipLevel": "Diamond",

"membershipNumber": "H12345"

},

"MarriottBonvoy": {

"membershipLevel": "Platinum Elite",

"membershipNumber": "M54321"

}

}

},

"carRentalPreferences": {

"carType": ["SUV", "Sedan"],

"fuelType": ["gasoline", "electric"],

"rentalCompany": ["Hertz", "Enterprise"],

"pickup": ["airport", "hotel"],

"features": ["GPS", "childSeat"],

"carRentalMemberships": {

"HertzGold": {

"membershipLevel": "President’s Circle",

"membershipNumber": "HG12345"

},

"EnterprisePlus": {

"membershipLevel": "Platinum",

"membershipNumber": "EP54321"

}

}

},

"passportDetails": {

"passportNumber": "123456789",

"issuingCountry": "USA",

"expiryDate": "2030-01-01",

"nationality": "American",

"birthDate": "1990-01-01",

"issueDate": "2020-01-01"

},

"executiveAssistant": {

"name": "Jane Doe",

"phone": "+9876543210",

"email": "janedoe@example.com"

},

"conciergeContact": {

"name": "John Smith",

"phone": "+1234567890",

"email": "johnsmith@example.com"

},

"specialConstraints": {

"budgetLimit": 4500,

"minimizeJetLag": true,

"arrivalDeadline": "2023-10-15T10:00:00Z",

"departureDeadline": "2023-10-16T16:00:00Z",

"returnDeadline": "2023-10-16T22:00:00Z",

"groupBooking": {

"sameFlight": true,

"seatsTogether": true

},

"lastMinuteBooking": true,

"nextAvailableFlight": true,

"emergencyContact": {

"name": "John Doe",

"phone": "+1234567890",

"email": "johndoe@example.com"

}

},

"bookingInfo": {

"currentBookings": [{

"bookingId": "ABCXYZ",

"flightDetails": {

"departure": "2023-10-15",

"return": "2023-10-16",

"from": "New York",

"to": "Chicago"

},

"hotelDetails": {

"checkIn": "2023-10-15",

"checkOut": "2023-10-16",

"hotelName": "Hilton Chicago",

"roomType": "Double"

},

"carRentalDetails": {

"pickup": "2023-10-15",

"dropoff": "2023-10-16",

"carType": "SUV",

"rentalCompany": "Hertz"

}

}],

"pastBookings": [],

"futureBookings": []

}

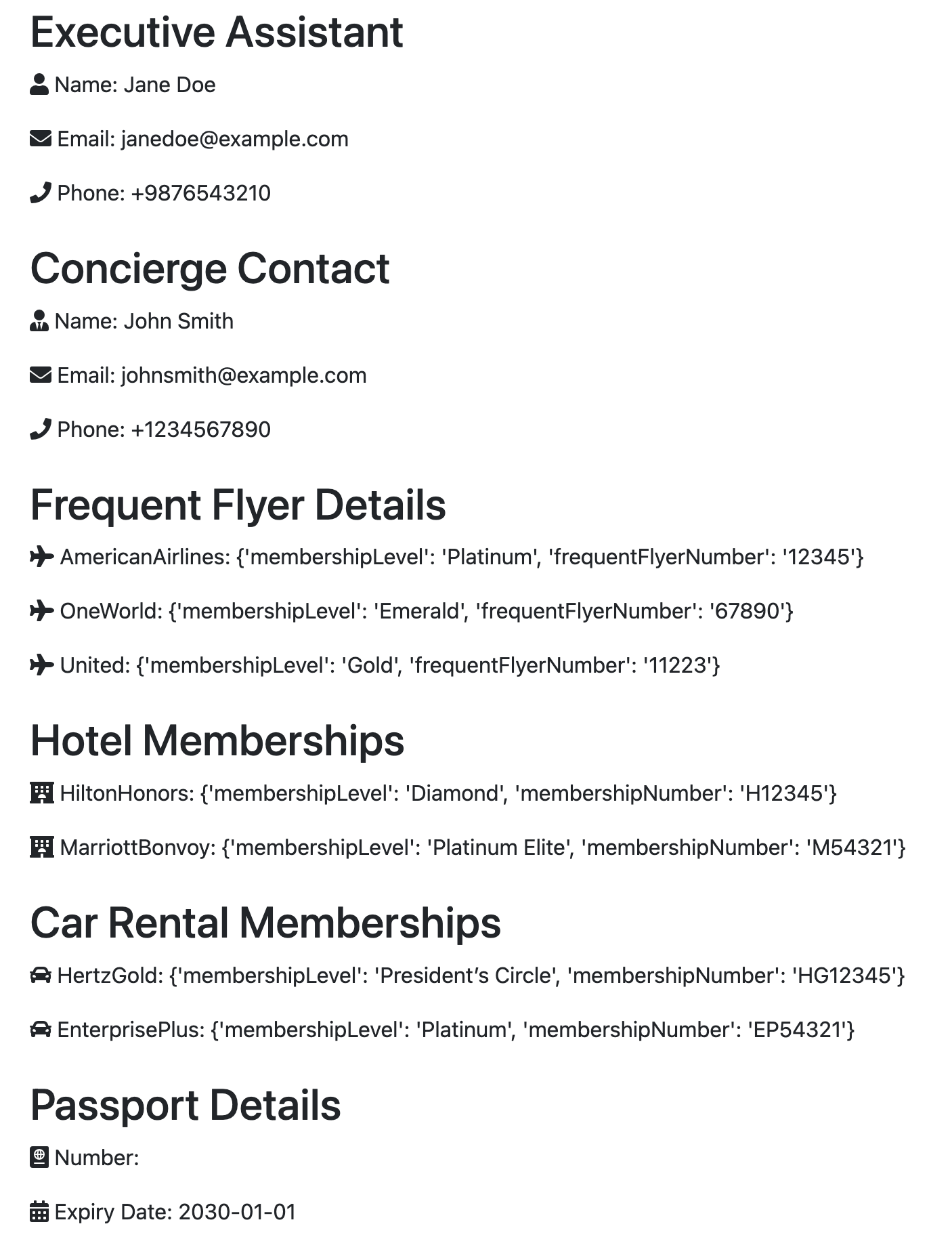
}

}

}

Above is not inclusive of all the things we need, it’s the first pass.

The user/admin assistant would be able to edit this profile. It would be displayed in an easy-to-edit, inline editing-allowed format. Here, it only shows the JSON at the top level as items; the real interface will make it all look pretty and very useable.



## Standing on the shoulders of Giants

Where possible, we use technologies that scale and are proven; the same goes for profiles and such.

In this space, a major competitor is Concur; their API and entity formats are well-defined and the license is Apache 2.0 license, so instead of having to work out everything we should store, we should use their entity models as a guide and improve.

<https://github.com/SAP-docs/preview.developer.concur.com>

<https://github.com/concur/developer.concur.com/blob/preview/src/api-reference/travel/itinerary-v4/v4.itinerary.md>

For example, the booking details of a trip are returned in JSON. We could even learn all the rules of Concur from the documentation and see if that is valid.

## JSON Example

{

"BookedVia": "Agent",

"Bookings": [

{

"AgencyName": "Outtask Travel Apollo",

"AgencyPCC": "C4I",

"AirfareQuotes": [

{

"BaseFare": 1264,

"BaseFareCurrency": "EUR",

"DateCreatedUtc": "2020-11-04T20:54:19.000+00:00",

"DateModifiedUtc": "2020-11-04T20:54:19.000+00:00",

"IssueByDate": "2020-09-25T15:54:18.000-00:00",

"TotalFare": 1341.2,

"TotalFareCurrency": "EUR"

}

],

"AirlineTickets": {

"AirlineAdjustment": [

{

"AddCollectAmount": 0,

"AdjustmentDateTime": "2020-11-02T00:00:00.000-00:00",

"AdjustmentType": "C",

"AirlineCharges": {

"Fixed": [

{

"Amount": -23,

"Currency": "EUR",

"Description": "Ice cream with meal",

"IsPaid": true,

"IsPrimary": false

}

]

},

"DateCreatedUtc": "2020-11-04T20:54:19.000+00:00",

"DateModifiedUtc": "2020-11-04T20:54:19.000+00:00",

"PassengerName": "EschIII/BenDwayne",

"PlatingCarrierNumericCode": "638",

"PlatingControlNumber": "0522482769",

"RecordLocator": "A56053",

"Taxes": [

{

"TaxAmount": 5,

"TaxAuthority": "USA",

"TaxName": "Silly Air Adjustment Tax",

"TaxRate": 0.58,

"TaxType": "US"

},

{

"TaxAmount": -5,

"TaxAuthority": "USA",

"TaxName": "Silly Air Adjustment Tax",

"TaxRate": 0.58,

"TaxType": "AY"

},

{

"TaxAmount": -9,

"TaxAuthority": "USA",

"TaxName": "Silly Air Adjustment Tax",

"TaxRate": 0.58,

"TaxType": "XF"

}

],

"TotalAdjustment": -100,

"TotalAdjustmentCurrency": "USD"

}

],

"AirlineTicket": [

{

"AccountingLine": {

"AirlineCode": "AA",

"Comment": "VIxxxxxxxxxxxx1111",

"Commission": "10.00",

"Fare": "1168",

"FOPMethod": "CX",

"MCOType": "AC",

"Tax": "72.40",

"TranControlNbr": "0251207588"

},

"AirlineTicketCoupons": [

{

"ClassOfService": "E",

"CouponNumber": 1,

"CouponStatus": "USED",

"EndCityCode": "DEN",

"FlightNumber": "5894",

"RateCode": "CQ690",

"StartCityCode": "IAD",

"StartDateLocal": "2020-12-11T15:54:18.000-00:00",

"Vendor": "AA"

},

{

"ClassOfService": "E",

"CouponNumber": 2,

"CouponStatus": "OPEN",

"EndCityCode": "LAX",

"FlightNumber": "6617",

"RateCode": "CQ210",

"StartCityCode": "IAD",

"StartDateLocal": "2020-12-16T15:54:18.000-00:00",

"Vendor": "UA"

}

],

"AirlineTicketFareBreakups": [

{

"BaseFare": 275.55,

"Currency": "USD",

"EndCityCode": "SFO",

"IsRefundable": true,

"StartCityCode": "LAX",

"TotalFare": 355.55,

"Vendor": "AA"

}

],

"BaseFare": 1168,

"BaseFareCurrency": "USD",

"ComparisonFare": 1,

"ComparisonFareCurrency": "USD",

"DateCreatedUtc": "2020-11-04T20:54:19.000+00:00",

"DateModifiedUtc": "2020-11-04T20:54:19.000+00:00",

"Endorsements": "NOREF/NOEXCH. NO VALUE AFTER FIRST FLT DATE",

"IssueDateTime": "2020-11-03T00:00:00.000-00:00",

"IssuingIataAgencyNumber": 82494313,

"IssuingPseudoCity": "13H1",

"LinearFareConstructor": "THIS IS A LINEAR FARE CONSTRUCTOR IAD-LAX",

"PassengerName": "DoverRN MSN CNBC/Emil",

"PlatingCarrierNumericCode": "001",

"PlatingControlNumber": "0251207588",

"ProgramCarrierCode": "AA",

"ProgramMembershipNumber": "387519635",

"RecordLocator": "A88372",

"Taxes": [

{

"TaxAmount": 58.4,

"TaxAuthority": "USA",

"TaxName": "Silly Air Ticket Tax",

"TaxRate": 0.57,

"TaxType": "US"

},

{

"TaxAmount": 5,

"TaxAuthority": "USA",

"TaxName": "SillyAir Ticket Tax",

"TaxRate": 0.57,

"TaxType": "AY"

},

{

"TaxAmount": 9,

"TaxAuthority": "USA",

"TaxName": "Silly Air Ticket Tax",

"TaxRate": 0.57,

"TaxType": "XF"

},

{

"TaxAmount": 3.6,

"TaxAuthority": "USA",

"TaxName": "Silly Air Ticket Tax",

"TaxRate": 0.57,

"TaxType": "ZP"

}

],

"Ticketless": false,

"TotalFare": 1244,

"TotalFareCurrency": "USD"

},

{

"AccountingLine": {

"AirlineCode": "AA",

"Comment": "VIxxxxxxxxxxxx1111",

"Commission": "10.00",

"Fare": "2341",

"FOPMethod": "CX",

"MCOType": "AC",

"Tax": "131.05",

"TranControlNbr": "0137973356"

},

"AirlineTicketCoupons": [

{

"ClassOfService": "E",

"CouponNumber": 1,

"CouponStatus": "EXCH",

"EndCityCode": "DEN",

"FlightNumber": "5894",

"RateCode": "CQ690",

"StartCityCode": "IAD",

"StartDateLocal": "2020-12-11T15:54:18.000-00:00",

"Vendor": "AA"

},

{

"ClassOfService": "E",

"CouponNumber": 2,

"CouponStatus": "EXCH",

"EndCityCode": "LAX",

"FlightNumber": "6617",

"RateCode": "CQ210",

"StartCityCode": "IAD",

"StartDateLocal": "2020-12-16T15:54:18.000-00:00",

"Vendor": "UA"

}

],

"AirlineTicketExchanges": [

{

"Amount": 100,

"AppliedSegment1": 1,

"AppliedSegment2": 1,

"Currency": "EUR",

"DateModifiedUtc": "2020-11-04T20:54:19.000+00:00",

"OldRecordLocator": "DEL324",

"PlatingCarrierNumericCode": "001",

"PlatingControlNumber": "0137973356"

}

],

"AirlineTicketFareBreakups": [

{

"BaseFare": 255.55,

"Currency": "USD",

"EndCityCode": "SFO",

"IsRefundable": true,

"StartCityCode": "JFK",

"TotalFare": 375.55,

"Vendor": "AA"

}

],

"BaseFare": 2341,

"BaseFareCurrency": "USD",

"ComparisonFare": 1,

"ComparisonFareCurrency": "USD",

"DateCreatedUtc": "2020-11-04T20:54:19.000+00:00",

"DateModifiedUtc": "2020-11-04T20:54:19.000+00:00",

"Endorsements": "NOREF/NOEXCH. NO VALUE AFTER FIRST FLT DATE",

"IssueDateTime": "2020-11-02T00:00:00.000-00:00",

"IssuingIataAgencyNumber": 96085218,

"IssuingPseudoCity": "13H1",

"LinearFareConstructor": "THIS IS A LINEAR FARE CONSTRUCTOR IAD-LAX",

"PassengerName": "DoverRN MSN CNBC/Emil",

"PlatingCarrierNumericCode": "001",

"PlatingControlNumber": "0137973356",

"ProgramCarrierCode": "AA",

"ProgramMembershipNumber": "387519635",

"RecordLocator": "A21012",

"Taxes": [

{

"TaxAmount": 117.05,

"TaxAuthority": "USA",

"TaxName": "Silly Air Ticket Tax",

"TaxRate": 0.57,

"TaxType": "US"

},

{

"TaxAmount": 5,

"TaxAuthority": "USA",

"TaxName": "SillyAir Ticket Tax",

"TaxRate": 0.57,

"TaxType": "AY"

},

{

"TaxAmount": 9,

"TaxAuthority": "USA",

"TaxName": "Silly Air Ticket Tax",

"TaxRate": 0.57,

"TaxType": "XF"

},

{

"TaxAmount": 3.6,

"TaxAuthority": "USA",

"TaxName": "Silly Air Ticket Tax",

"TaxRate": 0.57,

"TaxType": "ZP"

}

],

"Ticketless": false,

"TotalFare": 2475.65,

"TotalFareCurrency": "USD"

}

]

},

"BookingOwner": "ConcurTravel",

"BookingSource": "Manual",

"Charges": {

"Fixed": [

{

"Amount": 2,

"Currency": "USD",

"Description": "Booking fee",

"IsPrimary": false,

"SemanticsCode": "OTHER",

"SemanticsVendorType": "C",

"Vendor": "LH",

"VendorChargeCode": "BF2000"

}

],

"Percent": [

{

"Amount": 7,

"Currency": "USD",

"Description": "Tax",

"IsPrimary": false,

"SemanticsCode": "VAT",

"SemanticsVendorType": "H",

"VendorChargeCode": "Mars tax"

}

]

},

"DateBookedLocal": "2020-10-30T15:54:18.000-00:00",

"DateCreatedUtc": "2020-11-04T20:54:19.000+00:00",

"DateModifiedUtc": "2020-11-04T20:54:19.000+00:00",

"Delivery": {

"AddressLine1": "200 Street 1",

"AddressLine2": "Street 2",

"City": "London",

"Country": "UK",

"Email": "pasenger@keto.com",

"Latitude": 51.320000,

"LocationAdditionalDetails": "<Kiosk KioskLocation=\"On concourse\"/>",

"LocationDesc": "You will find the Self-service Ticket machine located at the front of the station. Aberystwyth Station is open 24 hours a day.",

"LocationName": "London Euston",

"Longitude": 0.500000,

"PhoneNumber": "(703)837.6100",

"ReferenceNumber": "RBK9G589",

"State": "MA",

"Type": "Kiosk",

"Zip": "32432"

},

"FormOfPaymentName": "CorporateAccount",

"FormOfPaymentType": "CA",

"IsGhostCard": false,

"PassPrograms": [

{

"Amount": 2,

"Name": "North America - Tango Plus 200 credits",

"Type": "Credits",

"UserFirstName": "Peter",

"UserLastName": "Neagle"

}

],

"Passengers": [

{

"FirstNameNumber": 0,

"FrequentTravelerPrograms": {

"FrequentFlyer": [

{

"FrequentFlyerNumber": "1234567890"

},

{

"AirlineVendor": "BA",

"FrequentFlyerNumber": "01234567890"

}

]

},

"LastNameNumber": 1,

"NameFirst": "Emil",

"NameLast": "Dover",

"NameSuffix": "RN MSN CNBC",

"NameTitle": "Mr.",

"TextName": "DoverRN MSN CNBC/Emil"

},

{

"FirstNameNumber": 1,

"FrequentTravelerPrograms": {

"FrequentFlyer": [

{

"FrequentFlyerNumber": "1234567890"

},

{

"AirlineVendor": "BA",

"FrequentFlyerNumber": "01234567890"

},

{

"FrequentFlyerNumber": "ABC12345",

"Status": "Gold",

"StatusExpirationDate": "2020-12-31T00:00:00.000-00:00"

}

]

},

"LastNameNumber": 1,

"NameFirst": "Ben",

"NameLast": "Esch",

"NameMiddle": "Dwayne",

"NameRemark": "FINANCE",

"NameSuffix": "III",

"NameTitle": "Mr.",

"TextName": "EschIII/BenDwayne"

}

],

"PhoneNumbers": [

{

"Description": "Residence",

"PassengerRPH": 0,

"PhoneNumber": "703-837-6100"

}

],

"RecordLocator": "BCC52120201104205418831",

"Remarks": {

"TripLinkRemarks": [

{

"TripLinkRemark": [

{

"Text": "TESTING"

}

]

}

]

},

"Segments": {

"Air": [

{

"AircraftCode": "767",

"Cabin": "E",

"CarbonEmissionLbs": 5470,

"CarbonModel": 158,

"CheckedBaggage": "Extra bag $25",

"ClassOfService": "E",

"ConfirmationNumber": "N1985820201104205418865",

"DateCreatedUtc": "2020-11-04T20:54:19.000+00:00",

"DateModifiedUtc": "2020-11-04T20:54:19.000+00:00",

"Duration": 140,

"EndCityCode": "DEN",

"EndDateLocal": "2020-12-11T18:14:18.000-00:00",

"EndDateUtc": "2020-12-12T01:14:18.000+00:00",

"EndGate": "68",

"EndTerminal": "D",

"FlightNumber": "5894",

"FrequentTravelerId": "387519635",

"IsUpgradeAllowed": true,

"LegId": 1,

"Meals": "Kebabs",

"Miles": 986,

"NumStops": 0,

"OperatedByFlightNumber": "8606",

"OperatedByVendor": "DL",

"OperatedByVendorName": "Delta",

"Seats": [

{

"PassengerRph": 0,

"SeatNumber": "12A",

"Status": "X"

},

{

"PassengerRph": 1,

"SeatNumber": "13B"

}

],

"SpecialInstructions": "Nothing special",

"StartCityCode": "IAD",

"StartDateLocal": "2020-12-11T15:54:18.000-00:00",

"StartDateUtc": "2020-12-11T20:54:18.000+00:00",

"StartGate": "47",

"StartTerminal": "A",

"Status": "HK",

"UpgradedDateTime": "2020-12-11T15:54:18.000-00:00",

"Vendor": "AA",

"VendorName": "American Airlines"

},

{

"AircraftCode": "767",

"Cabin": "E",

"CheckedBaggage": "Extra bag $25",

"ClassOfService": "E",

"ConfirmationNumber": "N7192620201104205418868",

"DateCreatedUtc": "2020-11-04T20:54:19.000+00:00",

"DateModifiedUtc": "2020-11-04T20:54:19.000+00:00",

"Duration": 81,

"EndCityCode": "LAX",

"EndDateLocal": "2020-12-16T17:15:18.000-00:00",

"EndDateUtc": "2020-12-17T01:15:18.000+00:00",

"EndGate": "20",

"EndTerminal": "D",

"FlightNumber": "6617",

"FrequentTravelerId": "1815857656",

"IsUpgradeAllowed": true,

"LegId": 2,

"Meals": "Kebabs",

"Miles": 663,

"NumStops": 2,

"OperatedByFlightNumber": "4047",

"OperatedByVendor": "UA",

"OperatedByVendorName": "United",

"Seats": [

{

"PassengerRph": 0,

"SeatNumber": "12A",

"Status": "X"

},

{

"PassengerRph": 1,

"SeatNumber": "13B"

}

],

"SpecialInstructions": "Nothing special",

"StartCityCode": "IAD",

"StartDateLocal": "2020-12-16T15:54:18.000-00:00",

"StartDateUtc": "2020-12-16T20:54:18.000+00:00",

"StartGate": "86",

"StartTerminal": "A",

"Status": "HK",

"UpgradedDateTime": "2020-12-11T15:54:18.000-00:00",

"Vendor": "UA",

"VendorName": "United"

}

]

},

"WebAddresses": [

{

"Description": "Work Email",

"Format": "E",

"PassengerRPH": 0,

"Type": "WRK",

"WebAddress": "Michaell@concur.com"

}

]

},

{

"AgencyName": "Outtask Travel Apollo",

"AgencyPCC": "C4I",

"BookingOwner": "ConcurTravel",

"BookingSource": "Manual",

"Charges": {

"Fixed": [

{

"Amount": 4,

"Currency": "USD",

"Description": "Booking fee",

"IsPrimary": false,

"SemanticsCode": "OTHER",

"SemanticsVendorType": "C",

"Vendor": "LH",

"VendorChargeCode": "BF2000"

}

],

"Percent": [

{

"Amount": 1,

"Currency": "USD",

"Description": "Tax",

"IsPrimary": false,

"SemanticsCode": "VAT",

"SemanticsVendorType": "H",

"VendorChargeCode": "Mars tax"

}

]

},

"DateBookedLocal": "2020-10-30T15:54:18.000-00:00",

"DateCreatedUtc": "2020-11-04T20:54:19.000+00:00",

"DateModifiedUtc": "2020-11-04T20:54:19.000+00:00",

"Delivery": {

"AddressLine1": "200 Street 1",

"AddressLine2": "Street 2",

"City": "London",

"Country": "UK",

"Email": "pasenger@keto.com",

"Latitude": 51.320000,

"LocationAdditionalDetails": "<Kiosk KioskLocation=\"On concourse\"/>",

"LocationDesc": "You will find the Self-service Ticket machine located at the front of the station. Aberystwyth Station is open 24 hours a day.",

"LocationName": "London Euston",

"Longitude": 0.500000,

"PhoneNumber": "(703)837.6100",

"ReferenceNumber": "RBK9G589",

"State": "MA",

"Type": "Kiosk",

"Zip": "32432"

},

"FormOfPaymentName": "CorporateAccount",

"FormOfPaymentType": "CA",

"PassPrograms": [

{

"Amount": 2,

"Name": "North America - Tango Plus 200 credits",

"Type": "Credits",

"UserFirstName": "Peter",

"UserLastName": "Neagle"

}

],

"Passengers": [

{

"FirstNameNumber": 0,

"LastNameNumber": 0,

"NameFirst": "Ann",

"NameLast": "Esch",

"NamePrefix": "Sgt.",

"NameRemark": "ABC\*123",

"NameSuffix": "III",

"NameTitle": "Mr.",

"TextName": "EschIII/AnnSgt."

}

],

"PhoneNumbers": [

{

"Description": "Agency",

"PhoneNumber": "703-837-6106",

"Type": "W"

}

],

"RecordLocator": "K5589420201104205418911",

"Segments": {},

"WebAddresses": [

{

"Description": "Work Email",

"Format": "E",

"PassengerRPH": 0,

"Type": "WRK",

"WebAddress": "Michaell@concur.com"

}

]

},

{

"AgencyName": "Outtask Travel Apollo",

"AgencyPCC": "C4I",

"BookingOwner": "ConcurTravel",

"BookingSource": "Manual",

"Charges": {

"Fixed": [

{

"Amount": 4,

"Currency": "USD",

"Description": "Booking fee",

"IsPrimary": false,

"SemanticsCode": "OTHER",

"SemanticsVendorType": "C",

"Vendor": "LH",

"VendorChargeCode": "BF2000"

}

],

"Percent": [

{

"Amount": 3,

"Currency": "USD",

"Description": "Tax",

"IsPrimary": false,

"SemanticsCode": "VAT",

"SemanticsVendorType": "H",

"VendorChargeCode": "Mars tax"

}

]

},

"DateBookedLocal": "2020-10-30T15:54:18.000-00:00",

"DateCreatedUtc": "2020-11-04T20:54:19.000+00:00",

"DateModifiedUtc": "2020-11-04T20:54:19.000+00:00",

"Delivery": {

"AddressLine1": "200 Street 1",

"AddressLine2": "Street 2",

"City": "London",

"Country": "UK",

"Email": "pasenger@keto.com",

"Latitude": 51.320000,

"LocationAdditionalDetails": "<Kiosk KioskLocation=\"On concourse\"/>",

"LocationDesc": "You will find the Self-service Ticket machine located at the front of the station. Aberystwyth Station is open 24 hours a day.",

"LocationName": "London Euston",

"Longitude": 0.500000,

"PhoneNumber": "(703)837.6100",

"ReferenceNumber": "RBK9G589",

"State": "MA",

"Type": "Kiosk",

"Zip": "32432"

},

"FormOfPaymentName": "CorporateAccount",

"FormOfPaymentType": "CA",

"PassPrograms": [

{

"Amount": 2,

"Name": "North America - Tango Plus 200 credits",

"Type": "Credits",

"UserFirstName": "Peter",

"UserLastName": "Neagle"

}

],

"Passengers": [

{

"FirstNameNumber": 1,

"LastNameNumber": 0,

"NameFirst": "Ann",

"NameLast": "Dover",

"NamePrefix": "Sgt.",

"NameRemark": "FINANCE",

"NameTitle": "Mr.",

"TextName": "Dover/AnnSgt."

}

],

"PhoneNumbers": [

{

"Description": "Business",

"PhoneNumber": "800 401 8412",

"Type": "B"

}

],

"RecordLocator": "TF630420201104205418911",

"Remarks": {

"TripLinkRemarks": [

{

"TripLinkRemark": [

{

"Text": "MID OFFICE STUFF"

},

{

"Text": "TESTING"

},

{

"Text": "TO TEST THE MID OFFICE REMARKS"

}

]

}

]

},

"Segments": {

"Ride": [

{

"CancellationPolicy": "Call 20 minutes in advance to avoid charge",

"ConfirmationNumber": "C74450820201104205418911",

"Currency": "USD",

"DateCreatedUtc": "2020-11-04T20:54:19.000+00:00",

"DateModifiedUtc": "2020-11-04T20:54:19.000+00:00",

"DropoffInstructions": "Open door and jump out when speed is below 10 MPH",

"Duration": 2,

"EndAddress": "DCA",

"EndAddress2": "Thomas Ave & Abingon",

"EndCity": "Alexandria",

"EndCityCode": "DCA",

"EndCountry": "US",

"EndDateLocal": "2020-12-16T13:54:18.000-00:00",

"EndDateUtc": "2020-12-16T18:54:18.000+00:00",

"EndLatitude": 38.852843,

"EndLongitude": -77.038536,

"EndPostalCode": "22202",

"EndState": "VA",

"MeetingInstructions": "Meet by hot dog stand in front of building",

"Miles": 10,

"Name": "Yellow Cab",

"NumPersons": 2,

"NumberOfHours": 0.03333333333333333,

"PhoneNumber": "703-837-6100",

"PickupInstructions": "Pickup at given address",

"Rate": 24,

"RateDescription": "Hourly rate",

"RateType": "H",

"StartAddress": "209 Madison Street",

"StartCity": "Alexandria",

"StartCityCode": "DCA",

"StartCountry": "US",

"StartDateLocal": "2020-12-16T11:54:18.000-00:00",

"StartDateUtc": "2020-12-16T16:54:18.000+00:00",

"StartLatitude": 38.814098,

"StartLongitude": -77.040939,

"StartPostalCode": "22314",

"StartState": "VA",

"TimeZoneId": 25,

"Vendor": "$R",

"VendorName": "RideCharge"

}

]

},

"WebAddresses": [

{

"Description": "Home AIM",

"Format": "I",

"Type": "RES",

"WebAddress": "mloreOuttask"

}

]

}

],

"Comments": "Generated from - MakeRandomItineraryWithSpecifiedSegment: 401722883",

"CustomAttributes": [

{

"Data": "1914163392\_1398750079\_63698747",

"DataType": "Enumeration",

"DisplayOnItinerary": true,

"DisplayTitle": "Title\_1882786037",

"Name": "Custom\_1047320375"

},

{

"Data": "2144949539\_499322633\_1732005039",

"DataType": "Enumeration",

"DisplayOnItinerary": false,

"DisplayTitle": "Title\_559871572",

"Name": "Custom\_1726736195"

},

{

"Data": "683267112\_1599532776\_1875679442",

"DataType": "Enumeration",

"DisplayOnItinerary": true,

"DisplayTitle": "Title\_119690786",

"ExternalId": 1088610323,

"Name": "Custom\_1899779539"

},

{

"Data": "56781286\_861560873\_80874171",

"DataType": "Enumeration",

"DisplayOnItinerary": true,

"DisplayTitle": "Title\_390352466",

"Name": "Custom\_1579191469"

},

{

"Data": "2067714888\_1442184475\_1016439405",

"DataType": "String",

"DisplayOnItinerary": true,

"DisplayTitle": "Title\_1544667785",

"Name": "Custom\_1166600630"

},

{

"Data": "351251522\_246017323",

"DataType": "Numeric",

"DisplayOnItinerary": true,

"DisplayTitle": "Title\_19093583",

"Name": "Custom\_750631891"

},

{

"Data": "821749202\_364197364\_762362918",

"DataType": "Enumeration",

"DisplayOnItinerary": false,

"DisplayTitle": "Title\_1467115321",

"ExternalId": 1986175686,

"Name": "Custom\_2007441754"

},

{

"Data": "473097600\_978534217",

"DataType": "Numeric",

"DisplayOnItinerary": true,

"DisplayTitle": "Title\_1137473737",

"Name": "Custom\_1832693806"

},

{

"Data": "1679301535\_433494967\_1252049254",

"DataType": "Enumeration",

"DisplayOnItinerary": true,

"DisplayTitle": "Title\_15390255",

"Name": "Custom\_1100573803"

}

],

"DateBookedLocal": "2020-10-30T15:54:18.000-00:00",

"DateCreatedUtc": "2020-11-04T20:54:19.000+00:00",

"DateModifiedUtc": "2020-11-04T20:54:51.026+00:00",

"Description": "Trip from somewhere to somewhere else 506643842",

"EndDateLocal": "2020-12-16T17:15:18.000-00:00",

"EndDateUtc": "2020-12-17T01:15:18.000+00:00",

"id": "https://us.api.concursolutions.com/travel/v4/trips/08d019f6-8c7e-5e96-bbf4-04726ac5def2",

"ItinLocator": "gWutRRSrKhRhhY9lx0GUfP1YhfAXDTBtYwFTA$pKU",

"ProjectName": "Big Project # 43534534",

"StartDateLocal": "2020-12-11T15:54:18.000-00:00",

"StartDateUtc": "2020-12-11T20:54:18.000+00:00",

"TravelRequestId": "TR98318081",

"TripName": "My Random Trip #266255879",

"TripStatus": 0,

"UserLoginId": "[itintrips@coreprofiletesting.com](mailto:itintrips@coreprofiletesting.com)”

}

We may not need everyone of these lines but it is good to understand what fields and such we would need.

Looking at Concur a competitor in this space, there is public

A thought summary

* Departure from Portland on September 5th, 2023: 9:00 AM
* Arrival in Miami on September 5th, 2023: 10:00 AM
* Seat Preference: Window seat
* Meeting in Miami on September 5th, 2023: "Bananas" at 3:00 PM (Miami local time)
* Departure from Miami to Denver on September 6th, 2023: 8:00 PM
* Meeting in Denver on September 6th, 2023: "Potato" at 4:00 PM (Denver local time)
* Departure from Denver to NYC on September 6th, 2023: 10:00 PM

{

"flights": [

{

"source": "Portland",

"destination": "Miami",

"airline": "American Airlines",

"departure\_time": "2023-09-05T09:00:00",

"arrival\_time": "2023-09-05T10:00:00",

"seat\_preference": "Window",

"meeting\_event": {

"title": "Financial Meeting",

"time": "2023-09-05T15:00:00",

"timezone": "America/New\_York"

}

},

{

"source": "Miami",

"destination": "Denver",

"airline": "American Airlines",

"departure\_time": "2023-09-06T20:00:00",

"arrival\_time": "2023-09-06T23:00:00",

"seat\_preference": "Window",

"meeting\_event": {

"title": "VC Meeting",

"time": "2023-09-06T16:00:00",

"timezone": "America/Denver"

}

},

{

"source": "Denver",

"destination": "New York City",

"airline": "American Airlines",

"departure\_time": "2023-09-06T22:00:00",

"arrival\_time": "2023-09-07T05:00:00",

"seat\_preference": "Window"

}

],

"average\_flight\_time": 5.5,

"timezones": {

"Portland": "America/Los\_Angeles",

"Miami": "America/New\_York",

"Denver": "America/Denver",

"New York City": "America/New\_York"

}

}

## Ticketing with Amadeus

Lots of rules are in the Amadeus documentation.

## Prompts

These are the test prompts for customers.

| **Prompt Number** | **Prompt** | **Tags** | **Expected Result (LLM Request Format and Booking / Ticketing Response)** |
| --- | --- | --- | --- |
| 1 | I would like to fly to Chicago from New York and on 15th Oct 2023 but have series of meetings in Chicago to attend so I need early morning flights and need to arrive by 10:00AM so I have time for my appointments and would like to come next day later in the evening after 4:00 PM and must be back by 10:00 PM. |  |  |
| 2 | I have a conference to attend in London on 20th Oct so I would like to fly on 18th so I get a day rest when I attend that conference and I would be returning back after a week time from the date I will arrive there. I would prefer One world partners and a nonstop with business class. |  |  |
| 3 | I have a business trip coming up in San Francisco and Las Vegas. Meeting in San Francisco will be on 18th Oct and then in Las Vegas will be on 19thOct. So, I need an option where I can fly from New York to San Francisco on 18th Oct and must arrive before noon and then leave San Francisco to Las Vegas after 06:00 PM and then from Las Vegas back to New York by 19th October again after 06:00 PM. |  |  |
| 4 | There is an expected meeting came up and I need to book ticket for my executive whichever is next available flight today. I need the nonstop flight from Los Angeles to Newark that will arrive anytime today in New Jersey as meeting is tomorrow early morning and would like to return next day. |  |  |
| 5 | I am looking for a flight from Los Angeles to Paris for leaving on 02nd Oct and return on 13th Oct.I need the cheapest flight as I am budget constraint and the shortest flight or a non stop which ever is cheapest. Also, I am flexible with my return date for a day or two plus or minus. |  |  |
| 6 | I am looking for a flight to Santorini, Greece from New York preferably JFK airport. I would like to fly on 10th Oct and return on 20th Oct and not flexible with the dates. I would like to go with American airlines, and I have 500K miles in my account so would like to use my miles for a first class and need to book a lounge in my layover. | (Using points, needs to book a lounge in my layover - might be included in first class) |  |
| 7 | I would like to fly from Boston to Shanghai and have my pet dog traveling with me. So, can you tell me which airline I can travel on with minimal paperwork and better flights. | (Tag - External airlines information and minimize jet lag information) |  |
| 8 | I want to go to India as have a series of meetings coming up. I want to fly from JFK and would like to go to Delhi on 02nd December but would like to minimize jet lag. So, can you get me the options that allow me for a gradual adjustment due to different time zone. |  |  |
| 9 | I am looking for a flight from CLE to San Diego that leaves around 1pm or later. For Tuesday, I want a flight that leaves San Diego between 8-11am. Shorter total flights are preferred. | (Group booking, wheel chair assistance) |  |
| 10 | I am admin of XYZ organization, and I would like to send 18 group members for an event in Los Angeles. They all will be flying from Boston to Los Angeles on 12th Oct and returning on 15th Oct. They all need seats together and couple of them might need wheel chair assistance. |  |  |
| 11 | I have flight booked with you guys for the director of my company and his confirmation number is ABCXYZ.I would like to upgrade his ticket to first class for long haul journeys and since he is platinum member with American Airlines so It will be good if you can use his miles. | (Upgrade a Flight with confirmation number and points if possible) |  |
| 12 | I need a flight from Boston to Frankfurt preferably with United or Lufthansa and I need the fare that allows me to change my reservation as much as I want as my schedule is not fixed. | (Ticket where changes are allowed, historical for if flight delayed) |  |
| 13 | I want to book a flight from Los Angeles to Miami to leave on 20th Dec and ret on 28th Dec. I see a flight with United and one stop in Chicago with a layover of 50 mins. I want to know what if first flight get delayed? |  |  |
| 14 | I would like to business class ticket from Chicago to Dubai leaving on 10th Oct and returning on 16th Nov and want a lie flat bed with all business class amenities. | (Needs external information airline of lie flat bed with business class) |  |
| 15 | I would like to fly from San Francisco to JFK on 23rdSep and return on 30th Sep for my business trip and would like a hotel preferably near downtown and 4-star hotel will work for me. | (Hotel booking for downtown) |  |
| 16 | I would like to get a best package deal in San Francisco going from Boston on 20th Oct and returning on 04th November and I want early morning departure on my departure and midafternoon flight on my return. I also need a hotel nearby to airport only and 3-star hotel as I am budget constraint. | (Hotel book with budget constraints) |  |
| 17 | (Group, doesn’t have where they are from, hotel and bookings) |  |  |
| 18 | Can you assist with last-minute flight and hotel bookings for an executive team attending an important meeting in London tomorrow?§ |  |  |
| 19 | I need to book a package for our annual corporate retreat in France. What options do you have for a destination that's great for team building activities? | (External data for team building activities) |  |
| 20 | I would like to book a flight for 6 of my clients going same dates that is 28th Sep but 2 will fly from Boston airport and 4 will fly from Chicago airport and all will be going to Phoenix, Arizona and returning on same dates which is 01st Oct to the airports they are flying to. | (Group, different flights ) |  |
| 21 | Do you offer any complimentary shuttle services between the hotel and the conference venue? | (Query external data hotel and conference venue) |  |
| 22 | Can you provide information on the available transportation options from the airport to the hotel? I have a flight booked already to Los Angeles and I am staying in Hilton and the dates are 23rd Sep through 25th Sep. | (External data) |  |
| 23 | I would like to book a flight from Las Vegas to Kathmandu to leave on 20th Aug and return on 15th September but my passport will be expiring on 20thNov. Will that be an issue? | (External data - passport regulations for Kathmandu) |  |
| 24 | I would like to book a flight from Los Angles to Manila to leave on 26th July and wants to 3 days stopover in Tokyo while going there and wants to come back from DPS on 15th Aug. | (A multi-way and layovers) |  |
| 25 | I would like to book flight to go from Boise to Sydney for 16 days. I am quite flexible with the dates, but my budget is only $1500.So, you can look for any month Between September and October. | (Budget constraints - no flight constraints) |  |
| 26 | I would like to book a flight for group of 20 people from Detroit to Los Angeles to leave on 15th Aug and return on 21st August. My dates are flexible with plus or minus 1 day and my budget is USD 6000, and my requirements are all needed to be seated together or nearby and we all need 1 check in bag and 1 carry-on bag for each person. | (Group booking) |  |
| 27 | I need to book a flight from Washington Dulles to Helsinki to leave on 22nd Sep and return on 26thOctober with 1 check in bag and also, I would like to take a bicycle with me. | (External data bicycle and 1 check in bag) |  |
| 28 | I want to book a flight from Los Angeles to Miami to leave on 20th Dec and ret on 28th Dec. I see a flight with United and one stop in Chicago with a layover of 50 mins. I want to know what if first flight get delayed? | (External data on delays) |  |
| 29 | I would like to book a flight from Las Vegas to Frankfurt for 2 adults, 1 child and 1 infant leaving on 20th Sep and returning on 28th November and wants all seats together and need 1 check in bag for each passenger. | (Seats together, infant and 1 check in bag) |  |
| 30 | What's the process for adding extra nights to the hotel stay I want to extend my business trip? | (External data - extending hotel stay) |  |
|  | Can you provide options for hotel accommodations and car rentals for a week-long business trip to Chicago in October first week? Going from Phoenix to Chicago departing on 03rd Oct and returning on 09thOct and I am flexible with a day or two. | (Hotel booking, car rentals) |  |
| 31 | Do you have last-minute availability for a hotel and car rental in Dallas for a business meeting tomorrow? | (Hotel and car bookings) |  |
| 32 | I would like to book a flight for me and my partner departing on 21st Oct from New York to Istanbul there I will stay for 3 days and then will leave Istanbul to Paris on 26th Oct and then returning on 01st Nov from Paris to New York. On my return I will prefer nonstop flight. What are my options? | (Flight options, need to understand the number of people flying and give options) |  |
| 33 | Do you offer options for environmentally friendly or electric car rentals? | (External data) |  |
| 34 | I want to book a hotel in Boston near XYZ lab on 25thSep for 3 nights.I need a hotel that will be pet friendly. | (Flight, hotel, requirements: pet friendly relates to hotel) |  |
| 35 | Can you provide details on the best business or first-class deals for a team flying to a conference in Tokyo next week? Departure will be from New York to Tokyo (any airport Haneda or Narita) but nonstop flight on 24th September and returning on 02nd October. | (Flight, external data - best business or firstl class) |  |
| 36 | Do you have any last-minute availability for business or first-class seats to Los Angeles for a meeting tomorrow? Flying from Chicago to Los Angeles 16thSeptember and returning on 20th September. | (Flying right now) |  |
| 37 | What's the policy for upgrading to business or first class using frequent flyer miles or points? | (External data - policy) |  |
| 38 | Can you recommend airports with dedicated lounges and services for business or first-class travelers? | (External data - airport lounges) |  |
| 39 | What's the most cost-effective way to book business or first-class flights for a two-month international project to Lagos beginning next month in mid? | (Cost effectiveness, Maybe external data) |  |
| 40 | I would like to book a hotel in Bahamas for 4 adults and need adults only hotel. I will be checking in on 26th Oct and checking out on 29th Oct. | (Hotel, External data for adults only hotel) |  |
| 41 | I would like to book a flight and hotel for 2 people going from Houston to Madrid and my budget is $4500 for both. I would like to fly from Houston to Madrid on 03rd December and would like to return on 03rd January and will need a hotel near to the market and preferably 4 star hotel which should be pet friendly as will be taking a chihuahua that is 8 months old. | (Flight, hotel, pet) |  |

## LLM Prompts

There is a combination of prompts that take into account the personalization JSON, plus the customer prompt, plus the reasoning of the prompt, then the prompt tuning.

## Validation

To make systems robust, validation is important. While there might be a faster way with Neurosymbolic here are a number of validation routines that need to be written. When it comes to robust coding, the code to run to validate/robust is 10-20x the code. This means using data models, schemas or Neurosymbolic will help reduce this coding effort.

Some examples of validation routines needed:

isValidAirport(airportCode): This will check if the provided airport code is valid (IATA format: 3 uppercase letters).

isValidDate(date): This will check if the provided date is valid and in the format "YYYY-MM-DD".

isValidTime(time): Check if the time is in HH:MM format.

isValidPassengerCount(count): Ensure that the passenger count is a positive number.

isValidClass(flightClass): Validate the flight class (e.g., Economy, Business).

isValidEmailAddress(email): Validate the email address format.

isValidPhoneNumber(phone): Validate the phone number format.

isValidCountry(countryCode): Validate the country code format (e.g., US, GB).

isValidCurrency(currencyCode): Validate the currency code (e.g., USD, EUR).

isValidDuration(duration): Check if the duration is in HH:MM format.

isValidFlightNumber(flightNumber): Validate the flight number.

isValidHotelName(hotelName): Basic check for hotel name validity.

isValidRoomType(roomType): Validate common room types (e.g., Single, Double).

isValidCreditCard(cardNumber): Basic check for credit card number validity.

isValidCVV(cvv): Validate the CVV of a credit card.

isFutureDate(date): Ensure the provided date is in the future.

isValidBookingCode(bookingCode): Check the format of a booking code.

isValidBaggageCount(count): Ensure the baggage count is a positive number.

isValidSeatPreference(seatType): Validate seat preferences (e.g., Window, Aisle).

isValidMealPreference(mealType): Validate meal preferences (e.g., Vegetarian, Vegan, Non-Vegetarian).

isValidLoyaltyNumber(loyaltyNumber): Validate the format of a loyalty program number.

isValidPromoCode(promoCode): Check the format of a promotional code.

isValidRating(rating): Validate the rating (e.g., 1 to 5 stars).

isValidAdultCount(adultCount): Ensure the adult count is a positive number.

isValidChildCount(childCount): Ensure the child count is a positive number.

isValidPrice(price): Check if the price is a positive value.

isValidTravelReason(reason): Validate common travel reasons (e.g., Business, Leisure).

isValidLayoverDuration(duration): Check if the layover duration is in a valid range.

isValidTravelDocument(documentType): Validate travel document types (e.g., Passport, Visa).

isValidName(name): Basic check for passenger name validity.

isValidHotelFacility(facility): Validate common hotel facilities (e.g., Pool, Gym).

isValidHotelChain(chainName): Basic check for hotel chain name validity.

isValidCheckInDate(checkIn): Ensure the check-in date for a hotel is valid.

isValidCheckOutDate(checkOut): Ensure the check-out date for a hotel is valid and is after the check-in date.

isValidPaymentMethod(method): Validate payment methods (e.g., Credit Card, Debit Card, PayPal, etc.).

isValidDestination(destination): Basic validation for destination names (e.g., city names).

isValidOrigin(origin): Basic validation for origin names (e.g., city names).

isValidAmenity(amenity): Validate common hotel amenities (e.g., WiFi, Air Conditioning, Breakfast Included).

isValidGender(gender): Validate gender options (e.g., Male, Female, Non-Binary, Prefer not to say).

isValidSpecialRequest(request): Check the format of special requests made by the passenger.

isValidRefundPolicy(policy): Validate common refund policies (e.g., Non-Refundable, Partially Refundable).

isValidBoardingZone(zone): Validate boarding zones (e.g., Zone 1, Priority).

isValidLanguagePreference(language): Validate language preferences (e.g., English, Spanish, French).

isValidStopOverCity(city): Validate names of cities where stopovers or layovers might occur.

isValidPassengerAge(age): Ensure age is a positive number and within a realistic range.

isValidHotelStarRating(rating): Validate hotel star ratings (e.g., 1 to 5 or 1 to 7 stars, depending on the scale you use).

isValidVoucher(voucherCode): Validate the format of gift voucher codes or discount codes.

isValidFlightDuration(duration): Check if the flight duration format is correct and logical.

isValidInfantCount(infantCount): Ensure the infant count is a non-negative number and possibly validate against adult count.

isValidCabinLuggageCount(luggageCount): Validate the number of cabin luggage items.

isValidSpecialAssistanceType(assistanceType): Validate types of special assistance (e.g., Wheelchair, Guide Dog).

isValidTransferOption(transferType): Validate transfer options (e.g., Shuttle, Private Car, None).

isValidAirlineName(airlineName): Basic check for airline name validity.

isValidVisaType(visaType): Validate common visa types (e.g., Tourist, Business).

isValidInsuranceOption(option): Validate insurance options for passengers (e.g., Medical, Cancellation, Baggage).

isValidSeatNumber(seatNumber): Validate seat numbers on a flight.

isValidArrivalTerminal(terminal): Validate terminal numbers or names for arrivals.

isValidDepartureTerminal(terminal): Validate terminal numbers or names for departures.

isValidBookingStatus(status): Validate booking statuses (e.g., Confirmed, Pending, Canceled).

isValidDietaryPreference(preference): Validate dietary preferences for in-flight meals (e.g., Gluten-Free, Halal).

isValidConnectionType(connection): Validate types of flight connections (e.g., Direct, One-Stop).

isValidInFlightEntertainmentOption(option): Validate in-flight entertainment options (e.g., Movie, Music, None).

isValidPaymentStatus(status): Validate payment statuses (e.g., Paid, Pending, Failed).

isValidBedType(bedType): Validate bed types in hotel rooms (e.g., King, Queen, Twin).

isValidViewType(viewType): Validate room view types (e.g., Sea View, City View).

isValidLoyaltyTier(tier): Validate tiers in loyalty programs (e.g., Silver, Gold, Platinum).

isValidCancellationReason(reason): Validate reasons for cancellation (e.g., Personal, Weather, Operational).

isValidInFlightService(service): Validate in-flight services (e.g., WiFi, USB Charging, In-seat Power).

isValidPackageType(packageType): Validate types of packages offered (e.g., All-Inclusive, Bed & Breakfast).

isValidBookingPlatform(platform): Validate the platform where the booking was made (e.g., Website, Mobile App, Agent).

isValidPetPreference(pet): Validate if passengers are traveling with pets and what type (e.g., Dog, Cat).

isValidTravelPackage(package): Validate travel package IDs or codes.

isValidDiscountType(discountType): Validate types of discounts applied (e.g., Early Bird, Last Minute).

isValidBillingAddress(address): Basic check for validity of a billing address.

isValidContactPreference(preference): Validate contact preferences (e.g., Email, SMS, Phone Call).

isValidDocumentExpiryDate(date): Ensure the travel document (e.g., passport) hasn't expired.

isValidTaxiPreference(taxiType): Validate types of taxi services (e.g., Shared, Private).

isValidEventBooking(eventType): Validate event types when booking in tandem with travel (e.g., Concerts, Sports).

isValidInclusion(inclusion): Validate inclusions in travel packages (e.g., Meals, Tours).

isValidExclusion(exclusion): Validate exclusions in travel packages.

isValidGroupSize(size): Validate the number of people for group bookings.

isValidFeedbackType(feedbackType): Validate types of feedback or reviews (e.g., Complaint, Praise, Suggestion).

## Misc

## Some emojis to add to the output text.

For Flights:

✈️ (Airplane)

🛫 (Airplane Departure)

🛬 (Airplane Arrival)

🎫 (Ticket)

🌍 (Earth Globe Europe-Africa) – you can choose the globe that's most relevant to your primary audience.

💺 (Seat) – indicating the seating in an airplane.

🛄 (Baggage Claim)

🛂 (Passport Control)

🧳 (Luggage)

For Hotels:

🏨 (Hotel)

🛏️ (Bed)

🚪 (Door)

🔑 (Key)

🛎️ (Bellhop Bell)

🛁 (Bathtub) – to signify amenities.

🍳 (Cooking) – if the hotel has kitchen facilities or complimentary breakfast.

🏊 (Swimmer) – if there's a pool.

🏋️ (Weight Lifter) – for gym facilities.

General Travel:

🌴 (Palm Tree) – for tropical destinations.

🏔️ (Mountain) – for hill stations or mountainous regions.

🗺️ (World Map)

🚌 (Bus) – for shuttle services.

🚖 (Taxi) – if taxi services are connected or recommended.

🌅 (Sunrise) – for morning flights or morning hotel check-ins.

🌇 (Sunset) – for evening flights or relaxing evenings at the hotel.

💼 (Briefcase) – for business travel.

🎒 (Backpack) – for backpackers or budget travelers.

# Amadeus API Calls

These fields seem the most important.

- `originLocationCode`: The IATA code of the departure airport.

- `destinationLocationCode`: The IATA code of the arrival airport.

- `departureDate`: The date of departure.

- `returnDate`: The return date, if it's a round trip.

- `adults`: Number of adult passengers.

- `children`: Number of child passengers.

- `infants`: Number of infants.

- `travelClass`: The class of the cabin (Economy, Premium Economy, Business, or First Class).

- `includedAirlines`: If you want to filter for specific airlines.

- `nonStop`: Boolean to indicate if you want only non-stop flights.

## Some Assumptions

For the Amadeus search there will be some assumptions made:

* That any trips that have the some departure city, will use the person’s profile airport first. If there are no flights from that airport for that city, then there will be prompting for which airline to use.
* For destination, by default we take the most popular airport code for this city. Unless, specific in the prompt or specific in the profile.
* Included airlines should be where possible to use their profile. If no airlines fly from their list of airlines, then a new search with any airlines is done and prompted by Rainbird
* Travel class should use were possible default profile. If no flights have their preferred class, then prompt for class or different airline or such?
* NonStop should be from the default profile, if there is no non-stop, then prompt for whether to book this with stops or do a different search.

## Reference Data Needed

* IATA code for airports
* Most common IATA for airports per city - top two or so. I have flight data from all airlines in the world, with travel for 2022.
* We can check routes if possible and number of hours.

The Amadeus API for flights is a set of RESTful web services provided by Amadeus, a global travel technology company, for searching and booking flights, among other travel services.

To call the Amadeus Flight Search API, you typically need to follow these steps:

1. \*\*Authentication\*\*: Obtain an API key by registering on the Amadeus for Developers portal. Use this key to authenticate your API requests.

2. \*\*Create a search\*\*: Make a GET request to the flight search endpoint, usually with parameters like origin, destination, departure date, number of passengers, and travel class.

3. \*\*Parse the response\*\*: Handle the JSON response which contains the available flights along with details such as price, duration, and layovers.

4. \*\*Flight Offers Search\*\*: If you need more detailed information, including seat availability, you can use the Flight Offers Search.

To get preference for a specific seat, you would typically:

1. \*\*Use the SeatMap Display\*\*: After finding a flight, you can call the SeatMap Display API to retrieve the seat map for a specific flight.

2. \*\*Select a seat\*\*: Once you have the seat map, you can provide the preference for a specific seat when you create a booking.

Here's an example of how a basic flight search API call might look using curl (note that this is just a conceptual example and may not match the current API specifications):

```bash

curl -X GET "https://test.api.amadeus.com/v2/shopping/flight-offers?originLocationCode=NYC&destinationLocationCode=LAX&departureDate=2023-12-15&adults=1&max=10" -H "Authorization: Bearer {your\_access\_token}"

```

In the above request, you would replace `{your\_access\_token}` with the actual token you received after authenticating with the Amadeus API.

As for the fields needed, these are some of the common parameters:

- `originLocationCode`: The IATA code of the departure airport.

- `destinationLocationCode`: The IATA code of the arrival airport.

- `departureDate`: The date of departure.

- `returnDate`: The return date, if it's a round trip.

- `adults`: Number of adult passengers.

- `children`: Number of child passengers.

- `infants`: Number of infants.

- `travelClass`: The class of the cabin (Economy, Premium Economy, Business, or First Class).

- `includedAirlines`: If you want to filter for specific airlines.

- `nonStop`: Boolean to indicate if you want only non-stop flights.

To specify seat preference, you would have to deal with the booking process, which is more complex and involves additional API calls.

To get started, you should visit the Amadeus for Developers website, check their documentation, and sign up to get your API key. They also provide SDKs for different programming languages, which can simplify the API call process. Keep in mind that the exact fields and methods to use can change, so the Amadeus API documentation should be your primary source of up-to-date information.