

---

# Software Requirements Specification

for

## Welcome Home

## Smart Escrow Contract

Version 1.0

Prepared by:

Casey Munga and Sergio Prieto

Smart Escrow Management

April 26<sup>th</sup> 2022

*Copyright © 2022 by Casey Munga and Sergio Prieto. All Rights reserved. This document cannot be shared distributed or its contents used in any way unless by express and written of the owners and developers of this product*

Copyright © 1994-1997 by Bradford D. Appleton. Permission is hereby granted to make and distribute verbatim copies of this document provided the copyright notice and this permission notice are preserved on all copies. (SDS sections)

Modified by Dr Renata Rand McFadden

# Table of Contents

<b>Table of Contents.....</b>	<b>ii</b>
<b>Revision History.....</b>	<b>ii</b>
<b>1. Introduction.....</b>	<b>1</b>
1.1 Purpose.....	1
1.2 Document Conventions.....	1
1.3 Intended Audience and Reading Suggestions.....	1
1.4 Product Scope.....	1
1.5 References.....	1
<b>2. Overall Description.....</b>	<b>2</b>
2.1 Product Perspective.....	2
2.2 Product Functions.....	2
2.3 User Classes and Characteristics.....	2
2.4 Operating Environment.....	2
2.5 Design and Implementation Constraints.....	2
2.6 User Documentation.....	2
2.7 Assumptions and Dependencies.....	2
<b>3. External Interface Requirements.....</b>	<b>3</b>
3.1 User Interfaces Overview.....	3
3.2 Hardware Interfaces.....	3
3.3 Software Interfaces.....	3
<b>4. System Features/Modules.....</b>	<b>3</b>
4.1 System Feature/Module 1.....	3
4.2 System Feature/Module 2 (and so on).....	4
<b>5. Nonfunctional Requirements.....</b>	<b>4</b>
5.1 Performance Requirements.....	4
5.2 Safety Requirements.....	4
5.3 Security Requirements.....	4
5.4 Software Quality Attributes.....	5
Appendix A: Glossary.....	5

## Revision History

Name	Date	Reason For Changes	Version
Casey Munga	04-26-2022	Initial Document	v1

# 1. Introduction

## 1.1 Purpose

*This is a Smart Contract for rental escrows using block chain technology. The smart contract solution will seek to reduce the need of trusted mediators, arbitration and enforcement costs when transacting rental property deposits, mitigating the risk of fraud and financial losses, as well as the reduction of malicious and accidental exceptions.*

## 1.2 Document Conventions

*There are no document conversions at this time.*

## 1.3 Intended Audience and Reading Suggestions

*This document's target audience is for the angel investor who are more technically inclined; the development team and stakeholders.*

## 1.4 Product Scope

*This is the first version of the project. The scope of the project will be completed in two phases. Phase I is the Smart contract creation – Code Name Welcome Home. The major functionality of creation of the user interface, transactional system and the deployment on the block chain. The second phase but out of scope will tackle a more comprehensive dashboard, the dispute process and an increase to the types of rentals such as short term vacation rentals, time shares and Air-BnBs. The smart contract solution will seek to reduce the need of trusted mediators, arbitration and enforcement costs when transacting rental property deposits, mitigating the risk of fraud and financial losses, as well as the reduction of malicious and accidental exceptions.*

## 1.5 References

*A **gateway** to the rental payment portal is listed below.*

*1. [Welcome Home Rental Payment Portal Smart Escrow Contract](#)*

### **Use Cases:**

*2. Project Overview*

*3. [Login](#)*

*4. [Initiate Contract](#)*

*5. [Create Contract](#)*

*6. [Create and Fill Wallet](#)*

*7. [Deploy Contract on Block Chain](#)*

## **2. Overall Description**

### **2.1 Product Perspective**

*This product is a new product that is designed to be scale-able web driven transactional app .*

### **2.2 Product Functions**

*Major features*

- *Smart contract creation*
- *Crypto wallet creation*
- *Users views*
- *Accounts*
- *Dashboards*
- *Contracts*
- *Transaction*
- *Block Chain Deployment*
- *Arbitration module (Phase II)*

### **2.3 User Classes and Characteristics**

*The intended target market will be a class of real estate holders known as landlord and renters or tenants.*

### **2.4 Operating Environment**

*Software environment will be a front end Web app that can accessed on a mobile device, tablet and laptop independent of any specific O/S.*

### **2.5 Design and Implementation Constraints**

*At this time developers will be limited to the web and software as a service. As the needs demands a downloadable app for different OS may be developed. However that sits outside the present scope. MongoDB or AWS will be the DB system. HTTP and latest security protocols must be adhered to. Block-chain and cryptocurrency will the official trading currency. HTML5 for client , php for server response. Simple and unfettered interface. Site must be responsive*

### **2.6 User Documentation**

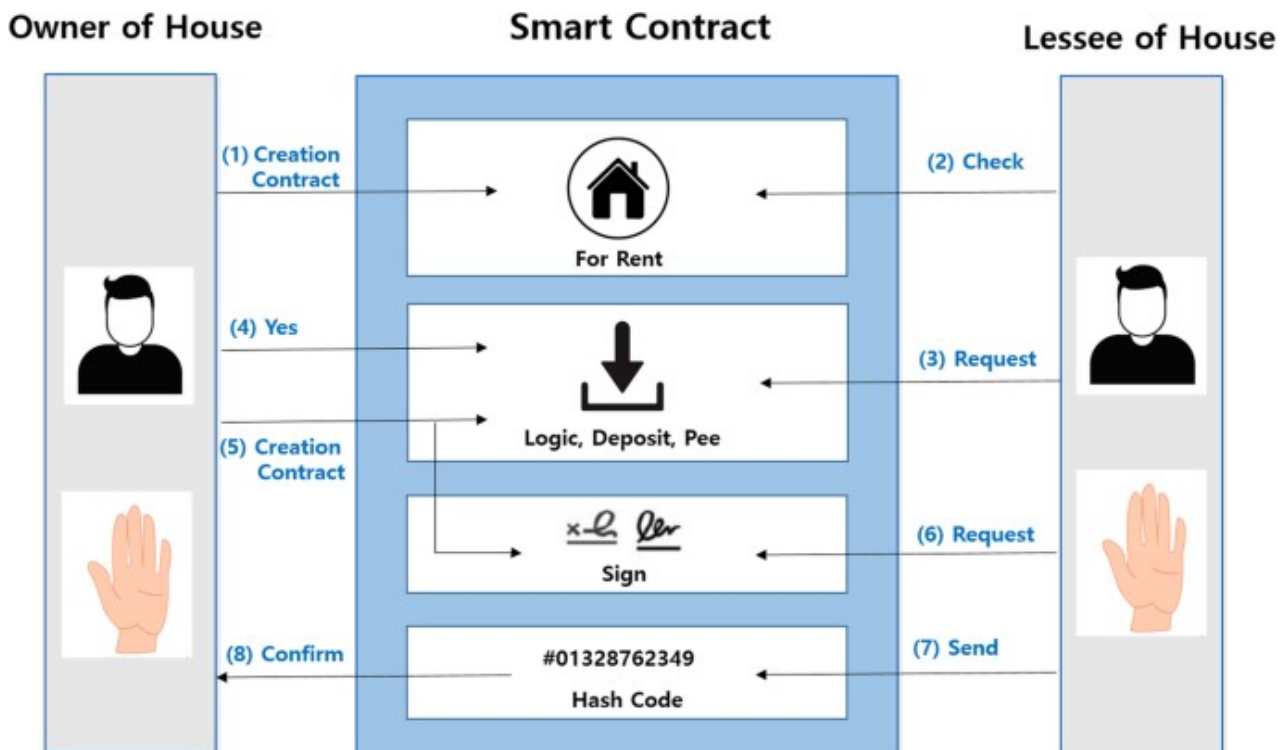
*Search and user help will be available inside the app. In app chat will be available inside the app during phase 2 but out of scope in this iteration.*

## 2.7 Assumptions and Dependencies

Selenium programming language, Truffle and block chain API s will be the main development.

## 3. External Interface Requirements

### 3.1 User Interfaces Overview



The entire interface is GUI Bases. The minimum requirements is a working device that has internet access. The rental portal gateway [Welcome Home Rental Portal Gateway](#) is a prototype. Standard buttons and interface will adopt the banding and themes for companies purchasing the Product as a Service. Buttons and navigation be programmatically changed according to the business process.

The details of each screen and the messages would be described in Human Interface section in design.

### 3.2 Hardware Interfaces

There are no requirements for hardware interfaces except that they should be internet accessible.

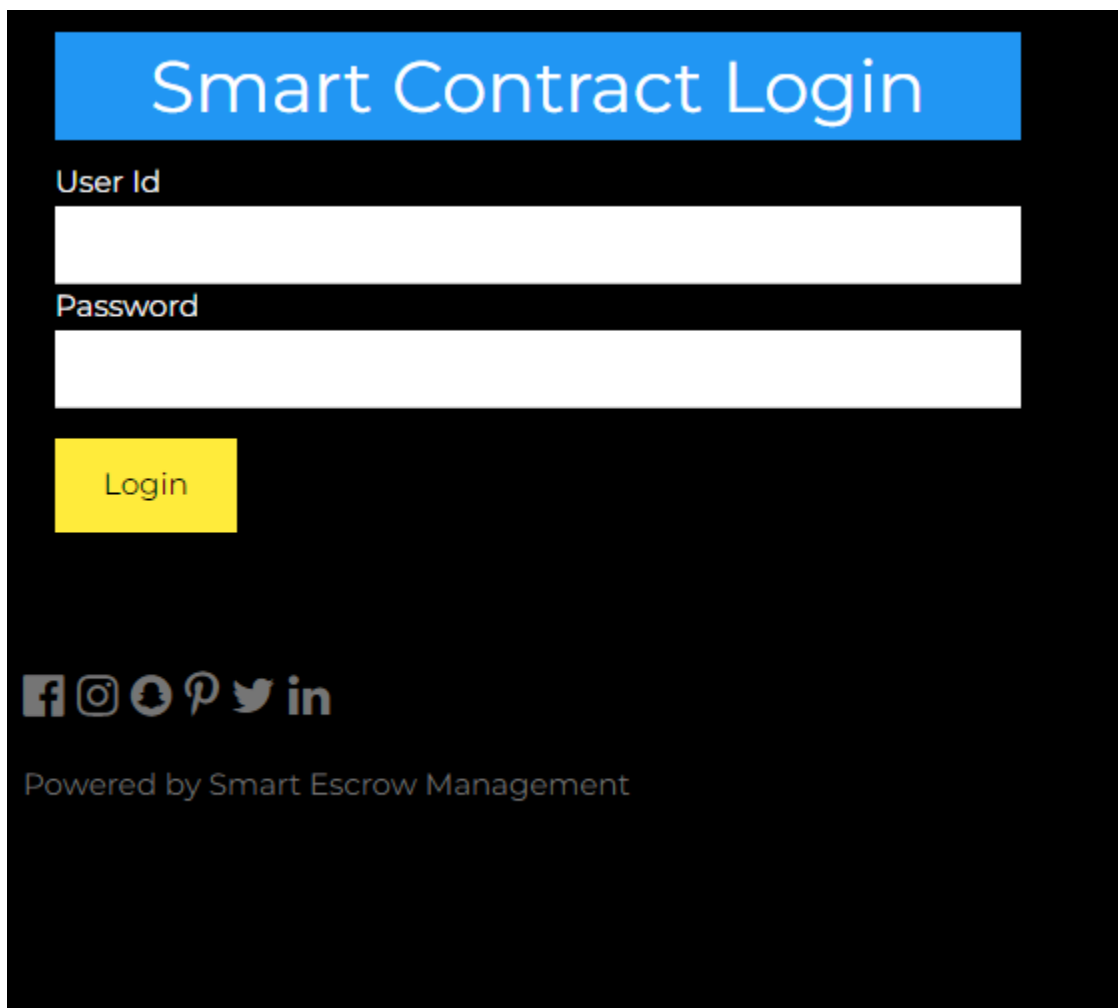
### 3.3 Software Interfaces

HTML5, node.js, Java script, PHP for Server side, MongoDB or AWS or Cassandra DB for the database access, https protocol, Interface with Banking for credit transaction to fill wallets. API for development on the Block-chain specifically truffle, Meta Mask, Selenium.

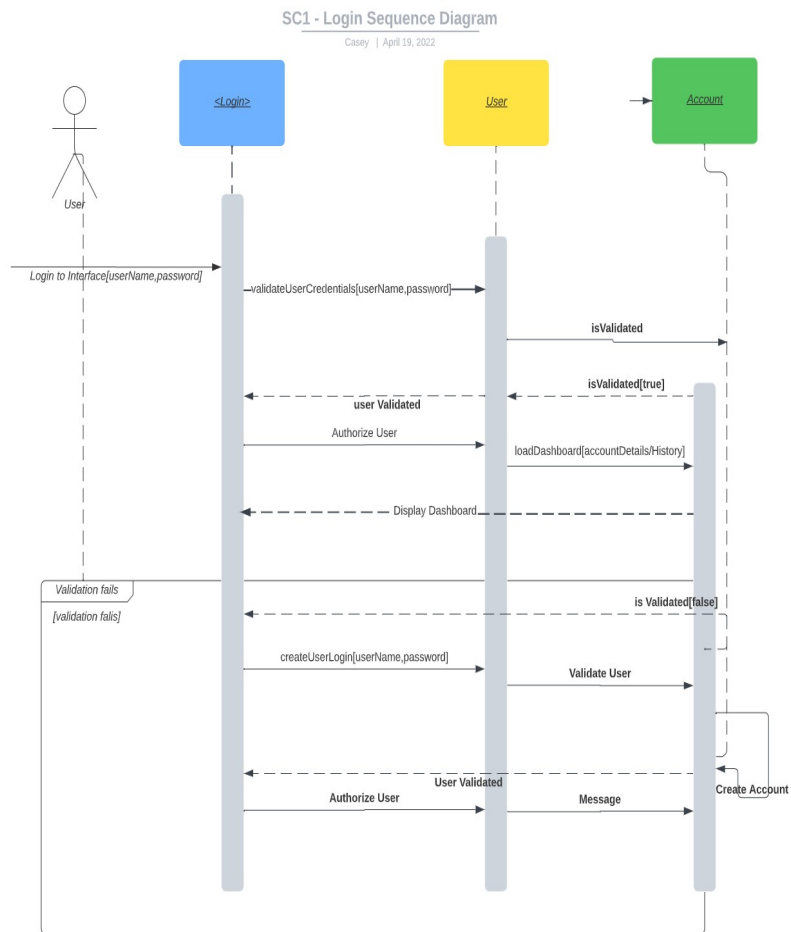
## 4. System Features/Modules

System Features and Modules will be available through the Sequence Diagrams

### 4.1 Login



The image shows a login interface for a 'Smart Contract'. It features a dark blue background. At the top, there is a light blue header bar with the text 'Smart Contract Login' in white. Below this, there are two white input fields: the first is labeled 'User Id' and the second is labeled 'Password'. A yellow 'Login' button is positioned below the password field. At the bottom of the interface, there is a row of social media icons (Facebook, Instagram, Snapchat, Pinterest, Twitter, and LinkedIn) and the text 'Powered by Smart Escrow Management'.



[Login -full-size-image](#)

## 4.2 Initiate Contract

*This form page is only available to the Owner/Landlord. The landlord will enter the information and select the initiate Contract button. A notification email will be sent to the tenant.*

**New Smart Contract**

Contract  
s567b2346d237

Tenant Name  
Jane Doe

Tenant address  
janedoe@gmail.com

Rental Date  
mm/dd/yyyy

Start Date  
mm/dd/yyyy

Release Date  
mm/dd/yyyy

Transfer Amount  
\$1,000,000.00

Transfer Date  
mm/dd/yyyy

Status

Transaction Fees  
\$1,000,000.00

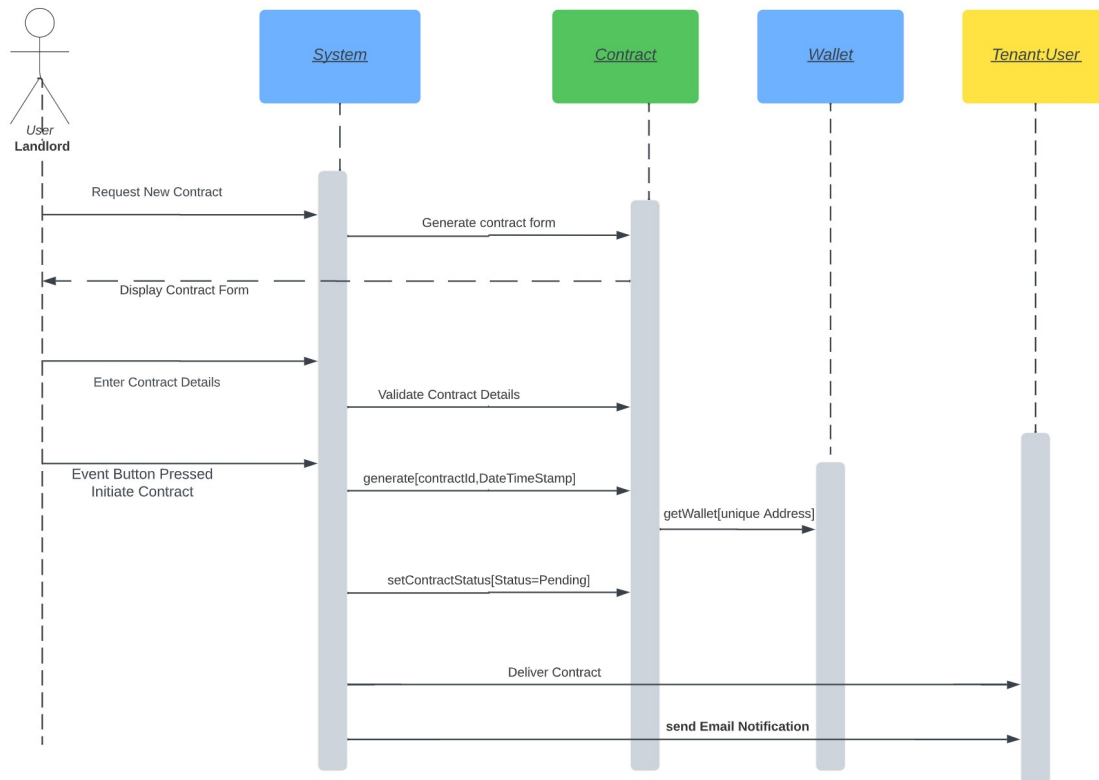
Contract Details

**Initiate Contract**



## SC2 - Create Contract Sequence Diagram

Casey Munga | April 19, 2022

[Initiate Contract full-size-image](#)

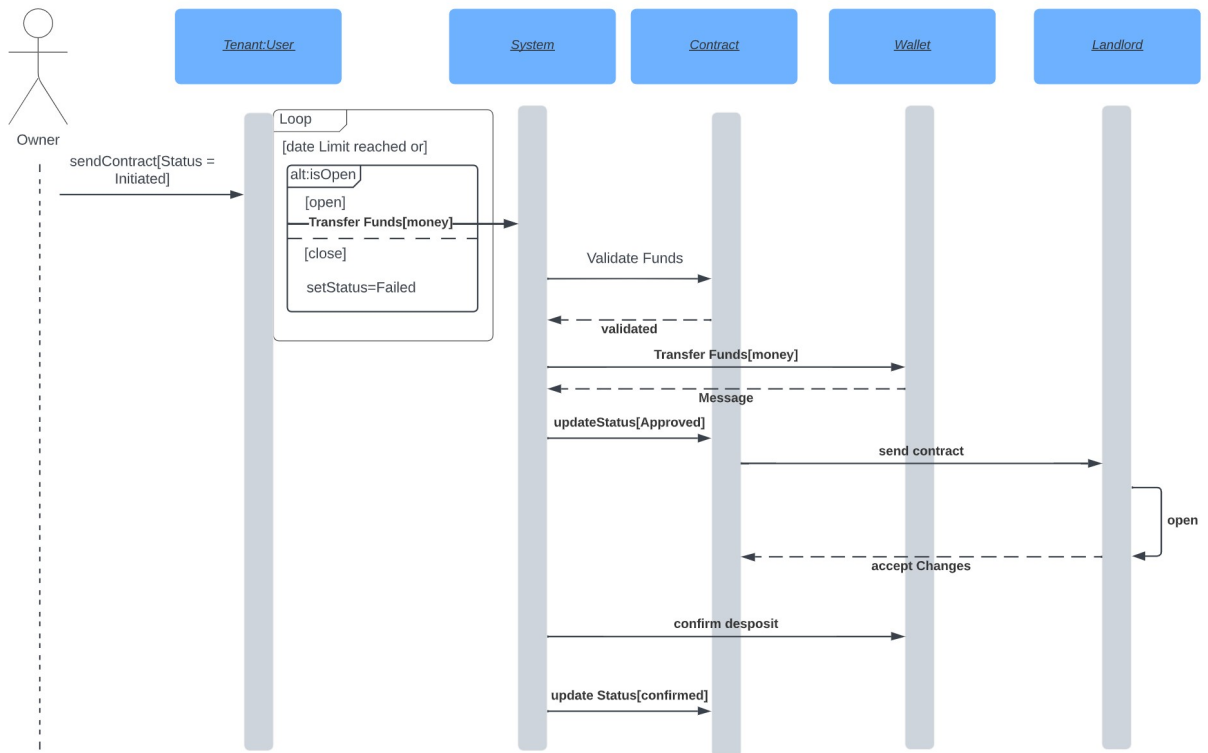
### 4.3 Create Contract

Owner receives an approved contract and will enter the transaction fees. The button changes to Confirm Contract.

The screenshot shows a mobile application interface for creating a new smart contract. On the left is a dark sidebar with navigation icons and labels: a menu icon, a magnifying glass labeled 'SEARCH', a house icon labeled 'HOME', a person icon labeled 'NEW', an eye icon labeled 'HISTORY', an envelope icon labeled 'DEPOSIT', and a right arrow icon labeled 'LOGOUT'. The main content area has a blue header 'New Smart Contract'. Below it are several input fields with labels and placeholder text: 'Contract' (placeholder: s567b2346d237), 'Tenant Name' (placeholder: Jane Doe), 'Tenant address' (placeholder: janedoe@gmail.com), 'Rental Date' (placeholder: mm/dd/yyyy with a calendar icon), 'Start Date' (placeholder: mm/dd/yyyy with a calendar icon), 'Release Date' (placeholder: mm/dd/yyyy with a calendar icon), 'Transfer Amount' (placeholder: \$1,000,000.00), 'Transfer Date' (placeholder: mm/dd/yyyy with a calendar icon), 'Status' (empty), 'Transaction Fees' (placeholder: \$1,000,000.00), and 'Contract Details' (empty). At the bottom is a yellow button labeled 'Initiate Contract'.

## SC3 - Create Contract Sequence Diagram

Casey Munga | April 19, 2022

[Create Contract-full-size-image](#)

## 4.4 Account

View when tenant log on to account. Navigation Buttons will be present. Tenant is presented with view of any new Initiated contract or the current contract. See use case 1.5 Tenant selects Accept contract and is presented with deposit page to deposit money into wallet

The screenshot displays a user interface for a tenant's account. On the left is a dark sidebar with navigation icons and labels: a hamburger menu, a magnifying glass labeled 'SEARCH', a house icon labeled 'HOME', an eye icon labeled 'HISTORY', an envelope icon labeled 'DEPOSIT', and a right-pointing arrow labeled 'LOGOUT'. The main content area has a blue header with the name 'Jane Doe'. Below this, various contract details are shown in white boxes with labels to their left: 'Contract' (s567b2346d237), 'Rental Date' (20/04/2022), 'Start Date' (20/04/2022), 'Release Date' (20/06/2023), 'Transfer Amount' (0.00), 'Transfer Date' (empty), 'Status' (INITIATED), 'Transaction Fees' (\$0.00), and 'Contract Details' (empty). At the bottom, there are two radio buttons labeled 'Accept Contract' and 'Decline Contract', and a yellow 'Update Contract' button.

Field	Value
Contract	s567b2346d237
Rental Date	20/04/2022
Start Date	20/04/2022
Release Date	20/06/2023
Transfer Amount	0.00
Transfer Date	
Status	INITIATED
Transaction Fees	\$0.00
Contract Details	

☒ Accept Contract ☐ Decline Contract

Update Contract

## 4.5 History

User is shown most recent contract transaction but can opt to see a list of all historical data of every transaction

**Contract History**

**Contract**

Contract: sa890dye8291ty

Rental Date: 06/25/2020

Start Date: 06/01/2020

Release Date: 05/30/2021

Transfer Amount: \$560.00

Transfer Date: 06/01/2021

Status: PAID

Transaction Fees: 0.0000362889 ETH

**Contract Details**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies, purus lectus malesuada libero, sit amet commodo magna eros quis urna. Nunc viverra imperdiet enim. Fusce est. Vivamus a tellus. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Proin pharetra nonummy pede. Mauris et orci. Aenean nec lorem. In porttitor. Donec laoreet nonummy augue. Suspendisse dui purus, scelerisque at, vulputate vitae, pretium mattis, nunc. Mauris eget neque at sem venenatis eleifend. Ut nonummy.

Next

## 4.6 Desposit

*This page is served when the user wants to deposit money from a banking institution into his wallet and purchase ethereum.*

The screenshot shows a mobile application interface for an "Ethereum Deposit" form. The interface has a dark theme with a sidebar on the left containing navigation icons: a hamburger menu, a magnifying glass labeled "SEARCH", a house icon labeled "HOME", a person icon labeled "NEW", an eye icon labeled "HISTORY", and an envelope icon labeled "DEPOSIT". The main content area has a blue header "Ethereum Deposit". Below the header are several input fields: "Contract" (empty), "Wallet" (empty), "Deposit Amount" (pre-filled with "\$1,000,000.00"), "Deposit Date" (placeholder "mm/dd/yyyy" with a calendar icon), "Transfer Date" (placeholder "mm/dd/yyyy" with a calendar icon), "Status" (empty), "Transaction Fees" (pre-filled with "\$1,000,000.00"), and "Contract Details" (empty). A yellow "Deposit Now" button is located below the "Contract Details" field. At the bottom of the form, there are social media icons for Facebook, Instagram, YouTube, Pinterest, Twitter, and LinkedIn, followed by the text "Powered by Smart Escrow Management".

Ethereum Deposit	
Contract	<input type="text"/>
Wallet	<input type="text"/>
Deposit Amount	<input type="text" value="\$1,000,000.00"/>
Deposit Date	<input type="text" value="mm/dd/yyyy"/>
Transfer Date	<input type="text" value="mm/dd/yyyy"/>
Status	<input type="text"/>
Transaction Fees	<input type="text" value="\$1,000,000.00"/>
Contract Details	<input type="text"/>

[Deposit Now](#)

[f](#) [i](#) [y](#) [p](#) [t](#) [in](#)

Powered by Smart Escrow Management

## 4.7 Create Wallet

All currency must be kept in a wallet. New users must create a wallet and assign an external entity such as a credit card, bank account, paypal and must transfer / deposit money into their account at this stage. The wallet is a transactional component and will be used to uniquely identify the account holder. An account can only have one wallet, but a wallet can be used in more than one account. This form will be repurposed programmatically to create wallet when served up

**Ethereum Deposit**

Contract

Wallet

Deposit Amount

\$1,000,000.00

Deposit Date

mm/dd/yyyy

Transfer Date

mm/dd/yyyy

Status

Transaction Fees

\$1,000,000.00

Contract Details

Deposit Now

SEARCH

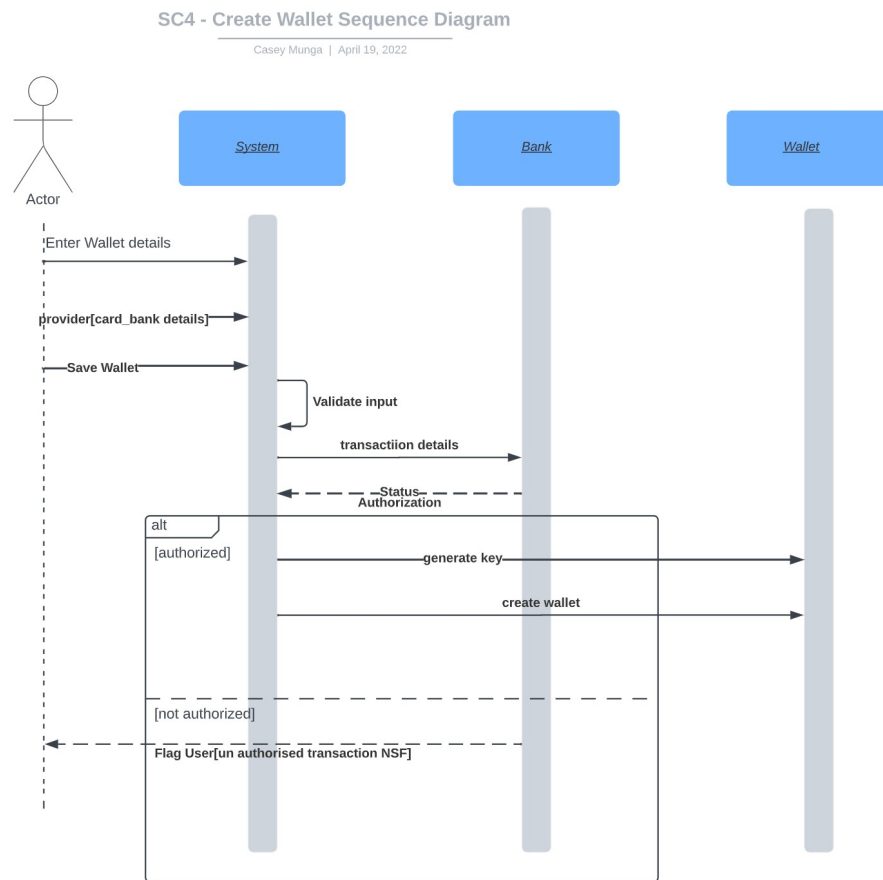
HOME

NEW

HISTORY

DEPOSIT

Powered by Smart Escrow Management

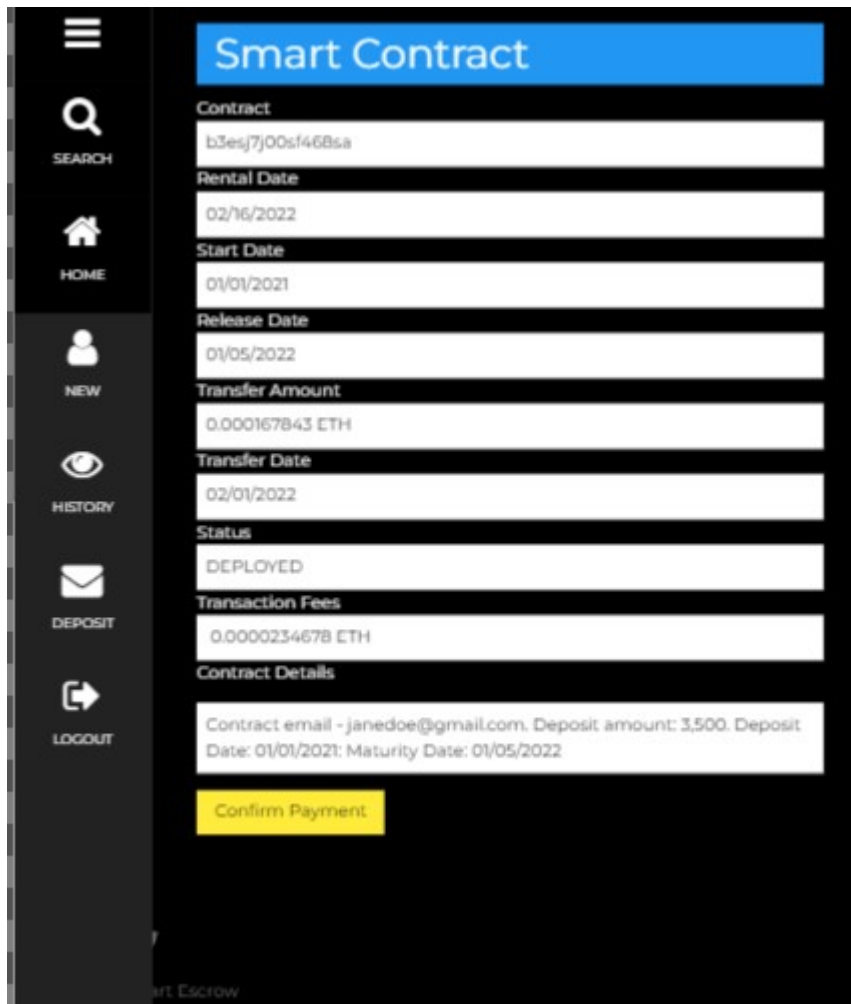


[Create Wallet full-size-image](#)



## 4.8 Deploy Contract

*This page is available after the contract had been confirmed and all fees paid. The button's text property will be changed to Deployed. The owner can now deploy the account to the block chain. Please see use case section 1.5*



The image shows a mobile application interface for a 'Smart Contract'. On the left is a dark sidebar with navigation icons and labels: a hamburger menu, a magnifying glass labeled 'SEARCH', a house icon labeled 'HOME', a person icon labeled 'NEW', an eye icon labeled 'HISTORY', an envelope icon labeled 'DEPOSIT', and a right-pointing arrow labeled 'LOGOUT'. The main content area has a blue header 'Smart Contract'. Below it, a list of contract details is shown in white boxes with black text: 'Contract' (b3esj7j00sf468sa), 'Rental Date' (02/16/2022), 'Start Date' (01/01/2021), 'Release Date' (01/05/2022), 'Transfer Amount' (0.000167843 ETH), 'Transfer Date' (02/01/2022), 'Status' (DEPLOYED), and 'Transaction Fees' (0.0000234678 ETH). A 'Contract Details' section contains the text: 'Contract email - janedoe@gmail.com. Deposit amount: 3,500. Deposit Date: 01/01/2021; Maturity Date: 01/05/2022'. At the bottom of the main area is a yellow button labeled 'Confirm Payment'.

Field	Value
Contract	b3esj7j00sf468sa
Rental Date	02/16/2022
Start Date	01/01/2021
Release Date	01/05/2022
Transfer Amount	0.000167843 ETH
Transfer Date	02/01/2022
Status	DEPLOYED
Transaction Fees	0.0000234678 ETH

**Contract Details**

Contract email - janedoe@gmail.com. Deposit amount: 3,500. Deposit Date: 01/01/2021; Maturity Date: 01/05/2022

[Confirm Payment](#)

## **5. Nonfunctional Requirements**

### **5.1 Performance Requirements**

*Access and throughput systems must be measured in nanoseconds or at bench marking levels.*

### **5.2 Safety Requirements**

*N/A*

### **5.3 Security Requirements**

*Users will be authenticated and authorized to use the system. Passwords should be strong and user prompted to change their password every 90 days. Additionally 2 step verification must be implemented. The latest hashing functions on databases and banking information needs to be protected. Data base and app will be kept exclusively in the cloud.*

### **5.4 Software Quality Attributes**

*The interface will be intuitive and easy to use without. All efforts will be made to keep functionality accessible to person with sight and mobility issues. All standard will be kept as per ADA and governmental regulations.*

## **Appendix A: Use Cases**

## Smart Contract SC1 - Login Use Case

**Scope:** User and System

**Primary Actor:**User, System

**Stakeholders and Interests:**

The Login System wants to have the User to be validated and authorized. User wants access to account.

**Preconditions:** User needs to already have a valid Account

**Success Scenario:**

1. User enters the credentials and presses the login button
2. The System open the file and checks the username and password in the file
3. The credentials are validated.
4. The system authorizes the account for access.

**Post Condition:**

System shows user account page. System unloads Login Page.

**Exception :**

1. User enters the incorrect credentials.  
Systems Displays error Message
2. User is not registered in the System  
User needs to create an account.
3. User is prompted to create an account to use the System.

## Smart Contract SC2 – Initiate Contract Use Case

**Scope:** System

**Primary Actor:** System, Smart Contract

**Secondary Actor:** User (Landlord)

**Stakeholders and Interests:**

The user wants to initiate a contract. The system want to Create a Smart Contract. The Smart Contract wants to be created without error and initiated.

**Preconditions:** The User must be validated and logged into the Escrow System

**Success Guarantee:**

**Main Success Scenario:**

1. User enters Contract Details
2. The System captures field
3. The System validate the fields for errors.
4. User selects Initiate Contract Button.
5. System initiates the Contract.
6. System generates a unique contract id and date time stamp.
7. System checks Landlord Wallet for Unique Key
8. System sets status field to Pending.
9. System serializes the Contract fields are stored in a json object.
10. Contract is sent to the renter for acceptance.
11. The Renter is sent a notification email to the email address on file for the account.

**Extensions:**

**\* User can close the form without saving a Smart Contract**

2.1 User does not press the Initiate Contract Button or page loses focus or Close button is selected.

- Flag User- Message to discard data.

2.1.1 User opts to discard data

- re-initialize page.

2.1.2 User opts to save data and initiate Contract.

- Serialize data json file and append data to file

4.1 System does not select Initiate Contract Button instead selects the Close Button

- The System clears the form and Closes the form

## Smart Contract SC3 – Create Contract Use Case

**Scope:** User and System

**Primary Actor:** Tenant, Smart contract

**Secondary Actor:** Landlord

**Stakeholders and Interests:** Smart Contract wants to be created. Rental deposit wants to be entered and transferred. Tenant wants to transfer money from the Wallet. Landlord wants to accept deposit and update Status from Initiated to Created.

**Preconditions:** Smart Contract must have been initiated

**Success Guarantee:** Tenant approves Rental Deposit Transaction. Wallet debits Rental deposit cost. Tenant approves Smart Contract.

**Main Success Scenario:**

1. Tenant receives the Initiated Smart Contract
2. Tenant opens the Smart Contract
3. Tenant transfers funds Rental Deposit from the Wallet
4. Tenant selects Approve Contract Button
5. System validates funds deposited is equal to funds requested
6. System updates Status from Initiated to Approved
7. System Updates Smart Contract
8. System sends approved Contract to Landlord
9. Landlord opens approved Contract
10. Landlord confirms Rental Deposit amount
11. System Updates Smart Contract from Approved to Confirmed

### Extensions

- 2.1 Tenant does not open the Smart Contract within the time limit
- Smart Contract Status is updated to Failed
  - System Flags Landlord

- 3.1 Tenant does not approve the Contract
- 3.1.1 Rental Amount is incorrect  
Tenant does not approve and selects reason  
Landlord adjust amount and resend contract
- 3.1.2 Tenant does not have enough funds in wallet  
Tenant fills wallet with funds  
Start at step 3 – 11.

- 10.1 Landlord does not Confirm contract
- 10.1.1 Incorrect rental deposit sent by tenant

## Smart Contract SC3 – Create Wallet Use Case

**Scope:** User and System

**Primary Actor:** Wallet, System

**Secondary Actor:** User

**Stakeholders and Interests:**

User wants to create a wallet. System wants the user to create a wallet and fill it with Ether.

**Preconditions:** User must have a validated account and Create Wallet Button has been selected

**Success Guarantee:** User enters Wallet details. Wallet links to bank for validated funds. Funds are deposited into Wallet. Wallet is created.

**Main Success Scenario:**

1. User enters Wallet details.
2. User deposits money from Cash, Bank Card, Credit Card into Wallet
3. User selects Save Wallet Button:
4. System validates Money
5. System sends transaction validation to Bank
6. System receives Bank authorization
7. System generate unique key for Wallet
8. System creates Wallet

**Extensions:**

\*\* User can cancel creating Wallet at any time

6.1 System receives NSF code

- System Flags User – Insufficient Funds
- System Flags User – Retry Transaction

8.1 Wallet not created

- Insufficient Funds – User can use alternate source of funds

**Post Conditions**

## Smart Contract 5 – Deploy Contract Use Case

**Scope:** User and System

**Primary Actor:** Contract, System

**Secondary Actor:** Landlord

**Stakeholders and Interests:**

Landlord wants to Deploy Contract to the Block Chain. Contract wants to be deployed on the Block Chain. Block chain wants a confirmed contract to be deployed.

**Preconditions:**

Smart contract has to be approved and confirmed.

**Success Guarantee:** Contract is created. Money is sent to contract wallet. Secure key is generated. Applicable gas fees are deducted and recorded. Contract receipt is generated. Contract is deployed on the block chain

**Main Success Scenario:**

1. Landlord triggers contract
2. System initiates a smart contract creation
3. Contract checks wallet of tenant for unique digital key
4. Contract checks wallet of landlord for unique digital key
5. Contract checks ether balance to be transferred from tenant wallet
6. Contract creates contract with rules and its own wallet
7. Contract calculates transaction fees, gas and max contract size
8. Contract checks landlord account for maximum gas and fees
9. Contract debits landlord and tenants wallet and credit its wallet
10. Contract deploys itself into the block chain
11. Contract receives transaction receipt and public digital key
12. Contract notifies System of public Keys
13. System flags itself of transaction success Status
14. System Updates contract Status to Deployed
15. Block Chain check rules daily for maturity date and time.

**Extensions:**

5.1 Tenant does not have enough ether for deposit

- Contract will not be created
- System will flag Tenant
- Tenant deposits funds to increase deposit amount
- 

4.1 Landlord does not have enough ether in his wallet to cover gas and transaction fees

- 4.1.1 Contract will not be created
  - System will Flag the Landlord