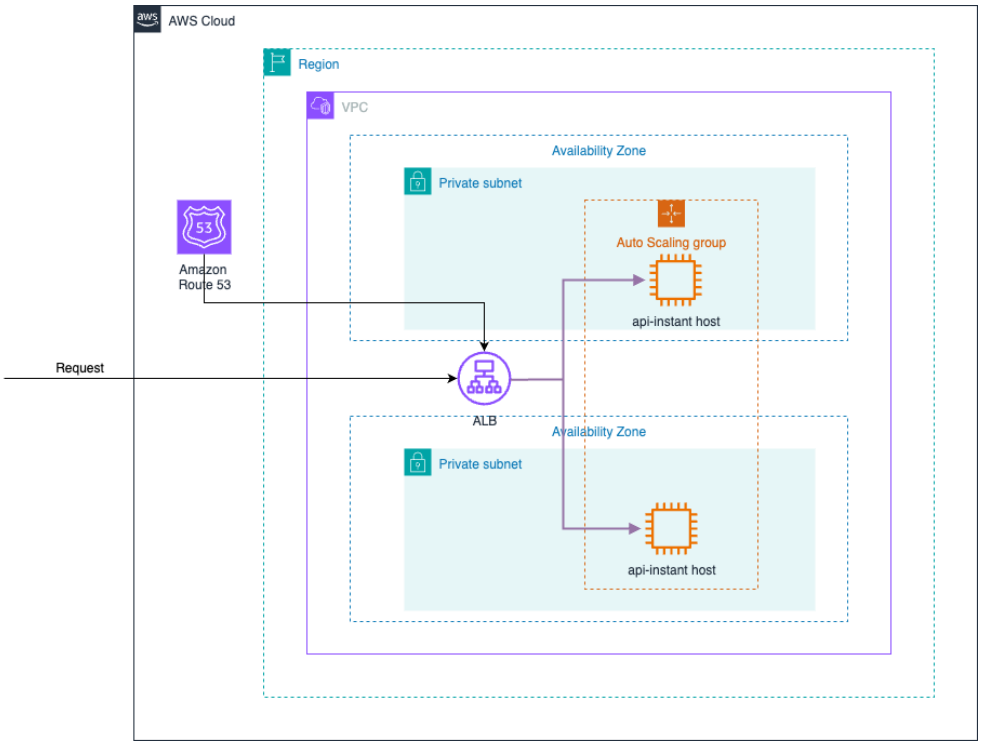
Getting Started with Oz Liveness

Following the instructions below, you'll create your EC2 instance on AWS Marketplace, launch Oz Liveness API, and then will be able to perform your first Liveness and Biometry analyses.

Configuring Your EC2 Instance in AWS Console

To launch Oz Liveness from AWS, you'll need to create and configure the corresponding instance. Subscribe to OZ Forensics product and move on to the guide.



Step 1. Create Security Groups

First, create the required security groups.

Application Load Balancer Security Group

1. Go to **EC2 → Security Groups → Create Security Group**.
2. Name it: **alb-sg**.
3. Select your VPC.
4. Add **Inbound Rule**:
   - Type: HTTPS (or HTTP if you do not want to use a secure connection).
   - Port: 443 for HTTPS (or 80 for HTTP).
   - Source: 0.0.0.0/0 and ::/0 (or use your preferred IP range).
   - Description: "Allow HTTPS traffic" (or "Allow HTTP traffic").
5. **Outbound Rule** should stay "All traffic → 0.0.0.0/0".
6. Save the security group.

Application (Instance) Security Group

1. Create another Security Group and name it **app-sg**.
2. Select the same VPC.
3. Add **Inbound Rule for health checks**:
    - Type: Custom TCP.
    - port: 8501.
    - Source: **alb-sg** (choose the security group you've created above).
    - Description: "Allow traffic from ALB for health checks".
4. Add **Inbound Rule for request for analysis**:
    - Type: HTTP.
    - Port: 80.
    - Source: **alb-sg** (choose the security group you've created above).
    - Description: "Allow traffic from ALB for request for analysis".
5. **Outbound Rule** should stay "All traffic → 0.0.0.0/0".
6. Save the security group.

Step 2. Create IAM Role and Instance Profile for Your EC2 Instances

You also require a policy and an IAM role.

Creating a Policy

1. Go to **IAM → Policies → Create policy**.
2. Choose **JSON** and copy-paste this:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "aws-marketplace:MeterUsage",
        "aws-marketplace:BatchMeterUsage",
        "aws-marketplace:GetEntitlements"
      ],
      "Resource": "*"
    }
  ]
}
```

3. Click **Next** and give the policy a name: **OZMarketplacePolicy**.
4. Click **Create policy**.

Creating an IAM Role

1. Go to **IAM** → **Roles** → **Create role**.
2. Choose trusted entity type: **AWS service**.
3. Choose the **EC2** use case and click **Next.**
4. Attach the permissions policy: **OZMarketplacePolicy** (the name of policy you've just created).
5. Click **Create Role.**
6. On the **OZMarketplace** role page, look for **Instance profile ARN**. When you're done with the next step, **IAM instance profile** and **Instance profile ARN** must be similar.

Step 3. Create Launch Template

1. Go to **EC2** → **Launch Templates** → **Create launch template**.
2. Name the template: **app-launch-template**.
3. Under **AMI**, choose your existing `api-instant` image.
4. Select instance type: c7i-flex.4xlarge.
5. Choose the key pair if you want SSH access (optional).
6. Network settings:
    - Don't attach to a subnet here (ASG will handle that).
    - Select **app-sg** as Security Group.
7. Storage: 50 GB.
8. IOPS: 12000.
9. Throughput: 500.
10. Go to **Advanced details** → **IAM instance profile**: select **OZMarketplace** (the instance profile from Step 2).
11. Click **Create launch template**.

Step 4. Create Auto Scaling Group + Application Load Balancer

1. Go to **EC2** → **Auto Scaling Groups** → **Create Auto Scaling group**.
2. Name the group: **app-asg**.
3. Network settings:
    - VPC & subnets (ASG): pick your VPC and private subnets only.

4. Load balancing:
    - Select **Attach to a new load balancer**.
    - Load balancer type: **Application Load Balancer**.
    - ALB name: **app-alb**.
    - Load balancer scheme: **Internal (preferred)** or **Internet-facing** if you need access from the Internet.
    - ALB subnets: for **Internal** scheme, select your private subnets (at least two AZs). For **Internet-facing,** select public subnets.
5. Listeners and routing:
    - If you require secure listeners, you can configure them from the **Load Balancing** console after your load balancer is created.
6. Target group: **Create new target group** → name it **app-tg**.
   Please note: here, you **cannot** change port/health check, so we'll do it later at Step 5.
7. Health checks: switch on **Elastic Load Balancing** health checks.
8. Group size: **Desired 2**, **Min 2**, **Max 91**.
9. Scaling policies:
    - **Target tracking scaling policy** → **Average CPU utilization** → **Target** = 30.
    - Instance warm-up: 120s.
10. Click **Create Auto Scaling group**.

Step 5 (CRITICAL). Update the Target Group Created by Wizard

The goal here is to make the TG check HTTP on port 8501 and treat 404 as healthy.

1. Go to **EC2** → **Target Groups** → select **app-tg** (the one created by the wizard).
2. Attributes: **Least outstanding requests**.
3. Proceed to **Health checks** → **Edit**:
    - Protocol: **HTTP**.
    - Path: **/**
    - Port: **8501**.
    - Success codes: **404.**
4. Save the group.

Step 6. Verify Health and Access.

1. Go to **Target Groups** → **app-tg** → **Targets**: wait for status **healthy** (ALB must receive **404** from GET / on **:8501**).
2. Go to Load Balancers → app-alb: copy DNS name; `test https://<alb-dns-name>` (or `http://` to verify redirect if kept).

Step 7. Create Alias Record to the ALB in Route 53

1. Go to Route 53 → Hosted zones → your zone → Create record.
2. Record name: app (for app.example.com).
3. Type: CNAME.
4. Alias: Yes, then choose your ALB (app-alb).
5. Click Create.
6. Test the alias: `https://app.example.com/api/version`. In response, you will receive the API version in JSON.

Troubleshooting Tips

Problem	What to Check
Unhealthy targets	• If service listens on 0.0.0.0:8501; app-sg allows 8501 from alb-sg (source must be the SG ID). • That healthcheck path returns 404 (by design)
ACM certificate is not listed	• Certificate must be ISSUED in the same region as ALB
Scale-out is not triggering	• Allow a few minutes under load: remember it's ASG average CPU 30% with warm-up
Role not present on instances	Go to Launch Template → Details → IAM instance profile and ensure that: • the name is OZMarketplace. • ASG uses the latest LT version. • On a running instance, EC2 → Instance → IAM role should show OZMarketplace.
You require more c7i-flex.4xlarge capacity	• Add more AZs or consider fallback types (mixed instances) later.

Validating Your Client's Liveness

To validate your client's Liveness using the combination of Web SDK and Instant API, please follow this guide.

Step 1: Getting Data from Web SDK

When Web SDK finishes capturing video, in the on_capture_complete callback, it sends the following data (for more details, please refer to [Capturing Video and Description of the on_capture_complete Callback | Oz Knowledge]):

```
{
  "data_container": null,
  "frame_list": [
    "...",
    "...",
    ...
    "..."
  ],
  "action": "video_selfie_blank",
  "additional_info": "MDA...",
  "meta": {
    ...
  }
  ...
}
```

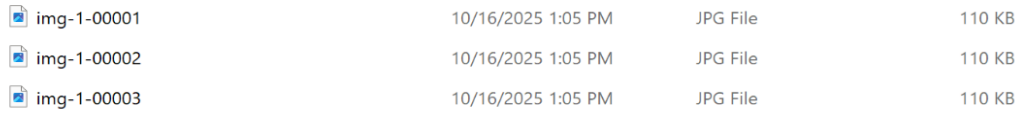
Web SDK `on_capture_complete` Callback → API Request Payload Mapping

From `on_capture_complete(arg)`, take these data points and send them to your backend for further transmission to Oz API:

```
on_capture_complete(arg)
 ├
 ├── arg.frame_list[1..N] → package into ZIP → upload as media_key1
(application/zip)
 ├── arg.action           → media:tags.media_key1[0]
 └── arg.meta { }         → folder:meta_data { same keys }
```

Step 2: Preparing Files on Your Backend

On your backend:
Save files from `arg.frame_list[1..N]` under names `img-1-0000N.jpeg` as shown in the screen below, zip it, and save as media_key1.zip



Step 3: Calling Oz API

To check client's Liveness using Instant API, call the endpoint provided by Oz team: POST `{{host}}/api/instant/folders/` (default: `POST https://host:port/api/instant/folders/`).
In your API request, send the media along with folder and media metadata, and action as tags: please check the example. Use the "multipart / form-data" content type for your HTTP request. When working in the Oz Sandbox environment:

1. **Pass your `ACCESS_TOKEN` in the X-Forensic-Access-Token header.**
2. **Add media files (media_key1.zip) to the request body. In the payload part, define the tags and metadata. Use the "multipart / form-data" content type for your HTTP request.**

When you deploy your own API Instant, no tokens will be needed as the service will be in your environment.

```
Curl request example
curl -X POST "{{host}}/api/instant/folders/" \
  -H "X-Forensic-Access-Token: ACCESS_TOKEN" \
  -F "media_key1=@media_key1.zip" \
  -F "payload=<payload.json;type=application/json"
```

Example of your request's payload.json

```
{
    // meta_data filled based on meta from Web SDK
on_capture_complete.arg.meta
    "folder:meta_data": {
      ...
    },
    // additional info sets based on additional_info from Web SDK
on_capture_complete.additional_info
    "media:meta_data": {
    },
    // tag sets based on action obtained from Web SDK
on_capture_complete.action
    "media:tags": {
        "media_key1": [
            "video_selfie_blank"
        ]
    },
    // please do not change this part
    "analyses": [
        {
            "type": "quality",
            "params": {
               "extract_best_shot":true
            }
        }
    ]
}
```

Step 4: Obtaining the Response

Response Example


```
{
    "company_id": "some_id",
    "time_created": 1760619006.595358,
    "folder_id": "some_id",
    "user_id": "some_id",
    "resolution_endpoint": null,
    "resolution_status": "FINISHED",
    "resolution_comment": "[]",
    "system_resolution": "SUCCESS",
    "resolution_time": 1760619008.801201,
    "resolution_author_id": null,
    "resolution_state_hash": "8521fc82c41d1a02",
    "operator_comment": null,
    "operator_status": null,
    "is_cleared": false,
    "meta_data": {
        ...
    },
    "media": [
        {
            "company_id": "some_id",
            "folder_id": "some_id",
            "folder_time_created": 1760619006.595358,
            "original_name": "media_key1.zip",
            "original_url": null,
            "media_id": "some_id",
            "media_type": "SHOTS_SET_FOLDER",
            "tags": [
                "video_selfie_blank"
            ],
            "info": {
                "original": {
                    ...
                    }
                },
                "preview": {
                    ...
                }
            },
            "time_created": 1760619006.596231,
            "time_updated": 1760619006.596232,
            "meta_data": {},
            "thumb_url": null,
            "preview_url": null,
            "preview_origin_url": null,
            "shots_set_id": "some_id",
            "video_url": null
        }
    ],
    "time_updated": 1760619006.595362,
    "analyses": [
        {
```

```
            "company_id": "some_id",
            "group_id": "some_id",
            "folder_id": "some_id",
            "folder_time_created": 1760619006.595358,
            "analysis_id": "some_id",
            "state": "FINISHED",
            "resolution_operator": null,
            "results_media": [
                {
                    "company_id": "some_id",
                    "folder_time_created": 1760619006.595358,
                    "media_association_id": 0,
                    "media_association_type": "SHOTS_SET",
                    "analysis_id": "some_id",
                    "results_data": {
                       ...
                    },
                    "source_media_id": "some_id",
                    "output_images": [
                        {
                            "company_id": "some_id",
                            "folder_id": "some_id",
                            "folder_time_created": 1760619006.595358,
                            "original_name": "ResultImage.jpeg",
                            "original_url": null,
                            "media_id": "some_id",
                            "media_type": "IMAGE_RESULT_ANALYSE_SINGLE",
                            "tags": [],
                            "info": {
                                "original": {
                                    ...
                                },
                                "thumb": {
                                    ...
                                }
                            },
                            "time_created": 1760619008.791088,
                            "time_updated": 1760619008.791091,
                            "meta_data": {},
                            "media_association_id": "some_id",
                            "thumb_url": null,
                          // best shot
                            "image_b64": "/9j...=="
                        }
                    ],
                    "collection_persons": [],
                    "analysis_id": "some_id",
                    "source_shots_set_id": "some_id"
                }
            ],
            "results_data": null,
            "confs": {
                ...
                "extract_best_shot": true,
                ...
            },
```

```
            "meta_data": null,
            "time_created": 1760619006.701419,
            "time_updated": 1760619008.801042,
            "error_code": null,
            "error_message": null,
            "source_media": [
                {
                    "company_id": "some_id",
                    "folder_id": "some_id",
                    "folder_time_created": 1760619006.595358,
                    "original_name": "media_key1.zip",
                    "original_url": null,
                    "media_id": "some_id",
                    "media_type": "SHOTS_SET_FOLDER",
                    "tags": [
                        "video_selfie_blank"
                    ],
                    "info": {
                        "original": {
                            ...
                            }
                        },
                        "preview": {
                            ...
                        }
                    },
                    "time_created": 1760619006.596231,
                    "time_updated": 1760619006.596232,
                    "meta_data": {},
                    "thumb_url": null,
                    "preview_url": null,
                    "preview_origin_url": null,
                    "shots_set_id": "some_id",
                    "video_url": null
                }
            ],
            "type": "QUALITY",
            "analysis_id": "some_id",
            "resolution_status": "SUCCESS",
            "resolution": "SUCCESS"
        }
    ]
}
```


In response, you receive analysis results.

```
$.resolution_status = "FINISHED" => The analysis is completed without technical errors, otherwise, the status would be FAILED
$.system_resolution = "SUCCESS" => The analysis confirms that the user is a real person
$.analyses[0].results_media[0].output_images[0].image_b64 => Take best shot from here
```

Comparing Your Client's Face with Reference Photo

In this section, we cover the Biometry functionality of our product.

Step 1: Obtaining the Best Shot

For the Biometry analysis, you require two media: the one from the Liveness analysis and your reference photo that will be used for comparison.

1. As the first media, use the best shot from the Liveness video you've received from Web SDK. Find it in your Liveness analysis response: `$.analyses[0].results_media[0].output_images[0].image_b64`. Save this media on your backend as best_shot.png.
2. As the second media, use the reference photo from your database (reference_photo.png).

Step 2: Calling Oz API

This step is pretty much the same as Step 3 in the Liveness guide. Once more, call the endpoint provided by Oz team:
`POST https://{{host}}/api/instant/folders/` (default: `POST https://host:port/api/instant/folders/`).
In your API request, send both media from Step 1 along with folder and media metadata. Use the "multipart / form-data" content type for your HTTP request.
When working in the Oz Sandbox environment:

1. Pass your `ACCESS_TOKEN` in the X-Forensic-Access-Token header.
2. Add media files (best_shot.png, reference_photo.png) to the request body. Use the "multipart / form-data" content type for your HTTP request.

When you deploy your own API Instant, no tokens will be needed as the service will be in your environment.

Request example

```
curl --location {{host}}/api/instant/folders' \
--header 'X-Forensic-Access-Token: <...your token...>' \
 -F "payload=<payload.json;type=application/json" \
 -F 'media_key1=@"/C:/Users/admin/Downloads/reference_photo.png"' \
 -F 'media_key2=@"/C:/Users/admin/Downloads/best_shot.png"'
```

Payload example

```
{
    // Take metadata from the request at Liveness Step 1
    "folder:meta_data":
    {
        ...
    },
    // Media should have the photo_selfie tag, please don't change it
    "media:tags":
    {
        "media_key1":
        [
            "photo_selfie"
        ],
        "media_key2":
        [
            "photo_selfie"
        ]
    },
    // To start the biometry analysis, define this analysis type; please
don't change it
    "analyses":
    [
        {
            "type": "biometry"
        }
    ]
}
```

Step 3: Obtaining the Response

Response Example

```
{
    "company_id": "some_id",
    "time_created": 1761031753.553993,
    "folder_id": "some_id",
    "user_id": "some_id",
    "resolution_endpoint": null,
    "resolution_status": "FINISHED",
    "resolution_comment": "[]",
    "system_resolution": "SUCCESS",
    "resolution_time": 1761031754.157704,
    "resolution_author_id": null,
    "resolution_state_hash": "4e577c04d3010519",
    "operator_comment": null,
    "operator_status": null,
    "is_cleared": false,
    "meta_data": {
        ...
    },
    "media": [
        {
            "company_id": "some_id",
```

```
            "folder_id": "some_id",
            "folder_time_created": 1761031753.553993,
            "original_name": "reference_photo.png",
            "original_url": null,
            "media_id": "some_id",
            "media_type": "IMAGE_FOLDER",
            "tags": [
                "photo_selfie"
            ],
            "info": {
                ...
            },
            "time_created": 1761031753.554501,
            "time_updated": 1761031753.554503,
            "meta_data": {},
            "thumb_url": null,
            "image_id": "some_id"
        },
        {
            "company_id": "some_id",
            "folder_id": "some_id",
            "folder_time_created": 1761031753.553993,
            "original_name": "best_shot.png",
            "original_url": null,
            "media_id": "some_id",
            "media_type": "IMAGE_FOLDER",
            "tags": [
                "photo_selfie"
            ],
            "info": {
                "original": {
                   ...
                },
                "thumb": {
                   ...
                }
            },
            "time_created": 1761031753.554867,
            "time_updated": 1761031753.554868,
            "meta_data": {},
            "thumb_url": null,
            "image_id": "some_id"
        }
    ],
    "time_updated": 1761031753.553997,
    "analyses": [
        {
            "company_id": "some_id",
            "group_id": "some_id",
            "folder_id": "some_id",
            "folder_time_created": 1761031753.553993,
            "analysis_id": "some_id",
            "state": "FINISHED",
            "resolution_operator": null,
            "results_media": [
                {
```

```
                "company_id": "some_id",
                "folder_time_created": 1761031753.553993,
                "media_association_id": 0,
                "media_association_type": "IMAGE",
                "analysis_id": "some_id",
                "results_data": null,
                "source_media_id": "some_id",
                "output_images": [
                    {
                      ...
                        }
                ],
                "collection_persons": [],
                "analysis_id": "some_id",
                "source_image_id": "some_id"
            },
            {
                "company_id": "some_id",
                "folder_time_created": 1761031753.553993,
                "media_association_id": 0,
                "media_association_type": "IMAGE",
                "analysis_id": "some_id",
                "results_data": null,
                "source_media_id": "some_id",
                "output_images": [
                    {
                       ...
                        },
                        "time_created": 1761031753.855716,
                        "time_updated": 1761031753.855717,
                        "meta_data": {},
                        "media_association_id": "some_id",
                        "thumb_url": null,
                        "image_b64": "/9j/4AAQSkZJR..."
                    }
                ],
                "collection_persons": [],
                "analysis_id": "some_id",
                "source_image_id": "some_id"
            }
        ],
        "results_data": {
            "min_confidence": 1.0,
            "max_confidence": 1.0
        },
        "confs": {
            ...
        },
        "meta_data": null,
        "time_created": 1761031753.660103,
        "time_updated": 1761031754.15722,
        "error_code": null,
        "error_message": null,
        "source_media": [
            {
               ...
```

```
                },
                {
                    "company_id": "some_id",
                    "folder_id": "some_id",
                    "folder_time_created": 1761031753.553993,
                    "original_name": "best_shot.png",
                    "original_url": null,
                    "media_id": "some_id",
                    "media_type": "IMAGE_FOLDER",
                    "tags": [
                        "photo_selfie"
                    ],
                    "info": {
                      ...
                    },
                    "time_created": 1761031753.554867,
                    "time_updated": 1761031753.554868,
                    "meta_data": {},
                    "thumb_url": null,
                    "image_id": "some_id"
                }
            ],
            "type": "BIOMETRY",
            "analysis_id": "some_id",
            "resolution_status": "SUCCESS",
            "resolution": "SUCCESS"
        }
    ]
}
```


Interpret the results similar to those of the Liveness analysis:

```
$.resolution_status = "FINISHED" => The analysis is completed without technical errors, otherwise, the status would be FAILED
$.system_resolution = "SUCCESS" => The analysis confirms that both photos represent the same person
$.analyses[0].results_data.min_confidence => similarity score
```

You're done.