Overview

The project is divided into **3 logical blocks**.
Each block provides a service that can be used alone by itself related to the other contracts in the repository.
The logical blocks are:

1. **Create and sell an ERC721 token collection**
2. **Claim ERC1155 tokens**
3. **Manage a simple marketplace with a soft-token.**

Below there will be listed all the assumptions that have been made during the smart contracts design and development divided by logical block.

Assumptions are divided in "*Roles assumptions*" and "*External contracts assumptions*" (eventually a "*General assumptions*" paragraph can be present).

1. Create and sell an ERC721 token collection

This logical block is composed of the contracts contained in the CollectionMinter.sol and Erc721Collection.sol, plus one or more external ERC20 contract instances.

1.1. Roles assumptions

1.1.1. The roles that appear in the **Erc721Collection** are the following: PAUSER_ROLE, URI_EDITOR_ROLE and MINTER_ROLE. All these roles are considered to be granted to trusted people (people that belong to the company that is going to deploy the contracts) so we expect no concerns that come from the actions taken by wallets with these roles assigned.

1.1.2. The roles that appear in the **CollectionMinter** are the following: PAUSER_ROLE, TEAM_MINTER_and MANAGER_ROLE. All these roles are considered to be granted to trusted people (people that belong to the company that is going to deploy the contracts) so we expect no concerns that come from the actions taken by wallets with these roles assigned.

1.2. External contracts assumptions

1.2.1. The Erc721Collection doesn't use any external contract instance.

1.2.2. The CollectionMinter uses **one or more instances of ERC20** (or extensions of these ones) tokens. These instances are considered safe. At deployment and contract usage time the company that deploys the contracts assures that these tokens (ERC20 instances used to pay within the CollectionMinter) are safe and that there's no malicious code inside their contract code that can lead to hacks, loss of funds or other threats.

1.3. General assumptions

1.3.1. The contracts Erc721Collection and CollectionMinter are custom made and so they are considered safe by themselves.

2. Create and sell an ERC721 token collection

This logical block is composed of the contracts contained in the Erc1155Claimer.sol, plus one or more external ERC1155 contract instances.

2.1. Roles assumptions

2.1.1. The roles that appear in the **Erc1155Claimer.sol** are the following: PAUSER_ROLE, MANEGER_ROLE and NFTS_OPERATOR_ROLE. All these roles are considered to be granted to trusted people (people that belong to the company that is going to deploy the contracts) so we expect no concerns that come from the actions taken by wallets with these roles assigned.

2.2. External contracts assumptions

2.2.1. The Erc1155Claimer uses **one or more instances of ERC1155** (or extensions of these ones) tokens. These instances are considered safe. At deployment and contract usage time the company that deploys the contracts assures that these tokens (that will be used in the claim events within the Erc1155Claimer) are safe and that there's no malicious code inside their contract code that can lead to hacks, loss of funds or other threats.

2.3. General assumptions

2.3.1. The contract ERC1155Claimer is custom made and so it is **considered safe by itself**.

2.3.2. The contract includes a custom method that produces random distribution (used in the Rando claim event) that is **not gas efficient**. The cost for calling the operation that calculates the distribution is not a problem since the target amount of values in the distribution to be calculated is really low (5) and so even if the procedure is not so efficient the cost is acceptable.

2.3.3. The distribution calculated for the Random claim event can be calculated ahead of the claim itself and so the **claim result is predictable**. This is not a problem since during the development this possibility has been taken into account and generating uniform distributions in the random claim event is not our priority.

3. Create and sell an ERC721 token collection

This logical block is composed of the contracts contained in the SnowTracker.sol and SnowMarketplace.sol, plus one or more external ERC1155 and ERC721 contract instances.

3.1. Roles assumptions

3.1.1. The roles that appear in the **SnowTracker.sol** are the following: PAUSER_ROLE, MANEGER_ROLE and SPENDER_ROLE. All these roles are considered to be granted to trusted people (people that belong to the company that is going to deploy the contracts) so we expect no concerns that come from the actions taken by wallets with these roles assigned.

3.1.2. The roles that appear in the **SnowMarketplace.sol** are the following: PAUSER_ROLE, MANEGER_ROLE, SPENDER_ROLE and ORDERS_MANAGER_ROLE. All these roles are considered to be granted to trusted people (people that belong to the company that is going to deploy the contracts) so we expect no concerns that come from the actions taken by wallets with these roles assigned.

3.2. External contracts assumptions

3.2.1. The SnowMarketplace uses **one or more instances of ERC1155 and ERC721** (or extensions of these ones) tokens. These instances are considered safe. At deployment and contract usage time the company that deploys the contracts assures that these tokens (that will be used in the marketplace orders within the SnowMarketplace) are safe and that there’s no malicious code inside their contract code that can lead to hacks, loss of funds or other threats.

3.3. General assumptions

3.3.1. The contracts SnowTracker and SnowMarketplace are custom made and so they are considered safe by themselves.