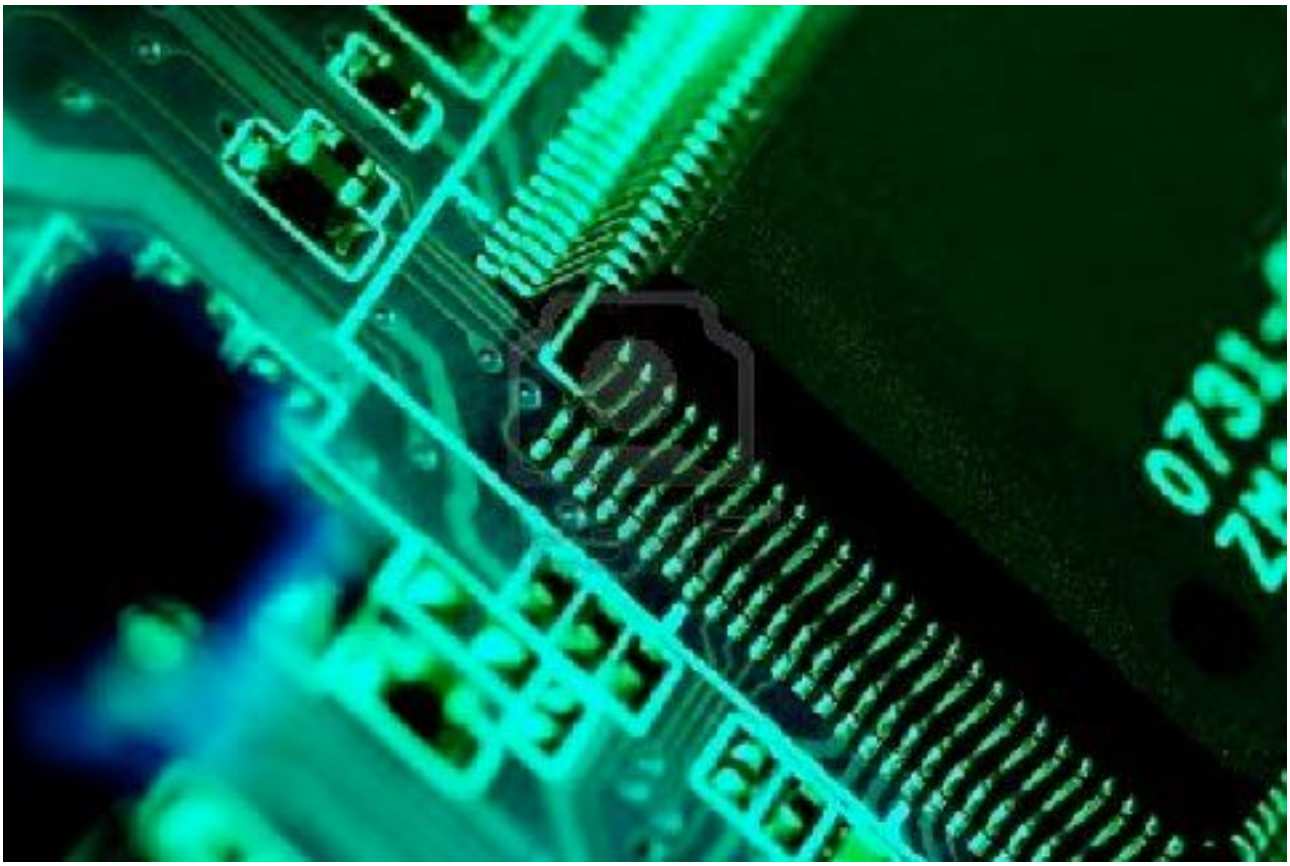
# *LOGIC CIRCUIT SIMULATOR*

*Software Requirements Specification*



Submitted by:

Sanchit Taliyan (B16CS031)
Uneet Meena (B16CS037)

# Table Of Contents

# 1. Introduction

## 1.1. Purpose

Purpose of this document is that a user by reading it can understand how the software is going operate from start to end. It shows how the development of software is done and what are its advantages.

## 1.2. Advantage of Logic Simulation

A logic simulator is a computer program which we(experimenters and designers) can use to perform virtual test of complex digital circuitry before we are going to work with any hardware. Before the emergence of logic simulators, engineers used to design electrical digital devices and systems by going through a monotonous mixture of trial and error of hardware manipulation and learned guess work.

## 1.3. Scope

This software will be designed to solve logic expressions and generate the corresponding truth table. This software can be used by anyone who has downloaded it for free.

User can give input in the form of boolean expression with variables and well defined operands. The software should be able to solve the boolean expression and should be able to generate the truth table corresponding to the solved result. All the truth tables that had been solved for a particular equation will be stored in the database.

Basically operands represents gates hence, for different combination of gates we can generate truth table by using this software ,which will tell the behaviour of a particular circuit.

## 1.4. Constraints

1. First input (variables) given to the software will of the form of English alphabets .

2. Particular operands( + , . , etc.) will be assigned to a unique gate which will represent that gate.

3. Any other input which is not defined if used in the Boolean expression then that expression will be taken as invalid.

## 1.5. Assumptions and Dependencies

1. All the users should know the different operands assigned for a particular gate.
2. Output will be generated only in the case of valid input. So , it is expected from the user to know the order of operators.
3. Number 1 represents true output/input and 0 represents false output/input .
4. One expression can be solved at a time. Only after generation of truth table of previous expression next expression will be generated.

# 2. Overall Description

## 2.1. Product Perspective

A logic circuit simulation system stores the following information.

- **Logic Gates and Symbols:**

  It includes the list of all the logic gates to be used, along with the symbols.

- **Registered Account:**

  It includes user's name, mail id and phone number. This information may be used for keeping the records of the user for any other kind of information.

- **Database**

  It includes the generated truth table by taking logic expression as input.

## 2.2. Product Functions

1. The user have to create an account where he will generate a user id and password for further login to the app.
2. For solving a given boolean expression, system will read the expression first, will decode it by separated recognise the symbol and characters present in the expression.
3. It will generate the output as truth table by using logic gates and symbols for solving the truth table and it will save the output for future use.

# 3. Requirement Specification

## 3.1 Functional Requirements

### 3.1.1. User: Login

| Use Case Name | Login |
|---|---|
| **Trigger** | User is a member |
| **Precondition** | 1. User should have registered.<br>2. Each user should have a unique user name. |
| **Basic Path** | 1. Select Login option.<br>2. Enter valid username and password.<br>3. The software will check for the validation.<br>4. If valid credentials user will be logged in.<br>5. If invalid password, system asks user if forgot password |
| **Alternative Paths** | No |
| **Postconditions** | User will be logged in. |
| **Exception Paths** | Wrong user name and password. |
| **Other** | None |

### 3.1.2. User: Register To The System

| Use Case Name | Register To The System |
|---|---|
| **Trigger** | When a new member needs to login |
| **Preconditions** | 1. User should be a human.<br>2. User should get a unique username, which has been not registered before. |
| **Basic Path** | User enter all the details which has been asked in the registration form. |
| **Alternative Path** | None |
| **Postconditions** | New account for the user will be created , generating a unique user name. |
| **Exception Paths** | If the the registration form is not filled validly no registration done. |

| Use Case Name | Register To The System |
|---|---|
| **Other** | User details contain first name , last name, date of birth, email id, sex, etc. |

## 3.1.3. User: Logical Expression

| Use Case Name | Logical Expression |
|---|---|
| **Trigger** | User feeds in boolean expression for evaluation. |
| **Preconditions** | 1. User should have logged in.<br>2. User should have a good knowledge about boolean expressions.<br>3. User should feed valid logical expression. |
| **Basic Path** | User enter valid logical expression. |
| **Alternative Path** | None |
| **Postconditions** | Data will be verified by the system and output in the form of truth table will be generated. |
| **Exception Path** | User enters invalid expression |
| **Others** | None |

## 3.1.4. System: Valid Input

| Use Case Name | Valid Input |
|---|---|
| **Trigger** | User feeds in the logical expression. |
| **Preconditions** | Feed should be given to the system. |
| **Basic Path** | System check if the input expression is valid. |
| **Alternative Path** | None |
| **Postcondition** | Expression will be approved and will be send for arithmetic operation. |
| **Exception Path** | System encounters invalid expression |
| **Others** | Input expression contains variables and operands. |

### 3.1.5. System: Decoding

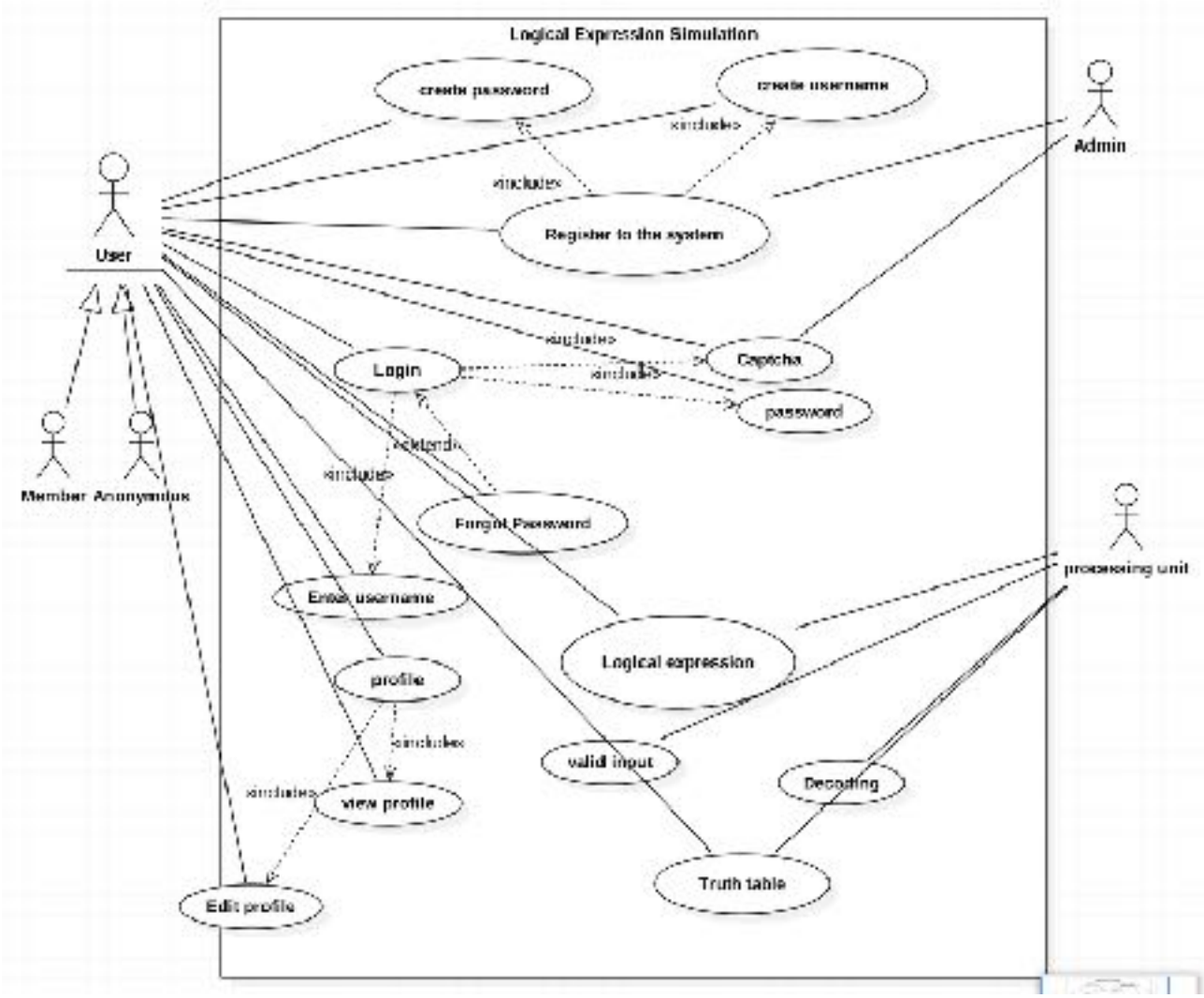
| Use Case Name | Decoding |
|---|---|
| **Trigger** | System has validated the input |
| **Preconditions** | Input expression must be valid. |
| **Basic Path** | 1. Firstly the system will convert the expression into it's preorder form.<br>2. Then it will store that form into the stack.<br>3. Data items will be popped one by one and evaluation will be done till whole equation is parsed. |
| **Alternative Path** | None |
| **Postconditions** | Arithematic operation will be completed , generation of truth table will be done. |
| **Exception Path** | No input is given |
| **Others** | Variables will be of bool type |

### 3.1.6. System: Truth Table

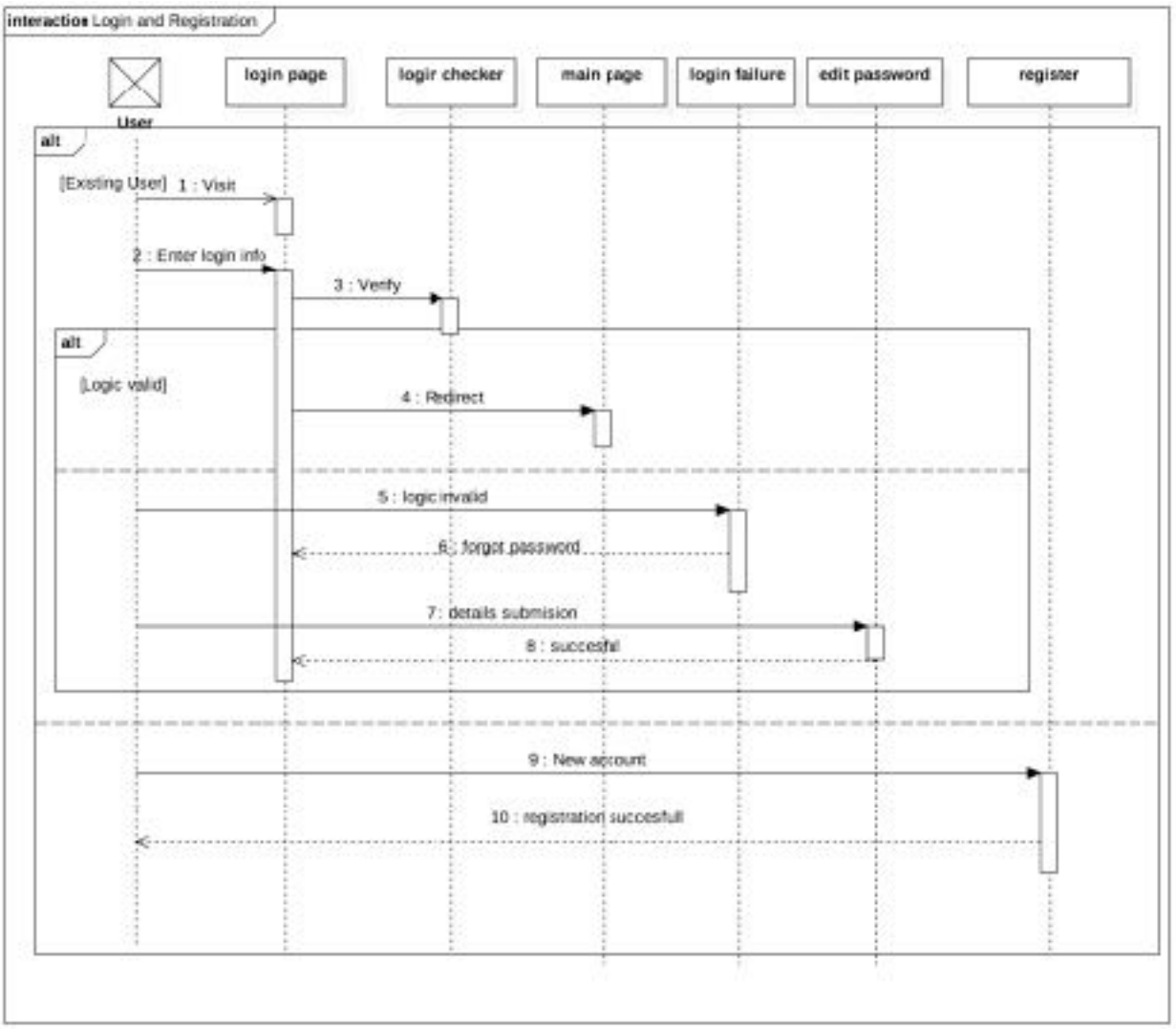
| Use Case Name | Truth Table |
|---|---|
| **Trigger** | 1. Valid input has been checked<br>2. System has done arithmetic operation on it. |
| **Preconditions** | Input expression is valid |
| **Basic Path** | For all the values of input variables output will be stored in a table. |
| **Alternative Path** | None |
| **Postconditions** | A truth table will be generated on the main screen go system and also the truth table will be stored int the database of the system. |
| **Exception Path** | No input is given |
| **Others** | The truth table is stored in the data base of user by updating its data. |

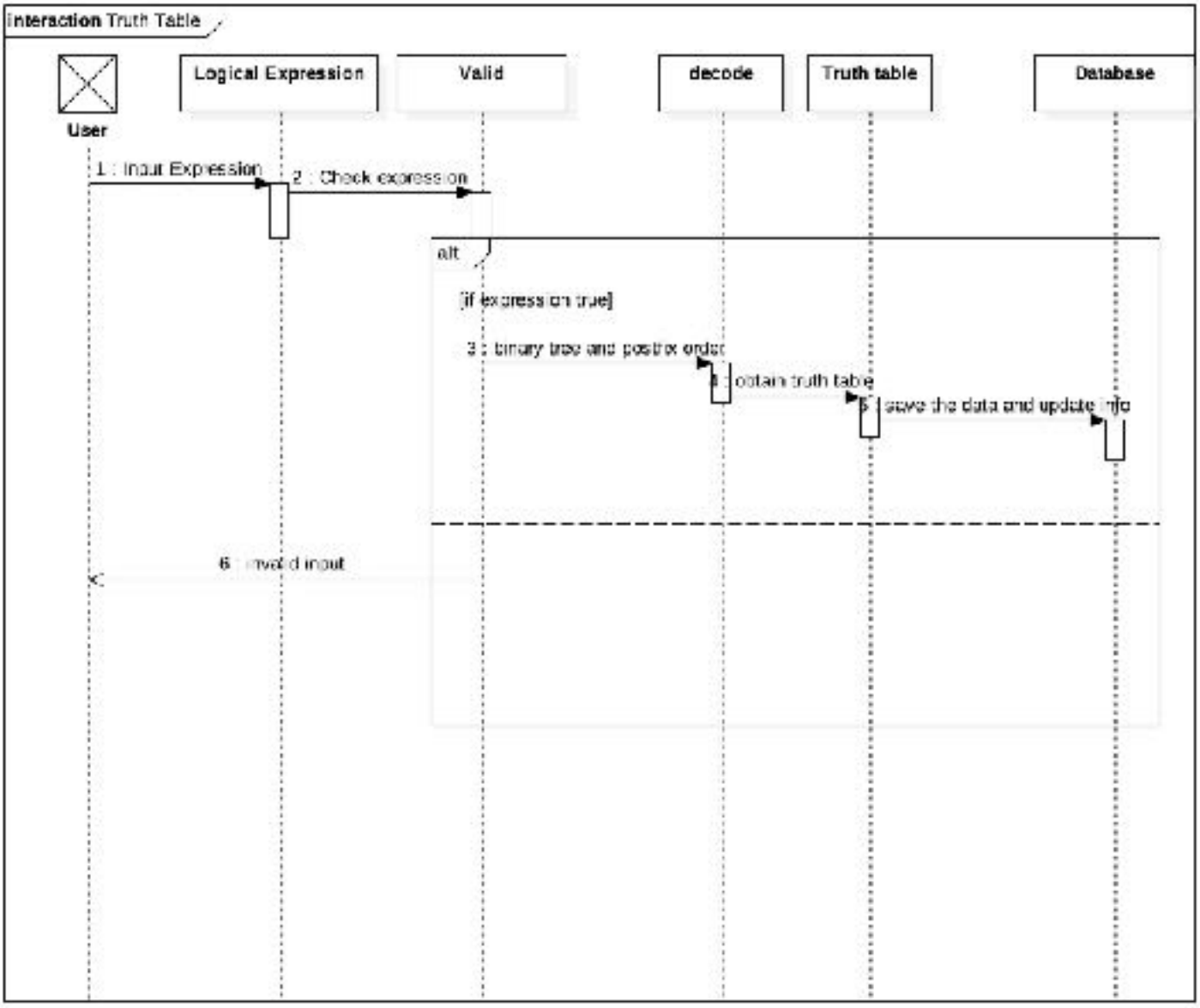
# 4. Uml Diagrams

## 4.1. Use Case Diagram

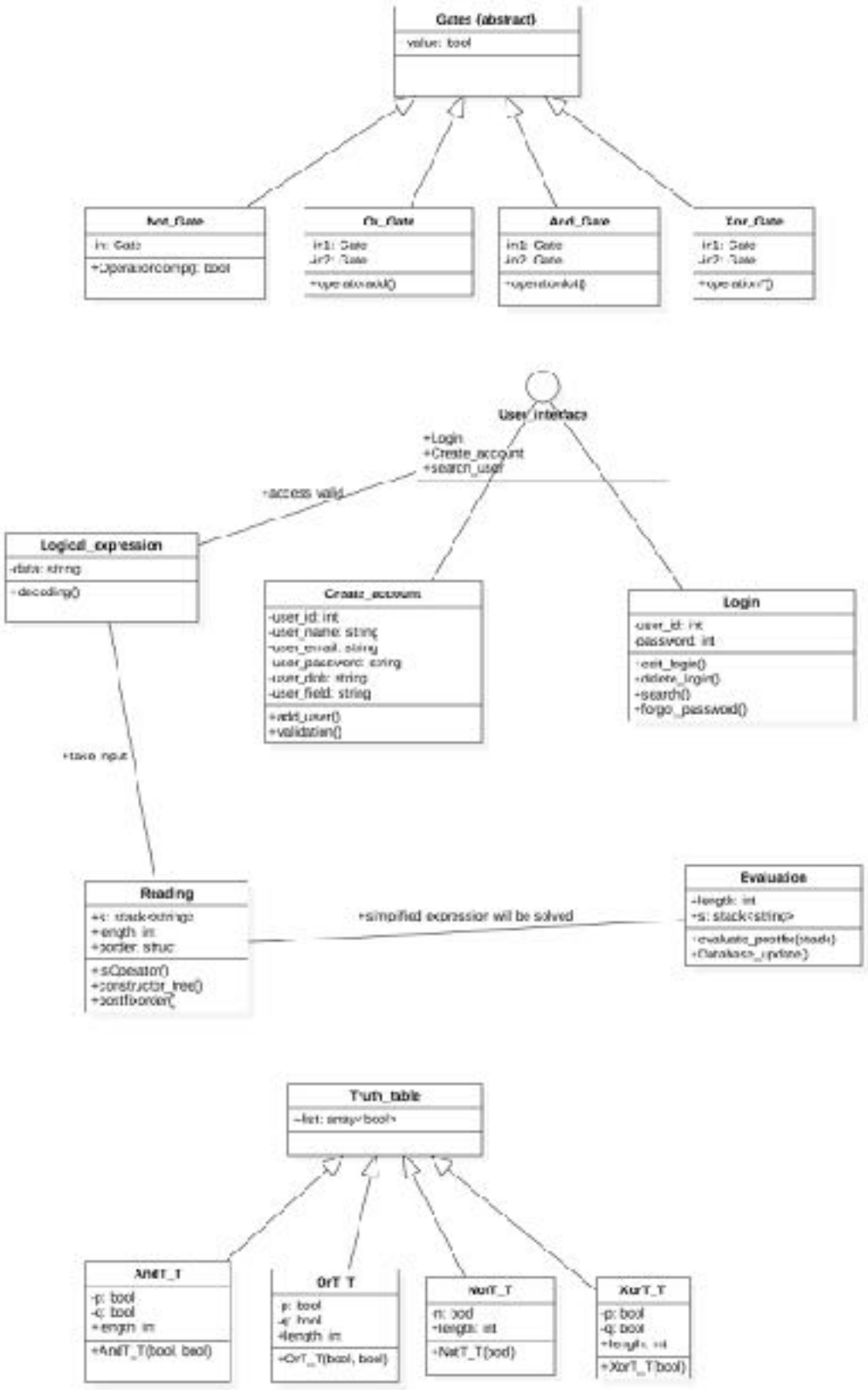


## 4.2. Sequence Diagram




interaction Truth Table
User
Logical Expression
Valid
decode
Truth table
Database
1 : Input Expression
2 : Check expression
alt
[if expression true]
3 : binary tree and postfix order
4 : obtain truth table
5 : save the data and update info
6 : invalid input

# 4.3. Class Diagram



# 5. System Features

## 5.1 Description and Priority

The logic simulator system maintains information of various logic gates, symbols and generated truth table. Of course , this project has a high priority because it is very difficult to decode some expression and the output.

## 5.2. Stimulus/Response Sequences

1. Decode the logic expression and separate the symbols and gates.
2. Displayed a detailed truth table after solving the given boolean expression.
3. Store three output of a solved boolean expression for future use.