Mars Immigration System

Main Vision

Colony has its known rules for applications

+

Colony has known limited resources

=

Pre-process visa application

High Level Architecture

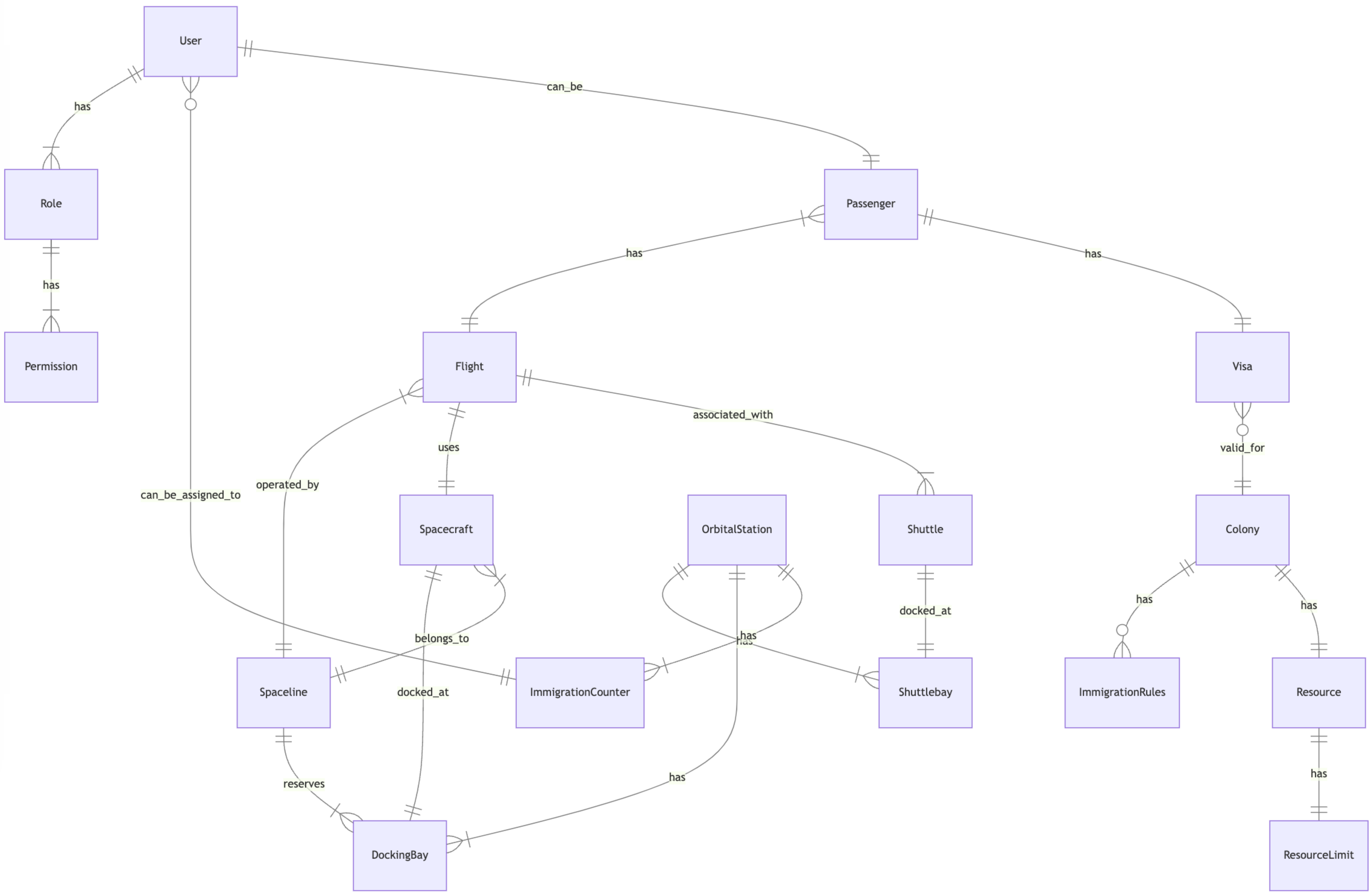
- Local Kubernetes Cluster
- Microservices
- Frontend applications
- RBAC
- 3rd party monitoring/logging/traffic managing
- CI/CD pipeline

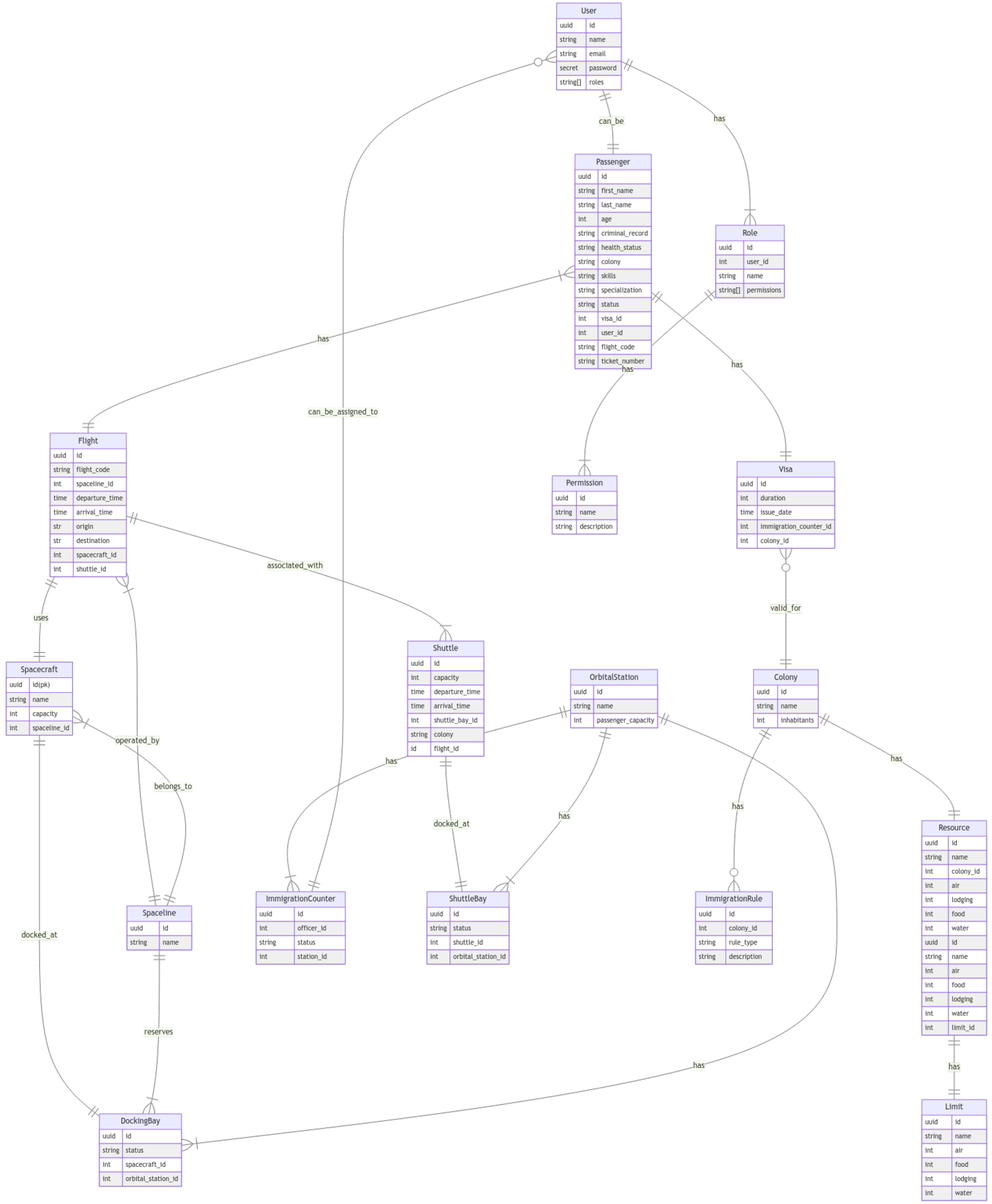
Roles

- Passegners
- Immigration Officers
- Spaceline Dispatcher
- Colony Officials
- Station Managers
- Shuttle Dispatchers
- System admin

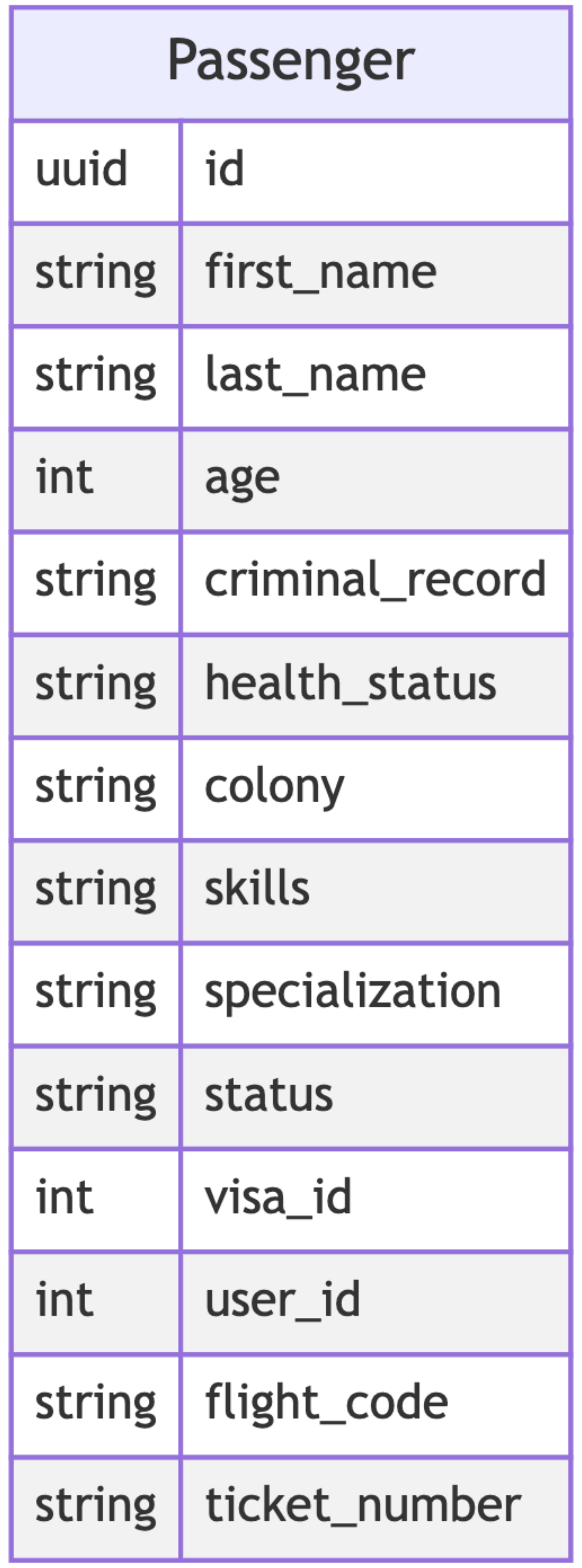
Main Business/Critical Processes

- Data Provisioning
- Passenger Registration (not implemented)
- Visa Application
- Visa Approval
- Resources Allocation
- Spacecraft Operations (not implemented)
- Shuttle Operations (not implemented)


User
has
can_be
Role
has
Permission
Passenger
has
has
Flight
Visa
associated_with
uses
operated_by
can_be_assigned_to
valid_for
Spacecraft
OrbitalStation
Shuttle
Colony
docked_at
has
has
belongs_to
has
has
Spaceline
docked_at
ImmigrationCounter
Shuttlebay
ImmigrationRules
Resource
reserves
has
has
DockingBay
ResourceLimit

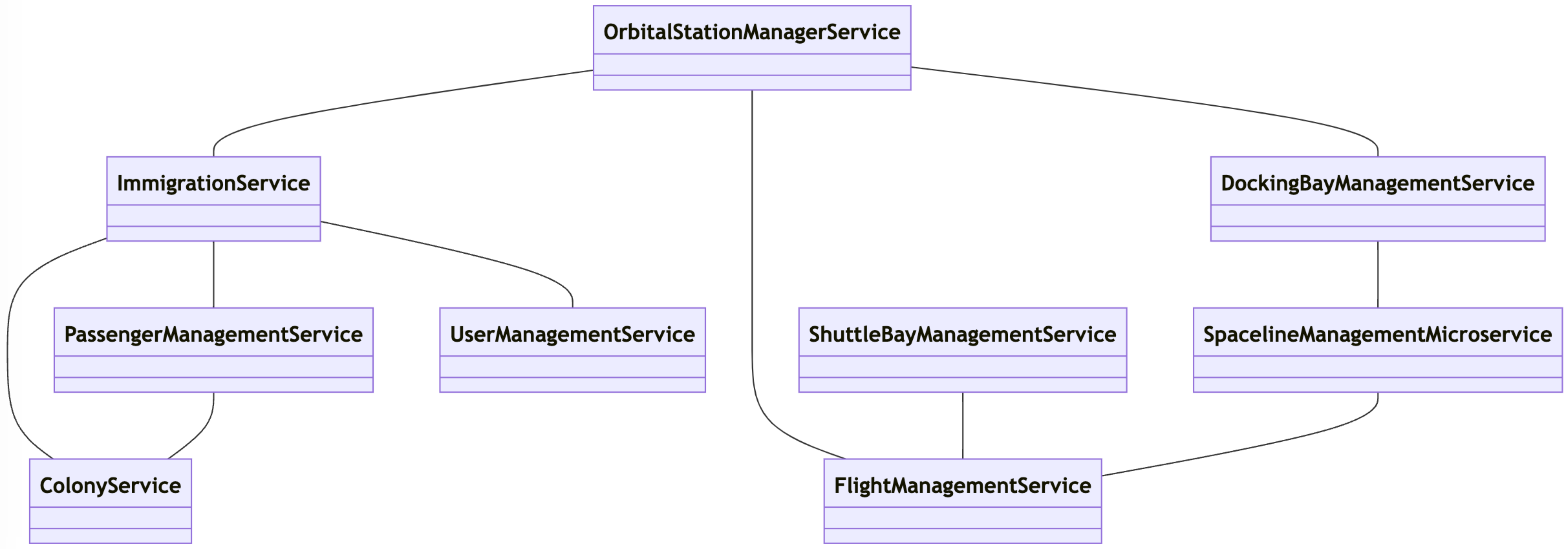

User
uuid id
string name
string email
secret password
string[] roles
can_be
has
Passenger
uuid id
string first_name
string last_name
int age
string criminal_record
string health_status
string colony
string skills
string specialization
string status
int visa_id
int user_id
string flight_code
string ticket_number
Role
uuid id
int user_id
string name
string[] permissions
has
has
has
can_be_assigned_to
Flight
uuid id
string flight_code
int spaceline_id
time departure_time
time arrival_time
str origin
str destination
int spacecraft_id
int shuttle_id
Permission
uuid id
string name
string description
Visa
uuid id
int duration
time issue_date
int immigration_counter_id
int colony_id
associated_with
uses
valid_for
Shuttle
uuid id
int capacity
time departure_time
time arrival_time
int shuttle_bay_id
string colony
id flight_id
Spacecraft
uuid id(pk)
string name
int capacity
int spaceline_id
OrbitalStation
uuid id
string name
int passenger_capacity
Colony
uuid id
string name
int inhabitants
operated_by
has
belongs_to
has
docked_at
has
has
Resource
uuid id
string name
int colony_id
int air
int lodging
int food
int water
uuid id
string name
int air
int food
int lodging
int water
int limit_id
ImmigrationCounter
uuid id
int officer_id
string status
int station_id
ShuttleBay
uuid id
string status
int shuttle_id
int orbital_station_id
ImmigrationRule
uuid id
int colony_id
string rule_type
string description
Spaceline
uuid id
string name
docked_at
reserves
has
has
DockingBay
uuid id
string status
int spacecraft_id
int orbital_station_id
Limit
uuid id
string name
int air
int food
int lodging
int water

Entity example - Passenger

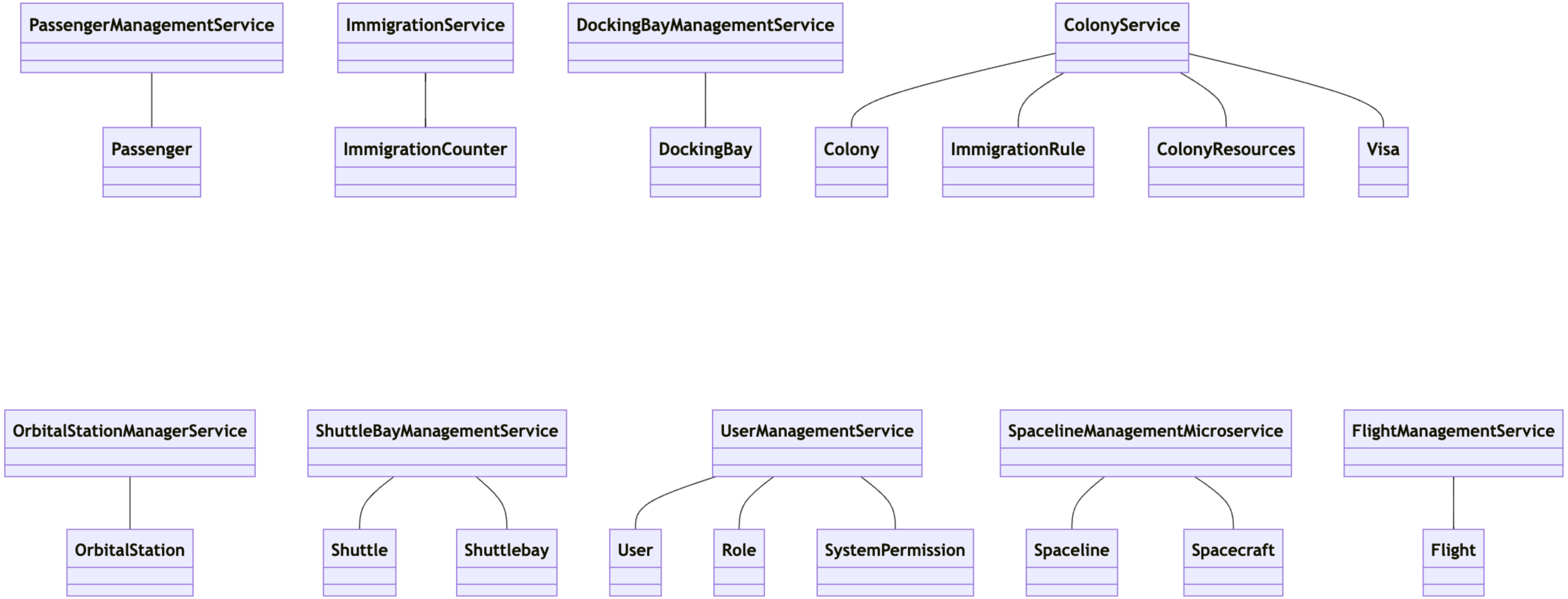


Passenger	
uuid	id
string	first_name
string	last_name
int	age
string	criminal_record
string	health_status
string	colony
string	skills
string	specialization
string	status
int	visa_id
int	user_id
string	flight_code
string	ticket_number

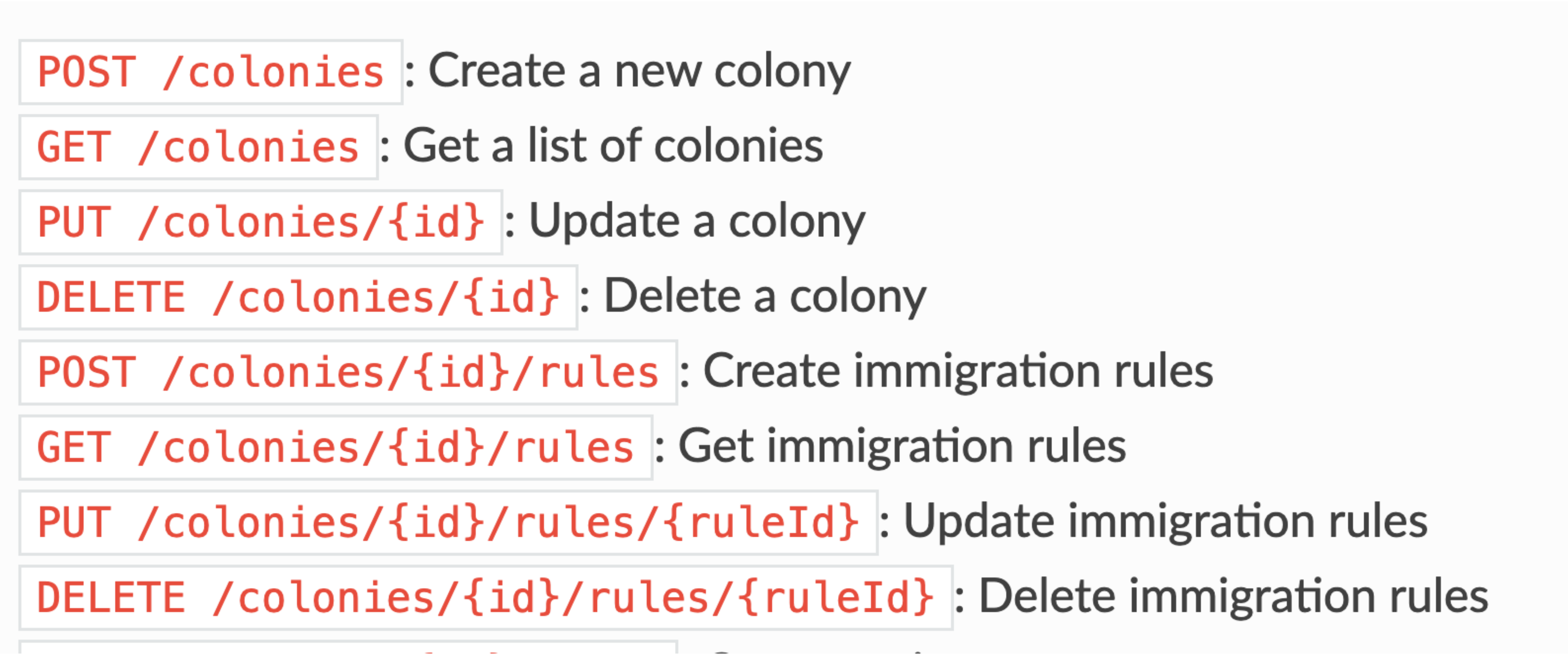
Microservices


OrbitalStationManagerService
ImmigrationService
DockingBayManagementService
PassengerManagementService
UserManagementService
ShuttleBayManagementService
SpacelineManagementMicroservice
ColonyService
FlightManagementService

Services - entities coverage



We are using REST API



`POST /colonies` : Create a new colony

`GET /colonies` : Get a list of colonies

`PUT /colonies/{id}` : Update a colony

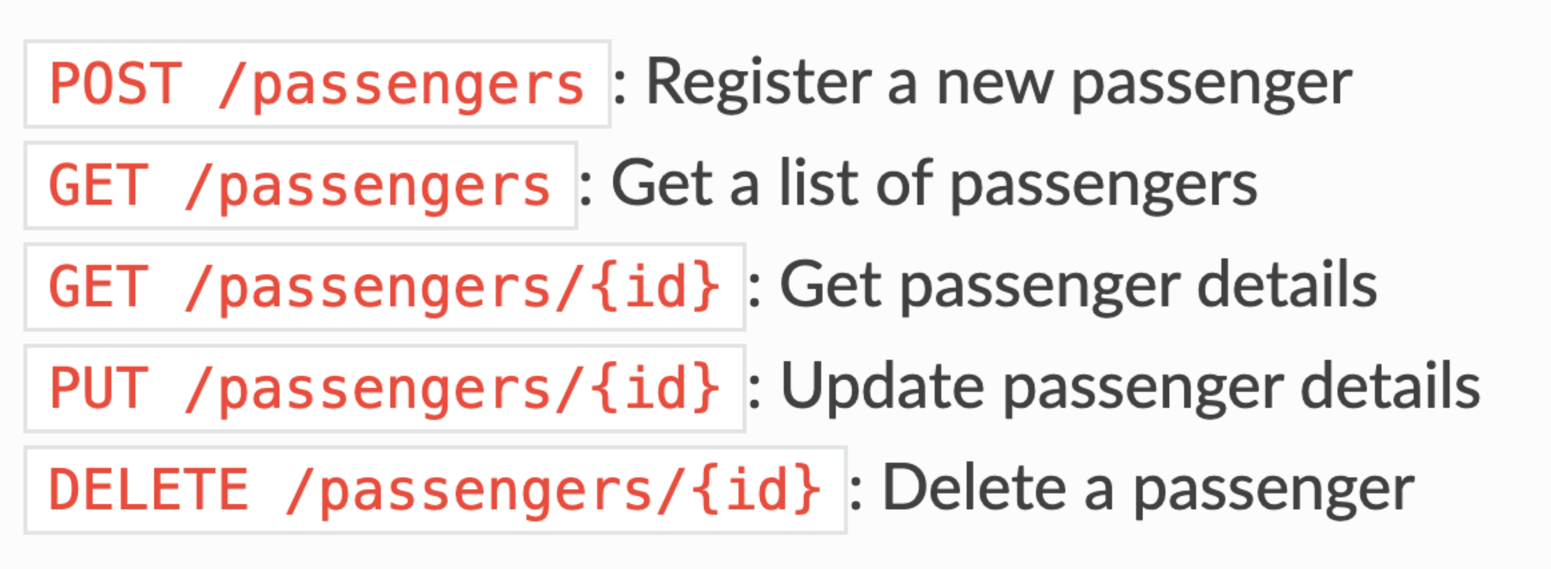
`DELETE /colonies/{id}` : Delete a colony

`POST /colonies/{id}/rules` : Create immigration rules

`GET /colonies/{id}/rules` : Get immigration rules

`PUT /colonies/{id}/rules/{ruleId}` : Update immigration rules

`DELETE /colonies/{id}/rules/{ruleId}` : Delete immigration rules



`POST /passengers` : Register a new passenger

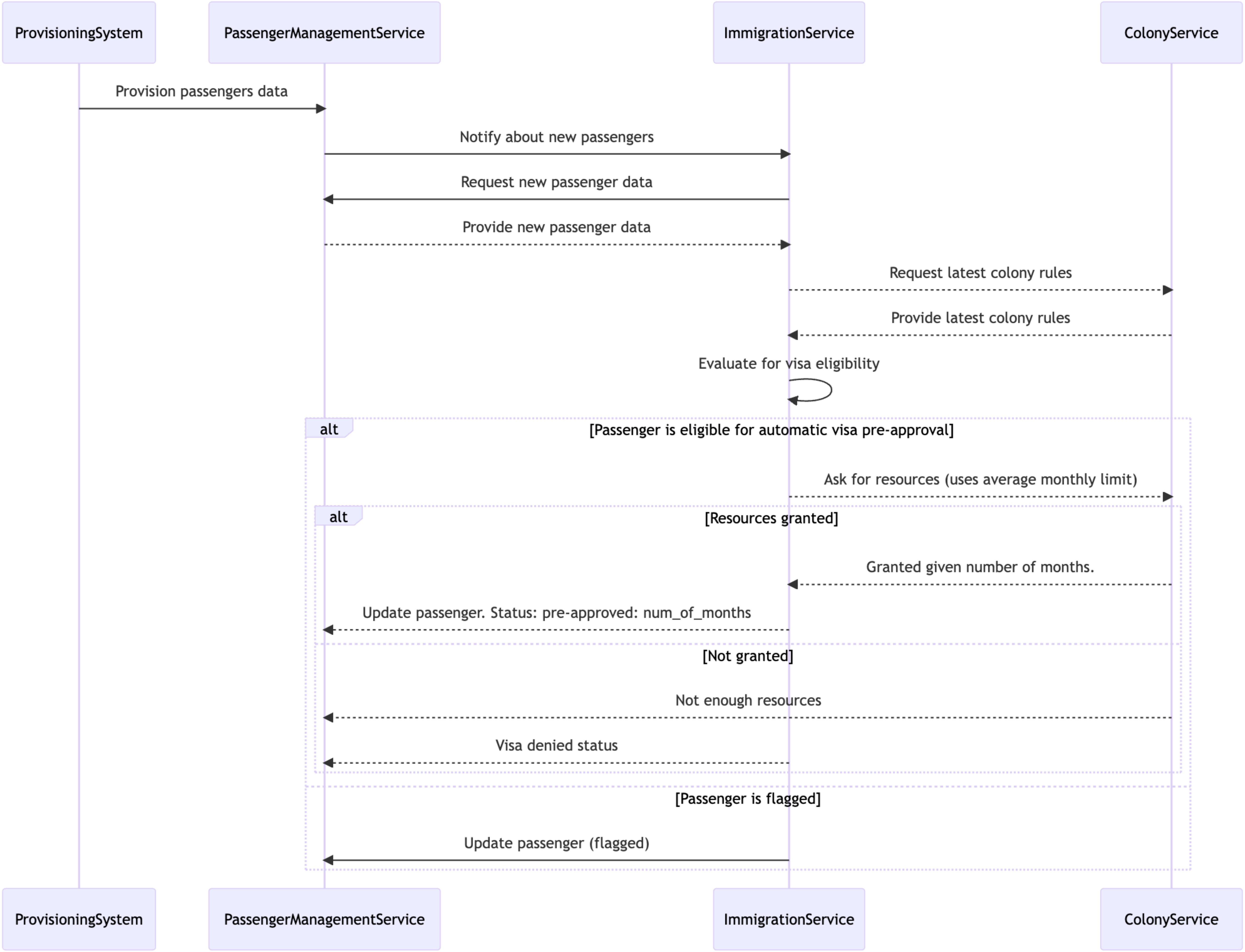
`GET /passengers` : Get a list of passengers

`GET /passengers/{id}` : Get passenger details

`PUT /passengers/{id}` : Update passenger details

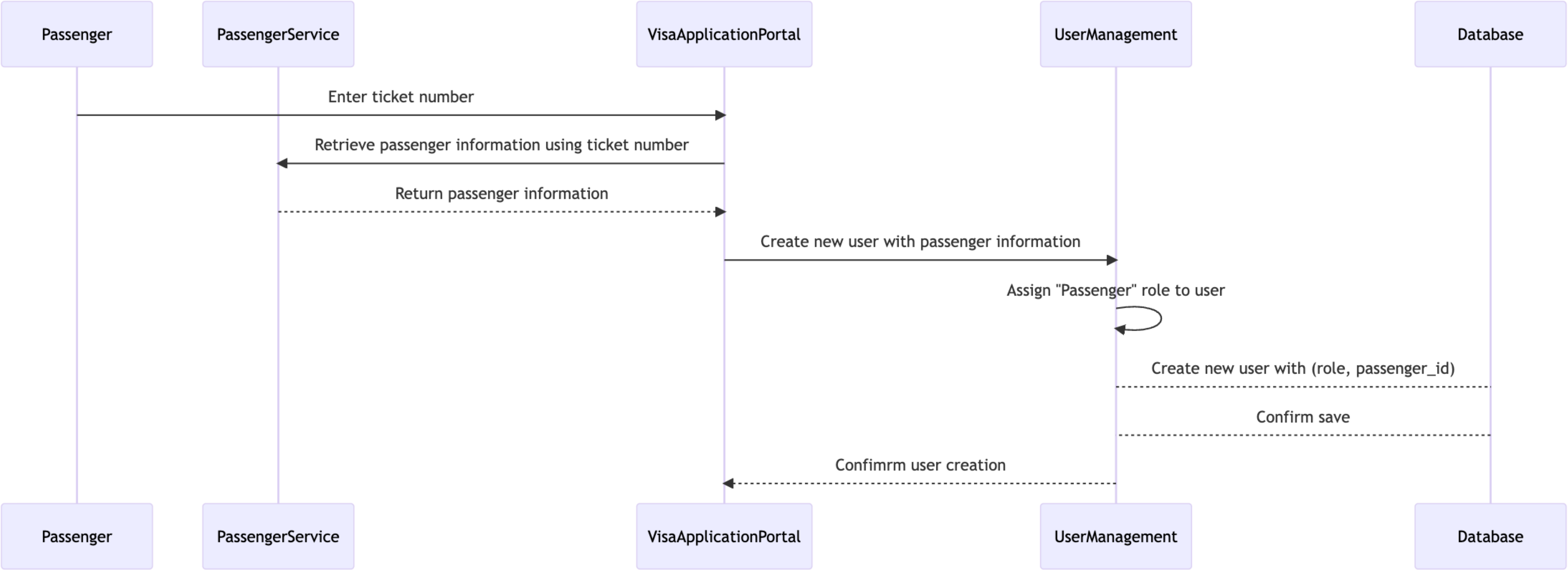
`DELETE /passengers/{id}` : Delete a passenger

Provisioning and Visa Pre-Approval

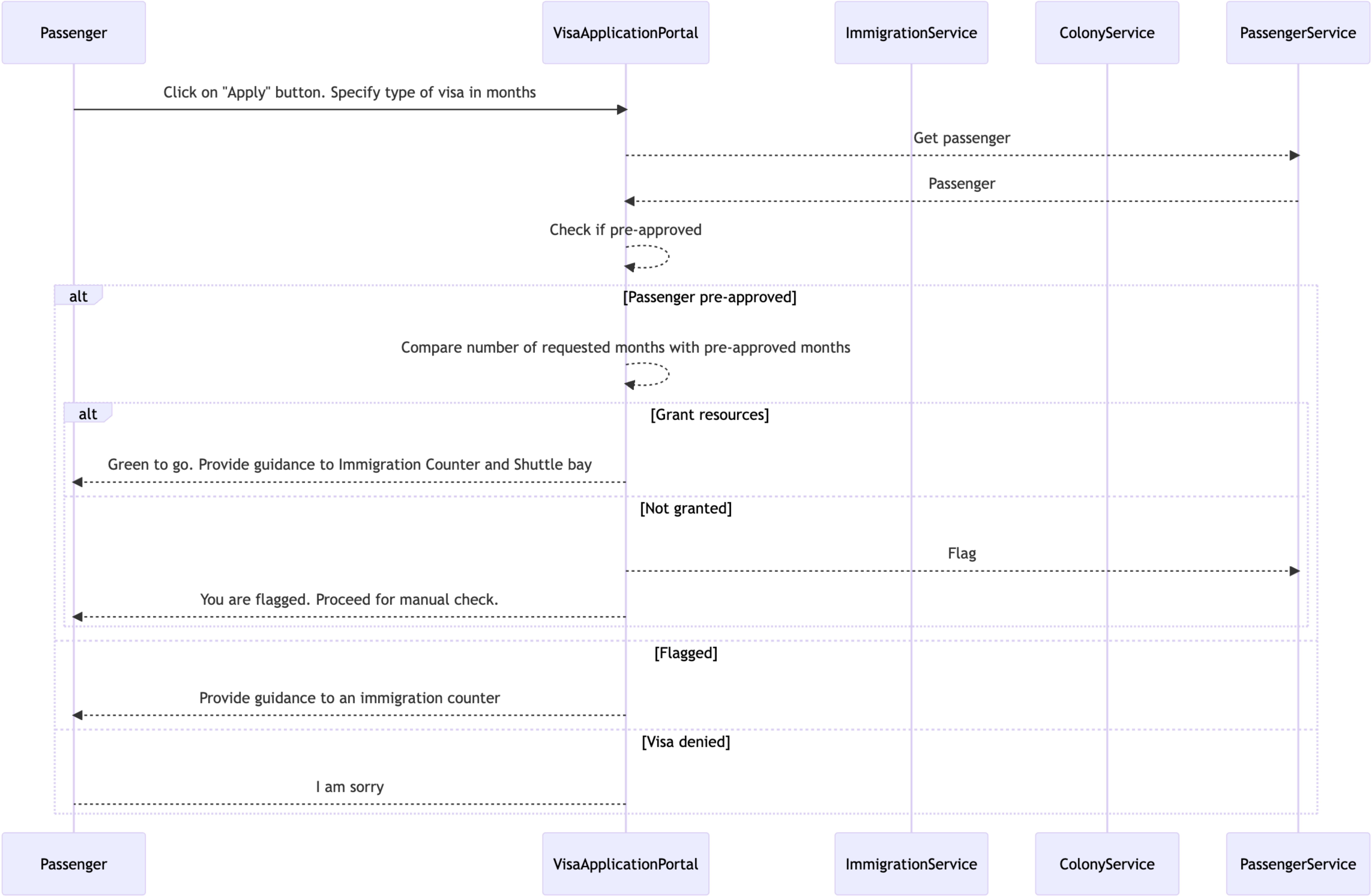


ProvisioningSystem
PassengerManagementService
ImmigrationService
ColonyService
Provision passengers data
Notify about new passengers
Request new passenger data
Provide new passenger data
Request latest colony rules
Provide latest colony rules
Evaluate for visa eligibility
alt
[Passenger is eligible for automatic visa pre-approval]
Ask for resources (uses average monthly limit)
alt
[Resources granted]
Granted given number of months.
Update passenger. Status: pre-approved: num_of_months
[Not granted]
Not enough resources
Visa denied status
[Passenger is flagged]
Update passenger (flagged)
ProvisioningSystem
PassengerManagementService
ImmigrationService
ColonyService

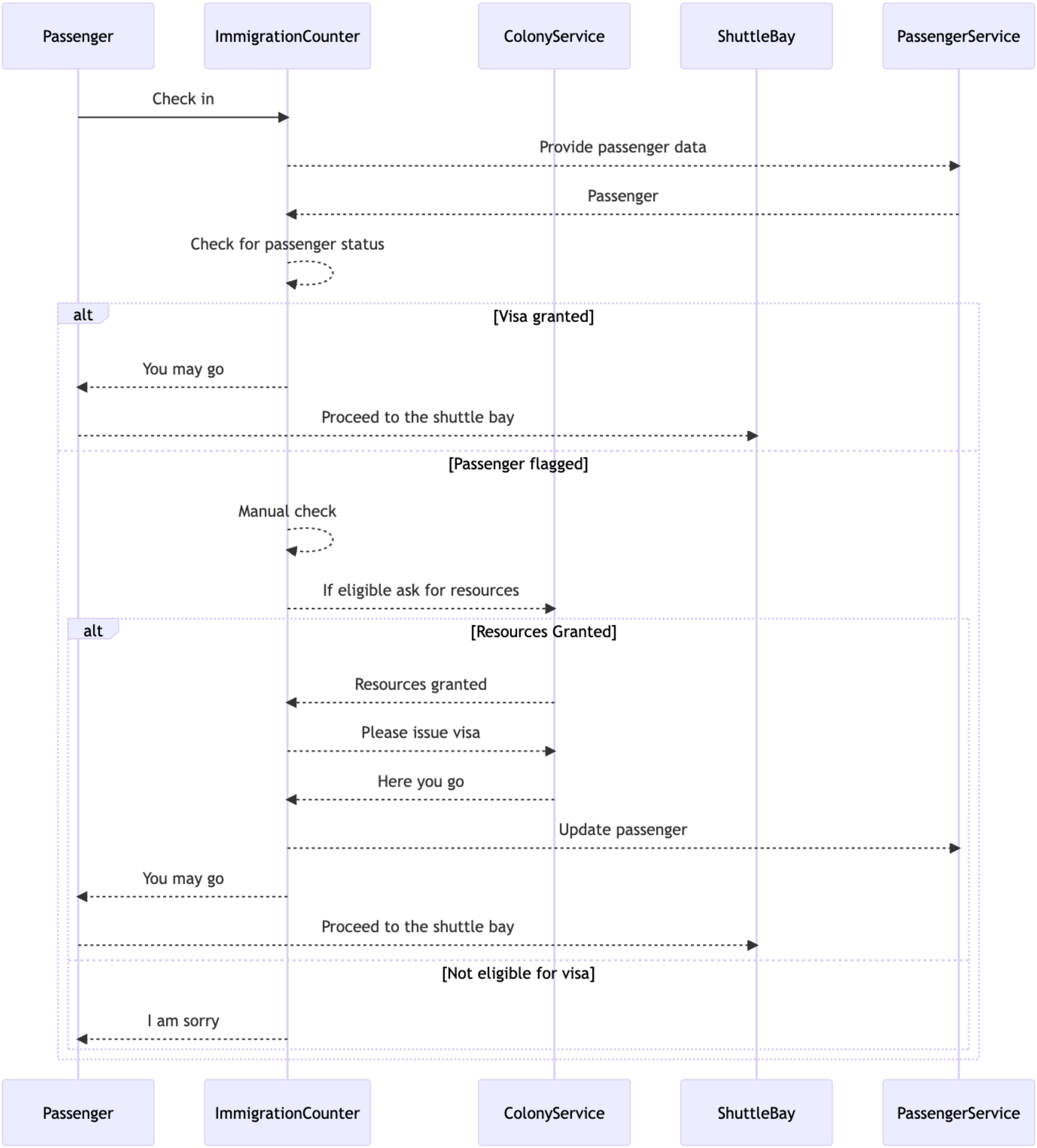
Passenger Registration


Passenger
PassengerService
VisaApplicationPortal
UserManagement
Database
Enter ticket number
Retrieve passenger information using ticket number
Return passenger information
Create new user with passenger information
Assign "Passenger" role to user
Create new user with (role, passenger_id)
Confirm save
Confimrm user creation
Passenger
PassengerService
VisaApplicationPortal
UserManagement
Database

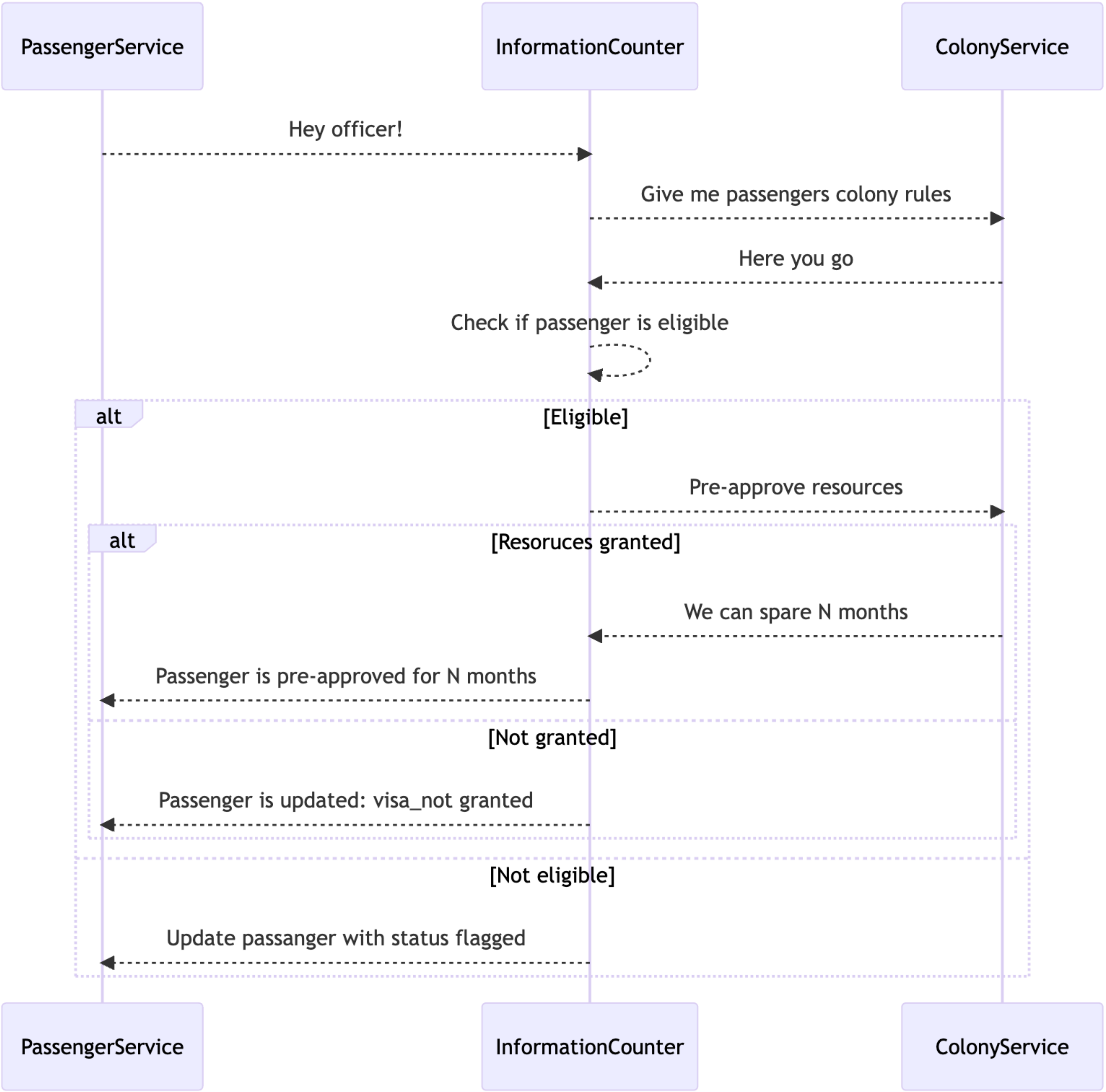
Visa Application (in portal)



Visa Application (At the Immigration Counter)



Colony rules Checking and Resource Granting



Tech Stack

- Backend Programming language: Rust/Go (NO python or C++)
- Frontend: React + typescript
- Database: CockroachDB
- Infrastructure: Local Kubernetes Cluster
- Orchestration and Deployment: Kubernetes + terraform + ansible
- Logging, monitoring, analytics: Grafana, Consul
- Security: SUSE Rancher for container security

Architecture Decisions

- serverless or kubernetes?
- db per microservice or shared db?
- RBAC?

Development Process Management

- No agile, SCRUM, daily standups, TDD …
- Common sense
- Clear communication channels. Collaboration tools (Notion)
- Code tracking tools (Wakatime)
- Time tracking (Toggle)

Cost estimation

Estimated length: 9 months

Development Team (9 months)

Technical Project Manager/Architect: 9000usd (not full time)

Backend Developer: 63000

Frontend Development (web + mobile): 65000

Devops Engineer: 57000

Infrastructure

Server Hosting: 200$/month = 1800usd

Database Hosting: 80$/month = 720

3rd party tools

300$/month = 2700

Cost estimation

Total Estimated Cost

Development Team: 221000

Infrastructure: 2520

Miscellaneous: 1000

Grand Total: 227520

Minimal Maintenance = 7000 usd/m (one engineer, servers, power etc)

DEMO