Guest Identity & Reservation Validator Logic 1
Overview 1
Data Structures 1
Datum: GuestDatum 1
Redeemer: GuestRedeemer 2
Validation Logic by Redeemer 2
1. SubmitIdentity 2
2. VerifyIdentity 3
3. ConfirmIdentity 3
4. ReserveRoom 3
5. ConfirmReservation 4
6. InitiateCheckIn 4
7. CheckOut 4
Security & Invariants 5
Smart contract low level diagram 5

Guest Identity & Reservation Validator Logic

Overview

This Plutus smart contract governs the lifecycle of a guest's identity verification and hotel room reservation, including check-in and check-out flows. It ensures data integrity across states and enforces authorization checks using PubKeyHash.

Data Structures

Datum: GuestDatum

Represents the full state of a guest's interaction:

Field	Type	Description

guestAddress	BuiltinByteString	Guest’s address
name	BuiltinByteString	Guest name
passportNumber	BuiltinByteString	Passport number
photoHash	BuiltinByteString	Hash of identity photo
isUserVerified	Bool	Admin has verified the guest
identityStatus	Bool	Identity is fully confirmed
isReserved	Bool	Room has been reserved
initiateCheckIn	Bool	Guest can opt for check-in
reservationStatus	Bool	Reservation is confirmed
reservationId	BuiltinByteString	Unique reservation identifier
roomId	BuiltinByteString	Reserved room identifier
checkInDate	BuiltinByteString	Check-in date
checkOutDate	BuiltinByteString	Check-out date
adminPKH	PubKeyHash	Admin’s public key hash

Redeemer: GuestRedeemer

Defines the action being validated:

- SubmitIdentity
- VerifyIdentity
- ConfirmIdentity
- ReserveRoom
- ConfirmReservation
- InitiateCheckIn
- CheckOut

Validation Logic by Redeemer

1. SubmitIdentity

Actor: Guest

Action: Guest submits their identity for the first time.

Preconditions:

- All Guest Datum fields must be empty or with default values as per their data types

Expected changes:

- guestAdress, name, passportNumber and photoHash are populated.
- Other fields will remain unchanged.

2. VerifyIdentity

Actor: Admin

Action: Admin verifies the guest's submitted identity.

Preconditions:

- guestAdress, name, passportNumber and photoHash must be fully submitted.
- isUserVerified == False

Expected changes:

- Transaction signed by adminPKH.
- isUserVerified == True
- Other fields remain unchanged.

3. ConfirmIdentity

Actor: Guest

Action: Confirms verified identity.

Preconditions:

- Guest must be verified as isUserVerified=True.
- identityStatus == False

Expected changes:

- Admin confirms the Identity of the Guest
- identityStatus == True

4. ReserveRoom

Actor: Guest

Action: Reserve a room.

Preconditions:

- Room info must be empty.
- isReserved == False

Expected changes:

- isReserved == True
- Reservation details will be submitted.

5. ConfirmReservation

Actor: Guest

Action: Confirm reservation with reservation ID.

Preconditions:

- Room is reserved.
- reservationId == empty

Expected changes:

- Admin (mimicking as Hotel Staff) will confirm the reservation.
- reservationId is set.
- reservationStatus == True

6. InitiateCheckIn

Actor: Guest

Action: Begin pre-check-in process.

Preconditions:

- Identity confirmed and verified.
- Room is reserved and Reservation is confirmed.
- initiateCheckIn == False

Expected changes:

- Admin will mark the Guest that he/she can opt for precheck-in.
- initiateCheckIn == True

7. CheckOut

Actor: Guest

Action: Guest checks out.

Preconditions:

- Reservation must be active and confirmed.
- Check-in must have been initiated.

Expected changes:

- Flags (isReserved, reservationStatus, initiateCheckIn) are reset.
- Reservation metadata is cleared.
- Identity info remains unchanged.

Security & Invariants

- Each transition ensures **only intended fields change**.
- VerifyIdentity is protected by **admin signature**.
- Expects **exactly one continuing output** with an inline datum.

Smart contract low level diagram

https://drive.google.com/drive/folders/1-ZhYD2nWyAH_k7zIyTv42eteB8uhiu59

*please download and open it in the browser