

Pasos para generar una reserva mediante la integración del api de Omnibees

El primer paso es buscar un hotel

```
$settings = $settings['settings']['logger_omnibees'];
$logger = new Monolog\Logger($settings['name']);
$logger->pushProcessor(new Monolog\Processor\UidProcessor());
$logger->pushHandler(new Monolog\Handler\StreamHandler($settings['path'], $settings['level']));

$omnibees = new \APIS\Omnibees($logger);

$response = $omnibees->HotelSearch();
```

Es identificado el hotel que se quiere utilizar se debe inicializar la búsqueda de fechas disponibles, con los siguientes pasos:

1. Crear el objeto cuarto, parametrizando las fechas
2. Crear los huéspedes
3. Agregar los huéspedes al cuarto
4. Configurar los datos para la búsqueda del cuarto, en caso de no insertarlos se manejarán los datos por defecto o se arrojará un error al no encontrar algún dato obligatorio
 - Rate per Market
 - Código del hotel
 - Id de tarifa
5. Se deberá obtener el array con la estructura de búsqueda y mandarlo a la api de Omnibees

```
$room = new \Modules\Reservation\Room([], "2025-12-05", "2025-12-09");

$guest1 = new \Modules\Schemas\Guest([
    "first_name" => "Ob Adult 1",
    "last_name" => "Doe",
    "email" => "john.doe@example.com",
    "phone" => "123-456-7890",
    "PrimaryIndicator" => true,
    "AgeQualifyingCode" => "Adult",
    "ResGuestRPH" => 1,
]);
$room->addGuest($guest1);

$guest2 = new \Modules\Schemas\Guest([
    "first_name" => "Ob adult 2",
    "last_name" => "Doe",
    "email" => "john.doe@example.com",
    "phone" => "123-456-7890",
    "PrimaryIndicator" => false,
    "AgeQualifyingCode" => "Adult",
    "ResGuestRPH" => 2,
]);
$room->addGuest($guest2);

$room->setISOMarket("CHE");
$room->setResortId(198336);
$room->setRatePlan(3236772);

$data = $room->availableArray();

$response = $omnibees->GetHotelAvail($data,[$room]);
```

Los parámetros de la función GetHotelAvail Permiten realizar la búsqueda de una reserva nueva tanto como realizar la búsqueda para una modificación de cuarto.

```
/**
 * Obtiene la disponibilidad de habitaciones para un hotel en un rango de fechas
 * @param string $cin Fecha de entrada
 * @param string $cout Fecha de salida
 * @param int $hotel_code Código del hotel
 * @param string $rate_plan Código del plan de tarifas
 * @param string $room_id Código de la habitación
 * @param string $ISOMarket Código del mercado ISO
 * @param int $adults Número de adultos
 * @param int $childs Número de niños
 * @return string Respuesta de la API
 */
function GetHotelAvail($data,$rooms,$isModify = false,$reservationNumber = null,$indexNumber = null)
```

1. Se carga en memoria el json de búsqueda inicial de cuartos
2. se agregan datos generales como el UID , el time stamp y la versión de la api a utilizar .
3. se agregan los datos de la reserva como el código de hotel, fechas de check in y check out, tarifa, el id del rate market,
 - en caso de existir el filtro de tarifa este será aplicado
4. se procesan cada uno de los cuartos mandados en el segundo parámetro para agregarlos en la petición de búsqueda .
5. en caso de ser una modificación se agrega el número de reservación y los datos pertinentes a la búsqueda.

```
function GetHotelAvail($data,$rooms,$isModify = false,$reservationNumber = null,$indexNumber = null){
    $string = file_get_contents(self::REQUEST_FOLDER . "HotelAvailRQ.json");

    $json_a = json_decode($string);
    $uuid = $this->generateGUID();
    // lowercase the uuid
    $uuid = strtolower($uuid);
    $json_a->EchoToken = "system_" . $uuid;
    $json_a->TimeStamp = date("c");
    $json_a->Version = "7.2";
    $json_a->HotelSearchCriteria->Criterion->HotelRefs[0]->HotelCode = intval($data["hotel_code"]);
    $json_a->HotelSearchCriteria->Criterion->StayDateRange->Start = $data["cin"] . "T00:00:00";
    $json_a->HotelSearchCriteria->Criterion->StayDateRange->End = $data["cout"] . "T00:00:00";

    if(!empty($data["rate_plan"])):
        if(!is_array($data["rate_plan"]))
            $data["rate_plan"] = [$data["rate_plan"]];
        $json_a->HotelSearchCriteria->Criterion->TPA_Extensions->RatePlanIds = $data["rate_plan"];
    endif;

    if(!empty($data["room_id"])):
        // $json_a->HotelSearchCriteria->Criterion->RoomStayCandidatesType->RoomStayCandidates[0]->RoomID = intval($room_id);
        $json_a->HotelSearchCriteria->Criterion->TPA_Extensions->RoomTypeIds = [intval($data["room_id"]);];
    endif;

    if(!empty($data["ISOMarket"]))
        $json_a->POS->Sources[0]->ISOCountry = $data["ISOMarket"];

    if($data["ISOMarket"] == "ALL" || $data["ISOMarket"] == NULL)
        $json_a->POS = NULL;

    foreach($rooms as $index => $room):
        $room_schema = $room->getSchema();
        $room_schema->RPH = $index;
        $json_a->HotelSearchCriteria->Criterion->RoomStayCandidatesType->RoomStayCandidates[] = $room_schema;
    endforeach;

    if($isModify):
        $ProfileInfos =
        [
            "UniqueID" => [
                "ID" => $reservationNumber,
                "Type" => 14
            ]
        ];
        $json_a->HotelSearchCriteria->Criterion->ProfilesType = new stdClass();
        $json_a->HotelSearchCriteria->Criterion->ProfilesType->ProfileInfos = [$ProfileInfos];
        $json_a->HotelSearchCriteria->Criterion->RoomStayCandidatesType->RoomStayCandidates[0]->BookingCode = $indexNumber;
        $json_a->IsModify = true;
    endif;

    $body = json_encode($json_a);
```

- 6 Una vez concluido el proceso de configuración de los datos para la búsqueda se procede a hacer la codificación de los mismos al formato json .
- 7 son enviados los datos asegurándose de almacenar un que contiene la información enviada como la información recibida y detalles relevantes para hacer depuración y corrección de errores

```
$body = json_encode($json_a);

$request = new Request('POST', $this->apiUrl . '/api/Pull/GetHotelAvail', $this->headers, $body);
$res = $this->client->sendAsync($request)->wait();
$content = $res->getBody()->getContents();
// Log the request and response
$this->logger->info("Petition",
[
    "incoming"=> [
        "function" => "GetHotelAvail",
        "Method" => "direct",
        "api" => "",
        "Route" => "/api/Pull/GetHotelAvail",
        "Data" => json_decode($body,true),
    ],
    "response" =>[
        "status" => $res->getStatusCode(),
        "type" => $res->getHeaderLine('Content-Type'),
        "body" => json_decode($content,true)
    ]
]);

return $content;
```

- Una vez que el proceso de búsqueda de cuartos ha sido exitoso se procede a llamar el método que devuelven los datos para iniciar el proceso de reserva.
- se mandan los cuartos junto al dato del market source y los huéspedes
- Datos extra como el tipo de pago se mandan por separado, en un objeto que contiene estas configuraciones especiales de la reserva

```
$room->setISOMarket("CHE");
$room->setResortId(198336);
$room->setRatePlan(3236772);
$room->setRoomId(1036387);
$ISOMarket = "AUS";
$rooms = [
    $room->reservationStartArray()
];

/*
a) DirectBill = 2
b) CreditCard = 5
c) Voucher = 3
d) PrePay = 4
*/

$response = $omnibees->SendHotelResInitiate($rooms, $ISOMarket,[$guest1, $guest2],false,[
    "PaymentType" => 3
]);
```

La función SendHotelResInitiate usa la información de los cuartos huéspedes código de mercado para rellenar cada uno de los datos necesarios dentro de la plantilla json; se realiza se realiza en validaciones de datos obligatorios para continuar con el proceso de reserva. Una vez pasadas estas validaciones se inyecta la información de cada uno de los cuartos a cotizar dentro del objeto.

```
function SendHotelResInitiate($rooms, $ISOMarket = "MEX", $guests, $isModify = false, $reservation_data = [])
{
    if(empty($rooms) || !is_array($rooms))
        throw new \InvalidArgumentException("Invalid rooms data");

    if(empty($guests) || !is_array($guests))
        throw new \InvalidArgumentException("Invalid guests data");

    $string = file_get_contents(self::REQUEST_FOLDER . "SendHotelRes_Initiate_RQ.json");
    $json_a = json_decode($string);
    $uuid = $this->generateGUID();
    // lowercase the uuid
    $uuid = strtolower($uuid);
    $json_a->EchoToken = "system_" . $uuid;
    $json_a->TimeStamp = date("c");

    foreach($guests as $index => $guest):
        $guest->setRPH($index + 1);
        $json_a->HotelReservationsType->HotelReservations[0]->ResGuestsType->ResGuests[] = $guest->getSchema();
    endforeach;

    foreach($rooms as $index => $room):
        if(empty($room['RoomID'])) throw new \Exception("RoomID is required for room ($index)");
        if(empty($room['GuestCounts'])) throw new \Exception("GuestCounts are required for room ($index)");
        if(empty($room['StartDate'])) throw new \Exception("StartDate is required for room ($index)");
        if(empty($room['EndDate'])) throw new \Exception("EndDate is required for room ($index)");
        if(empty($room['HotelCode'])) throw new \Exception("HotelCode is required for room ($index)");
        if(empty($room['RatePlanID'])) throw new \Exception("RatePlanID is required for room ($index)");

        $cin = $room['StartDate'];
        $cout = $room['EndDate'];
        $hotel_code = $room['HotelCode'];
        $rate_plan_id = $room['RatePlanID'];
        $room_id = $room['RoomID'];

        // add a room stay
        $basic_room_stay = json_decode(file_get_contents(self::SCHEMAS_FOLDER . "RoomStay.json"));
        $json_a->HotelReservationsType->HotelReservations[0]->RoomStaysType->RoomStays[] = $basic_room_stay;

        $json_a->HotelReservationsType->HotelReservations[0]->RoomStaysType->RoomStays[0]->CommentsType->Comments = [];

        // add a comment
        $json_a->HotelReservationsType->HotelReservations[0]->RoomStaysType->RoomStays[0]->CommentsType->Comments[] = (object)[
            "Description" => "This is my ($index) room comment",
            "Language" => null,
            "Name" => null
        ];

        $json_a->HotelReservationsType->HotelReservations[0]->RoomStaysType->RoomStays[$index]->BasicPropertyInfo->HotelRef->HotelCode = intval($hotel_code);
        $json_a->HotelReservationsType->HotelReservations[0]->RoomStaysType->RoomStays[$index]->RoomRates[0]->RatePlanID = intval($rate_plan_id);
        $json_a->HotelReservationsType->HotelReservations[0]->RoomStaysType->RoomStays[$index]->RoomRates[0]->RoomID = intval($room_id);
        $json_a->HotelReservationsType->HotelReservations[0]->RoomStaysType->RoomStays[$index]->RoomRates[0]->EffectiveDate = $cin . "T00:00:00";
        $json_a->HotelReservationsType->HotelReservations[0]->RoomStaysType->RoomStays[$index]->RoomRates[0]->ExpireDate = $cout . "T00:00:00";
        $json_a->HotelReservationsType->HotelReservations[0]->RoomStaysType->RoomStays[$index]->GuestCountsType->GuestCounts = $room['GuestCounts'];
    endforeach;

    if($isModify):

```

En caso de ser una modificación se agregan los datos necesarios para la búsqueda

```
if($isModify):
    $ProfileInfos =
    [
        "UniqueID" => [
            "ID" => $reservation_data["reservationNumber"],
            "Type" => 14
        ]
    ];

    $json_a->HotelReservationsType->HotelReservations[0]->ProfilesType = new stdClass();
    $json_a->HotelReservationsType->HotelReservations[0]->ProfilesType->ProfileInfos = [$ProfileInfos];
    $json_a->HotelReservationsType->HotelReservations[0]->RoomStayCandidatesType->RoomStayCandidates[0]->BookingCode = $reservation_data["IndexNumber"];
endif;
```

Se agregan los datos faltantes para completar la petición y se validan algunas configuraciones extra de la reserva como el método de pago; posterior a esto se procede a hacer la petición y guardará los correspondiente

```
$json_a->HotelReservationsType->HotelReservations[0]->ResGlobalInfo->Guarantee->GuaranteesAcceptedType->GuaranteesAccepted[0]->GuaranteeTypeCode = 2; // credit card as default
if(!empty($reservation_data["PaymentType"]))
    $json_a->HotelReservationsType->HotelReservations[0]->ResGlobalInfo->Guarantee->GuaranteesAcceptedType->GuaranteesAccepted[0]->GuaranteeTypeCode = $reservation_data["PaymentType"];

$json_a->TransactionIdentifier = $uuid;
$json_a->Version = "7.2";
$body = json_encode($json_a);

$request = new Request('POST', $this->apiUrl . '/api/Pull/SendHotelRes', $this->headers, $body);
$res = $this->client->sendAsync($request)->wait();
$content = $res->getBody()->getContents();
// Log the request and response
$this->logger->info("Petition",
[
    "incoming"=> [
        "function" => "SendHotelRes",
        "method" => "direct",
        "api" => "",
        "route" => "/api/Pull/SendHotelRes",
        "data" => json_decode($body,true),
    ],
    "response" =>[
        "status" => $res->getStatusCode(),
        "type" => $res->getHeaderLine('Content-Type'),
        "body" => json_decode($content,true)
    ]
]);
return $content;
}
```

Una vez que se ha mandado la petición a la reserva y se confirma que se ha generado de manera exitosa se procede a hacer el commit, Usando los datos del código de reserva y el id de la transacción

```
$reservationId = "RES000481-198336";
$transactionId = "ab258d87-4923-3b20-938a-22dc87a7b8c8";

$response = $omnibeas->SendHotelResCommit($reservationId, $transactionId);
// ----- termina la reserva original -----
```

Dentro de la función que manda el cómic se anexan los datos del código de reserva y la transacción.

```
function SendHotelResCommit($reservation_id, $transaction_id){
    $string = file_get_contents(self::REQUEST_FOLDER . 'SendHotelRes_Commit_RQ.json');
    $json_a = json_decode($string);
    $uuid = $this->generateGUID();
    $uuid = strtolower($uuid);

    // Set the EchoToken, TimeStamp, TransactionIdentifier and UniqueID
    $json_a->EchoToken = "system." . $uuid;
    $json_a->TimeStamp = date("c");
    $json_a->TransactionIdentifier = $transaction_id;
    $json_a->HotelReservationsType->HotelReservations[0]->UniqueID->ID = $reservation_id;

    $body = json_encode($json_a);
    $request = new Request('POST', $this->apiUrl . '/api/Pull/SendHotelRes', $this->headers, $body);
    $res = $this->client->sendAsync($request)->wait();
    $content = $res->getBody()->getContents();
    // Log the request and response
    $this->logger->info("Petition",
    [
        "incoming"=> [
            "function" => "SendHotelResCommit",
            "method" => "direct",
            "api" => "",
            "route" => "/api/Pull/SendHotelResCommit",
            "data" => json_decode($body,true),
        ],
        "response" =>[
            "status" => $res->getStatusCode(),
            "type" => $res->getHeaderLine('Content-Type'),
            "body" => json_decode($content,true)
        ]
    ]);
    return $content;
}
```