Webpage contract using OpenZeppelin libraries. This contract validates the payment of a token and prints the result of the transaction on the RSK Testnet blockchain.

```solidity
pragma solidity ^0.8.0;

import "@openzeppelin/contracts/token/ERC20/IERC20.sol";

contract PaymentValidation {
address private paymentToken;

event PaymentVerified(address sender, uint256 amount);

constructor(address _paymentToken) {
paymentToken = _paymentToken;
}

function validatePayment(uint256 amount) external {
require(IERC20(paymentToken).balanceOf(msg.sender) >= amount, "Insufficient funds");
require(IERC20(paymentToken).allowance(msg.sender, address(this)) >= amount, "Insufficient allowance");

// Perform further validation if needed

// Print the result of the transaction
emit PaymentVerified(msg.sender, amount);
}
}
```

To deploy this contract on the RSK Testnet, follow these steps:

1. Install the necessary dependencies. This example assumes you are using Truffle framework to compile and deploy the contract. Run the following command in your project directory:

```
npm install @openzeppelin/contracts truffle truffle-hdwallet-provider@1.0.17
```

2. Create a `truffle-config.js` file in your project directory with the following content:

```javascript
```

```
const HDWalletProvider = require('truffle-hdwallet-provider');
const mnemonic = 'your twelve word mnemonic';
const infuraApiKey = 'your infura API key';

module.exports = {
networks: {
rskTestnet: {
provider: () => new HDWalletProvider(mnemonic, `https://public-node.testnet.rsk.co/${infuraApiKey}`),
network_id: 31,
gas: 6721975,
confirmations: 2,
timeoutBlocks: 200,
skipDryRun: true,
},
},
};
```

Make sure to replace `your twelve word mnemonic` with your actual mnemonic generated during wallet creation and `your infura API key` with your Infura API key.

3. Compile and deploy the contract by creating a new migration file, for example, `2_deploy_contract.js`, in your project's migrations directory with the following content:

```javascript
const PaymentValidation = artifacts.require("PaymentValidation");
const Token = "your token address"; // Replace with the token address you want to use for payment validation

module.exports = function (deployer) {
deployer.deploy(PaymentValidation, Token);
};
```

Replace `"your token address"` with the actual address of the ERC20 token you want to use for payment validation.

4. Run the migration by executing the following command from your project directory:

```
truffle migrate --network rskTestnet
```

```

This will deploy the contract on the RSK Testnet blockchain.

You can then interact with the contract through a web interface or any other method you prefer. The `validatePayment` function can be called with the required payment amount, and the contract will verify the balance and allowance of the payment token for the sender. If the validation succeeds, it will emit the `PaymentVerified` event which you can listen to and print the result of the transaction.
```