

A

Project Report

On

RentAToy

Submitted by

Akhilesh Mangesh Sathe

Roll No.: 22356

MCA–II

SEM–III

Under the guidance of

Prof. Punam Chaudhari

For the Academic Year 2022-23



Sinhgad Technical Education Society's

Sinhgad Institute of Management

Vadgaon Bk Pune 411041

(Affiliated to SPPU Pune & Approved by AICTE New Delhi)

Prof. M. N. Navale
M.E. (ELECT.), MIE, MBA
FOUNDER PRESIDENT

Dr. (Mrs.) Sunanda M. Navale
B.A., MPM, Ph.D
FOUNDER SECRETARY

Dr. Chandrani Singh
MCA, ME, (Com. Sci.), Ph.D
DIRECTOR - MCA

Date:

CERTIFICATE

This is to certify that Mr Akhilesh Mangesh Sathe has successfully completed his project work entitled “**RentAToy**” in partial fulfillment of MCA – II SEM – III Mini Project for the year 2023-2024. He has worked under our guidance and direction.

Prof. Punam Chaudhari
Project Guide

Dr. Chandrani Singh
Director, SIOM-MCA

Examiner 1

Examiner 2

Date:

Place: Pune

Celebrating 25 Years
OF ACADEMIC EXCELLENCE

DECLARATION

I certify that the work contained in this report is original and has been done by me under the guidance of my supervisor(s).

- The work has not been submitted to any other Institute for any degree or diploma.
- I have followed the guidelines provided by the Institute in preparing the report.
- I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.
- Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the report and giving their details in the references.

Name and Signature of Project Team Members:

Sr. No.	Seat No.	Name of students	Signature of students
1		Akhilesh Mangesh Sathe	

ACKNOWLEDGEMENT

It is very difficult task to acknowledge all those who have been of tremendous help in this project. I would like to thank my respected guide **Prof. Punam Chaudhari** for providing me necessary facilities to complete my project and also for their guidance and encouragement in completing my project successfully without which it wouldn't be possible. I wish to convey my special thanks and immeasurable feelings of gratitude towards **Dr. Chandrani Singh, Director SIOM-MCA**. I wish to convey my special thanks to all teaching and non-teaching staff members of **Sinhgad Institute of Management, Pune** for their support.

Thank You

Yours Sincerely,

Akhilesh Mangesh Sathe

INDEX

Sr. No.	Chapter	Page No.
1	CHAPTER 1: INTRODUCTION	
1.1	Abstract	
1.2	Existing System and Need for System	
1.3	Scope of System	
1.4	Operating Environment Hardware and Software	
1.5	Brief Description of Technology used	
2	CHAPTER 2: PROPOSED SYSTEM	
2.1	Feasibility Study	
2.2	Objectives of the proposed system	
2.3	Users of the system	
3	CHAPTER 3: ANALYSIS AND DESIGN	
3.1	Entity Relationship Diagram (ERD)	
3.2	Class Diagram	
3.3	Use Case Diagrams	
3.4	Activity Diagram	
3.5	Sequence Diagram	
3.6	Component Diagram	
3.7	Module and Hierarchy Diagram	
3.8	Deployment Diagram	
3.9	Table Design	
3.10	Data Dictionary	
3.11	Sample Input and Output Screens (Screens must have valid data. All reports must have at-least 5 valid records.)	
4	CHAPTER 4: CODING Sample code	
5	CHAPTER 5: Testing	
5.1	Test Cases/Test Scripts	
6	CHAPTER 6: LIMITATIONS OF SYSTEM	
7	CHAPTER 7: PROPOSED ENHANCEMENTS	
8	CHAPTER 8: CONCLUSION	
9	CHAPTER 9: BIBLIOGRAPHY	

CHAPTER 1: INTRODUCTION

1.1 Abstract

RentAToy, an Android application dedicated to enhancing the parenting experience, takes the hassle out of toy rentals for children. Its primary goal is to offer a hassle-free platform for parents and guardians, providing them with a seamless way to discover, choose, and rent toys for their little ones. The user-friendly interface ensures a smooth navigation experience, while the extensive array of toy options spans across various age groups and interests.

This innovative app goes beyond mere convenience; it's a one-stop solution for cost-effective toy rentals. By opting for RentAToy, families can enjoy an affordable alternative to purchasing toys outright. This not only promotes financial savings but also aligns with a sustainable approach, contributing to a more environmentally conscious parenting choice. With RentAToy, the joy of playtime becomes not only accessible but also economically savvy for families of all kinds.

1.2 Existing System and Need for System

Currently, there is no centralized platform or mobile application dedicated to toy rentals for children. Parents often have to rely on physical toy rental stores, which may have limited selections and inconvenient operating hours. The lack of an efficient system for toy rentals leads to difficulties for parents in providing a variety of toys for their children.

Right now, there's no one-stop app or place where parents can easily rent toys for their kids. They often have to rely on traditional toy rental stores, which can be a bit inconvenient—they might not have many options, and their opening hours might not match busy parent schedules.

1.3 Scope of System

- The scope of the Toy Rental App project includes the following:
- Development of a user-friendly Android mobile application.
- Integration with a database of toys available for rent.
- User registration and authentication system.
- A secure payment gateway for transactions.
- User-friendly interface for browsing and selecting toys.
- Booking and reservation system.
- Toy delivery and pickup logistics.

1.4 Operating Environment Hardware and Software

Hardware Requirements

- Android Device

Software Requirement

- Operating System: Android

1.5 Brief Description of Technology used

- Front End: XML,Kotlin,
- Back End: Firebase
- Database: Firebase

CHAPTER 2: PROPOSED SYSTEM

2.1 Feasibility Study

Feasibility Study: The RentAToy Project

1. Executive Summary: RentAToy aims to revolutionize the way parents access toys for their children by providing a user-friendly platform for renting toys. This feasibility study assesses the viability of the RentAToy project.

2. Business Concept: RentAToy addresses the gap in the market for a centralized, mobile application dedicated to toy rentals. The platform offers convenience, a diverse selection of toys, and a cost-effective alternative to traditional purchasing.

3. Market Analysis:

a. **Market Demand:** There is a growing demand for convenient and affordable options for accessing a variety of toys for children.

b. **Target Audience:** Parents and guardians looking for hassle-free access to a wide range of toys for different age groups.

Conclusion: The feasibility study indicates that RentAToy is a viable and potentially successful venture. The market demand, coupled with the technical, operational, and financial feasibility, positions RentAToy as a promising solution in the toy rental industry.

This study provides a solid foundation for the development and implementation of RentAToy, presenting a compelling case for investors and stakeholders.

2.2 Objectives of the proposed system

The main objectives of the RentAToy project are as follows:

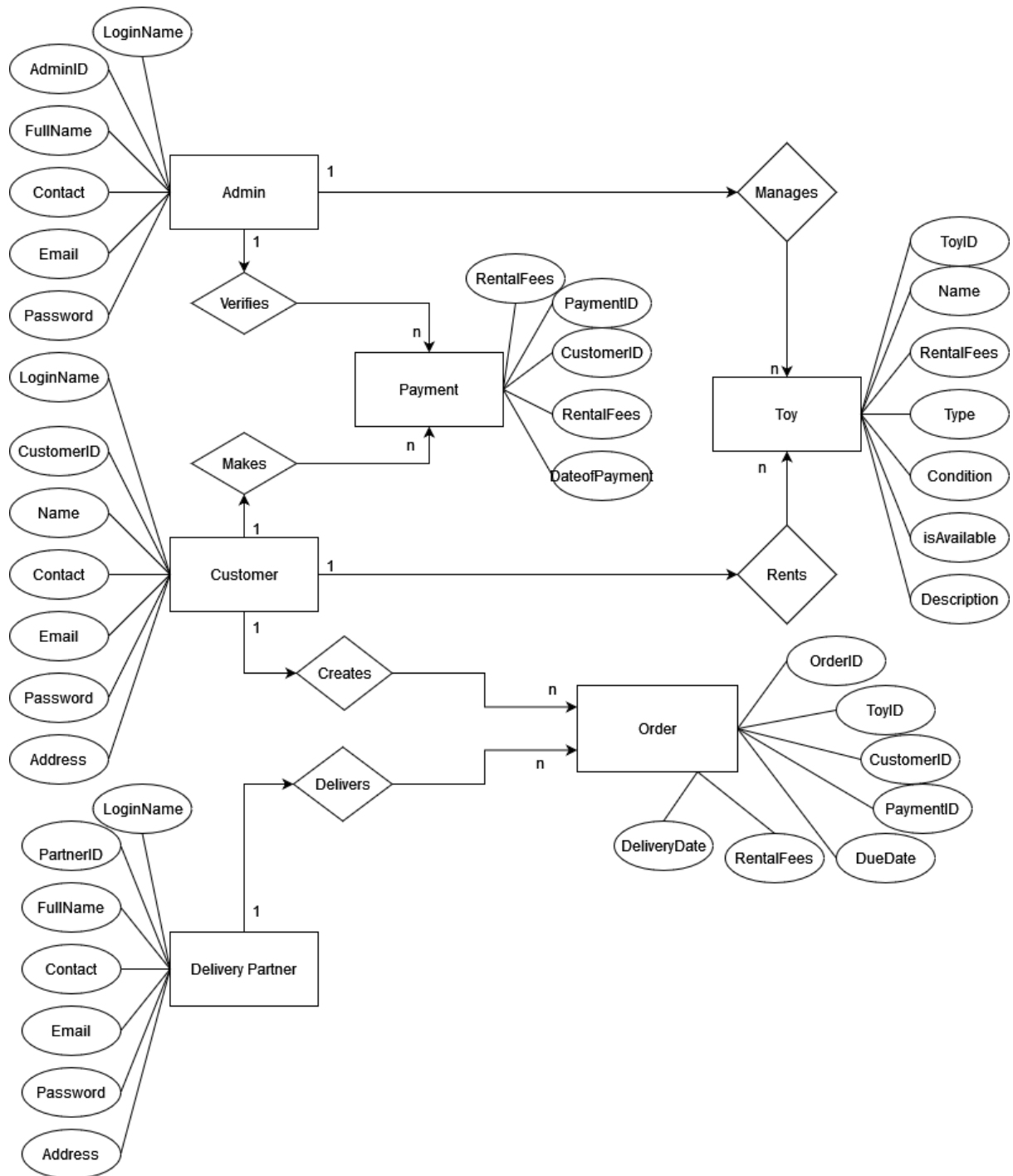
- The primary objectives of the Toy Rental App are as follows:
- Provide a user-friendly and intuitive mobile application for toy rentals.
- Create a secure and reliable platform for toy owners to list their toys for rent.
- Offer a seamless payment process for users.
- Ensure efficient toy delivery and pickup logistics.
- Enhance the overall experience for both toy renters and toy owners.

2.3 Users of the system

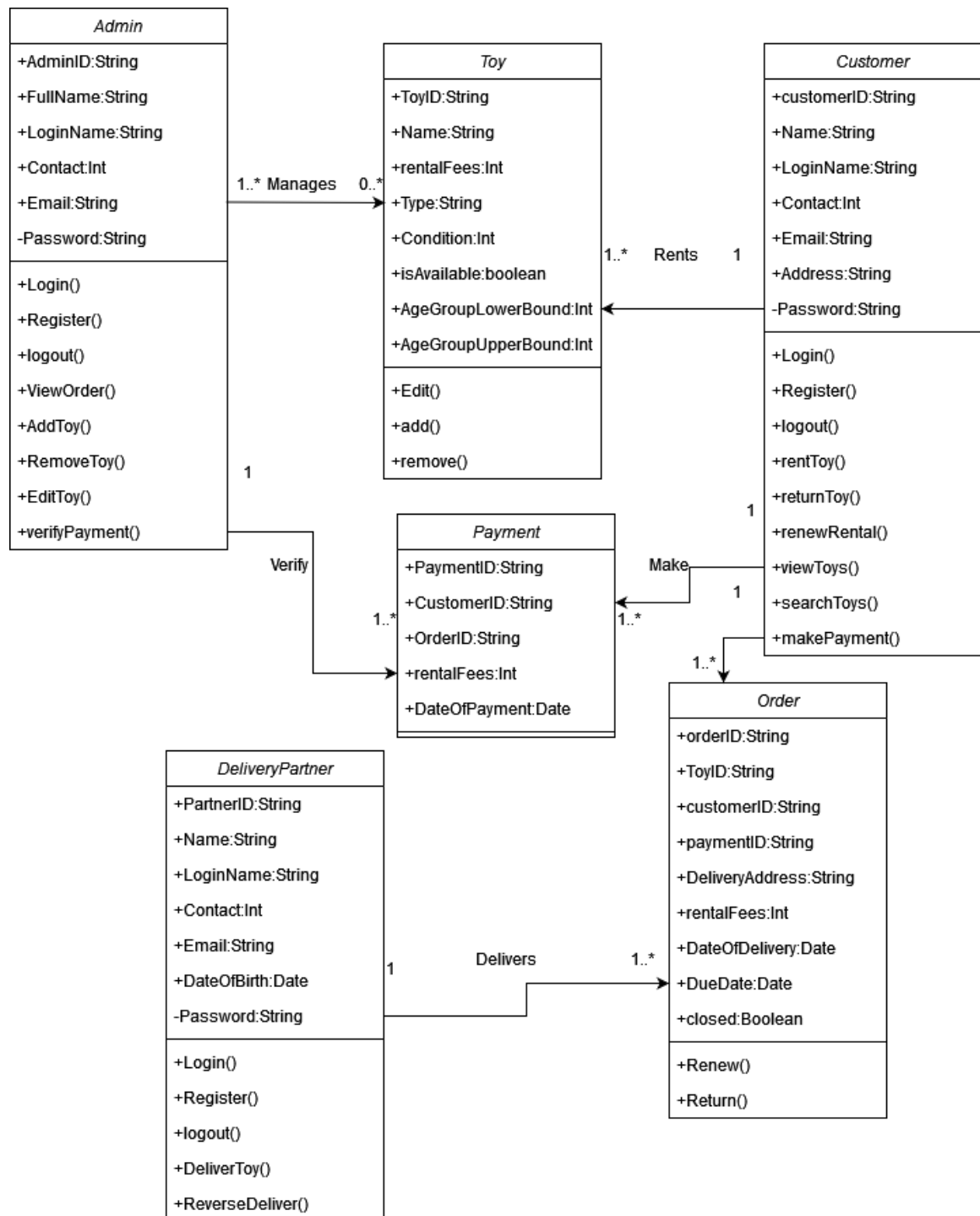
- Admin
The Admin manages the toys
The Admin can:
 1. Register
 2. Login
 3. Logout
 4. Manage Toys
 5. Manage Users
 6. Manage Orders
 7. Manage Payments
- Customer
The Customer can rent toys
The Customer can:
 1. Register
 2. Login
 3. Logout
 4. Manage rented toys
 5. Make payments
- Delivery Partner
The Delivery Partner delivers the toys
The Delivery Partner can:
 1. Register
 2. Login
 3. Logout
 4. Deliver Toys

CHAPTER 3: ANALYSIS AND DESIGN

3.1 Entity Relationship Diagram (ERD)

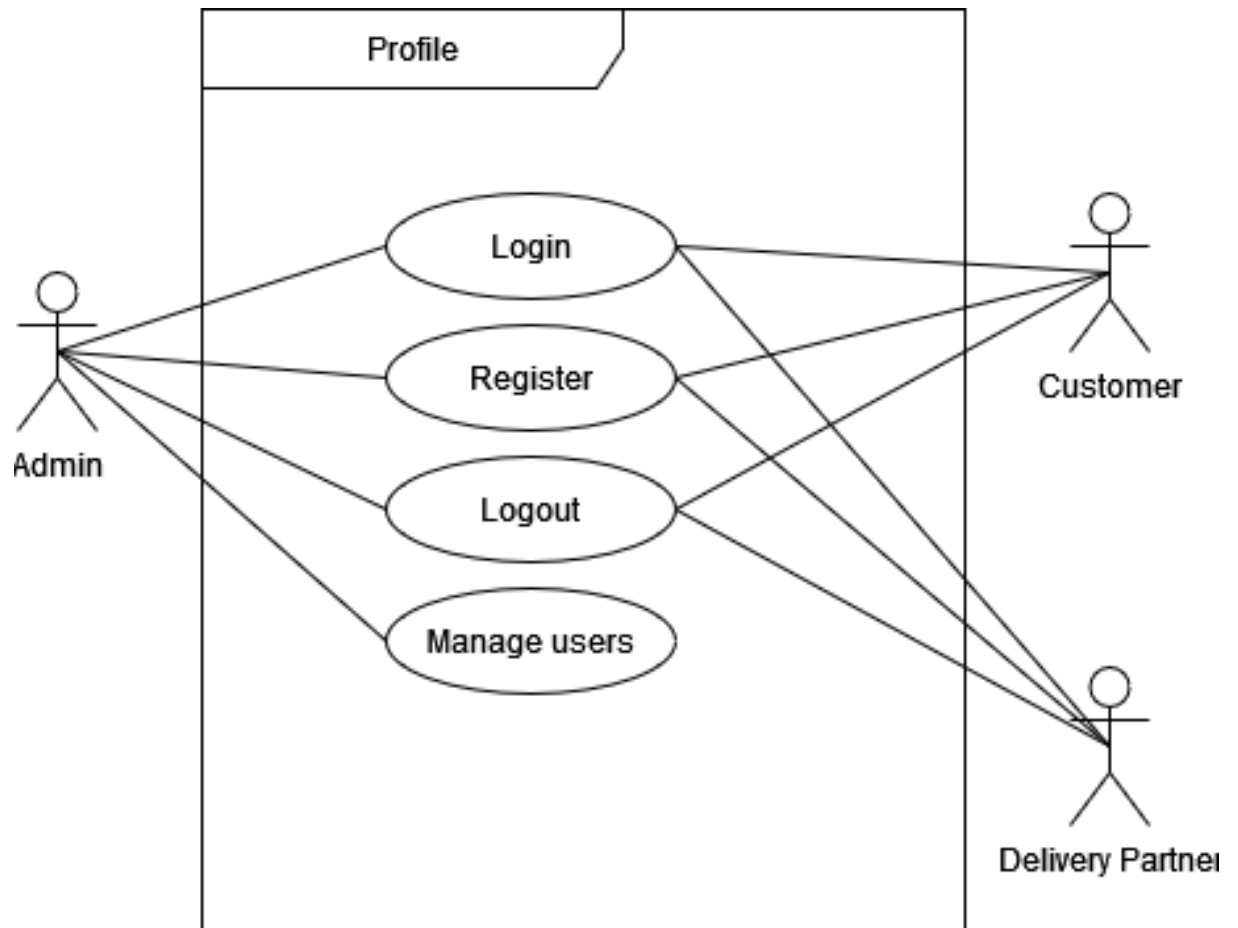


3.2 Class Diagram

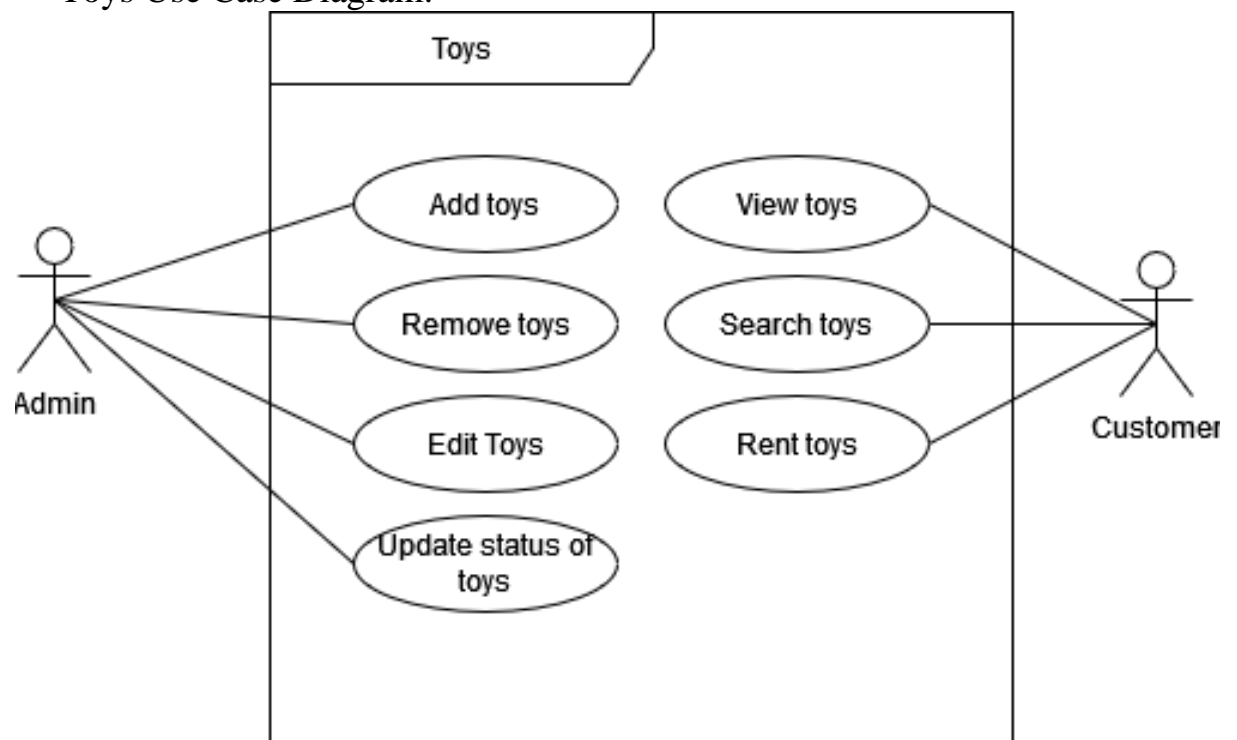


3.3 Use Case Diagrams

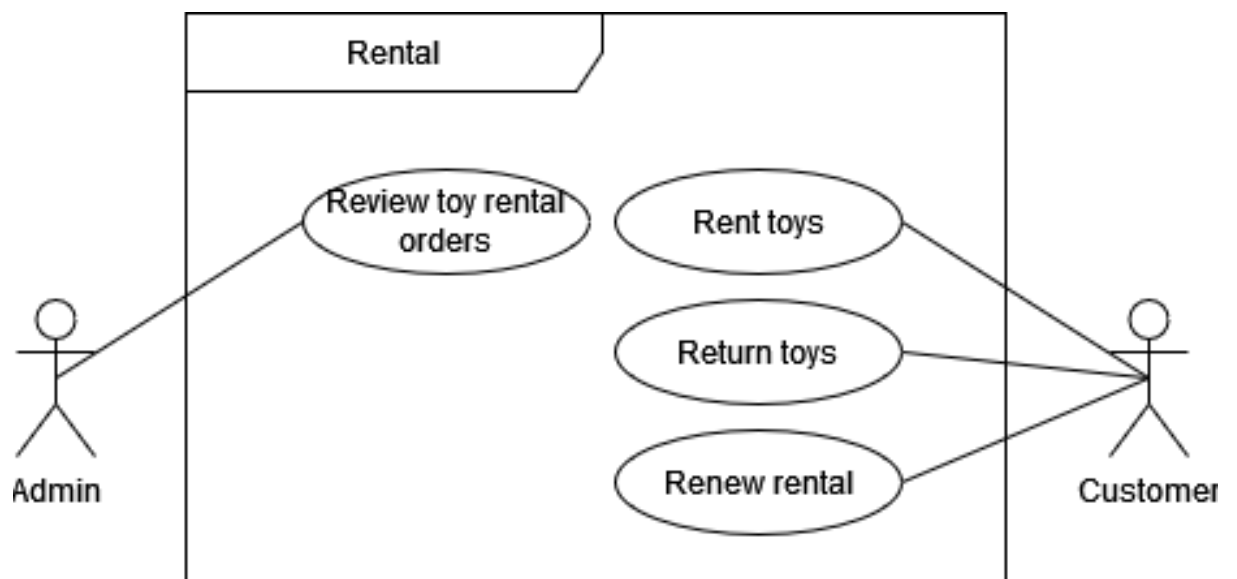
Profile Use Case Diagram:



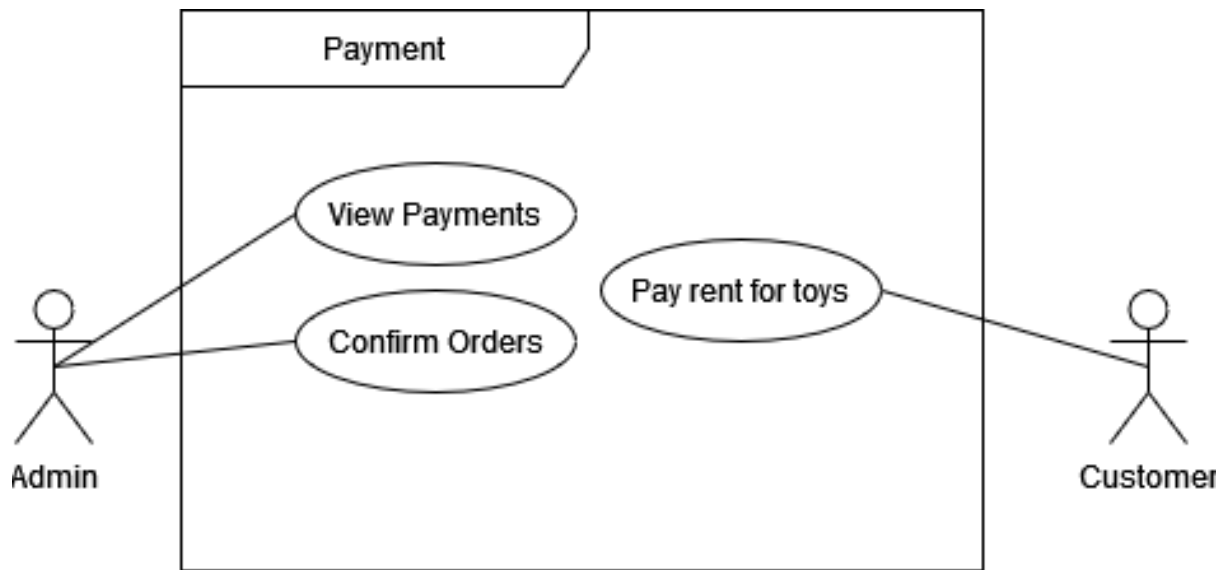
Toys Use Case Diagram:



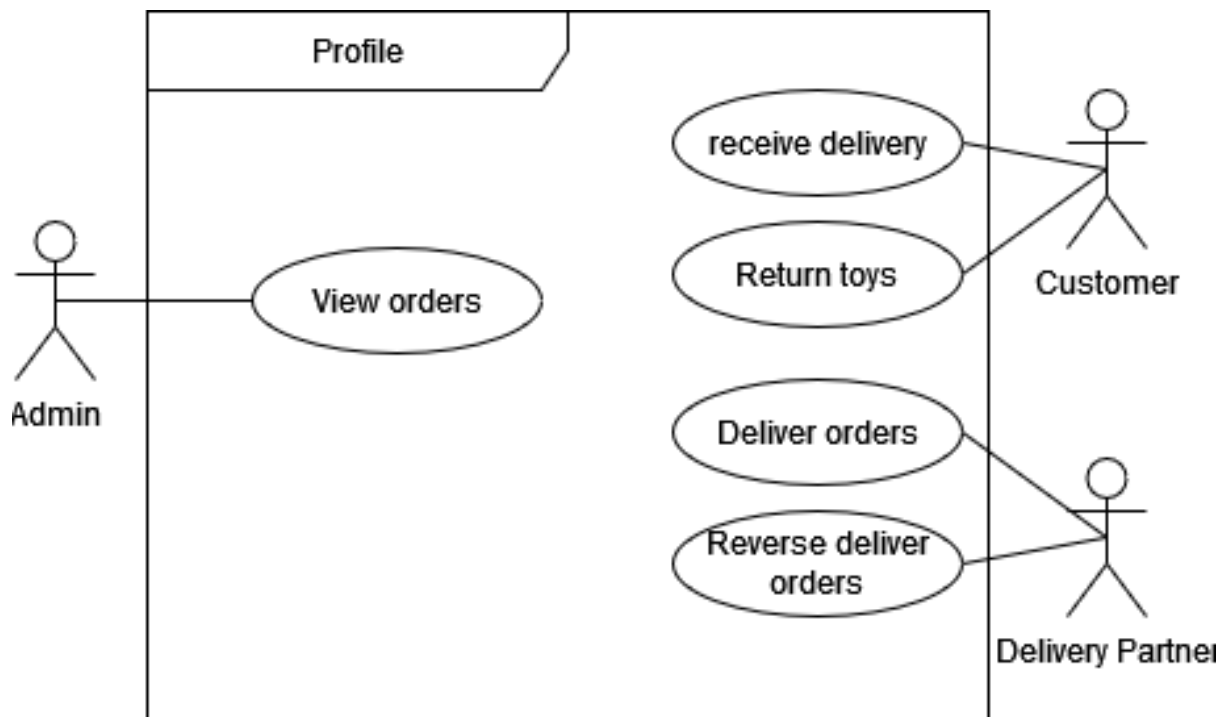
Rental Use Case Diagram:



Payment Use Case Diagram:

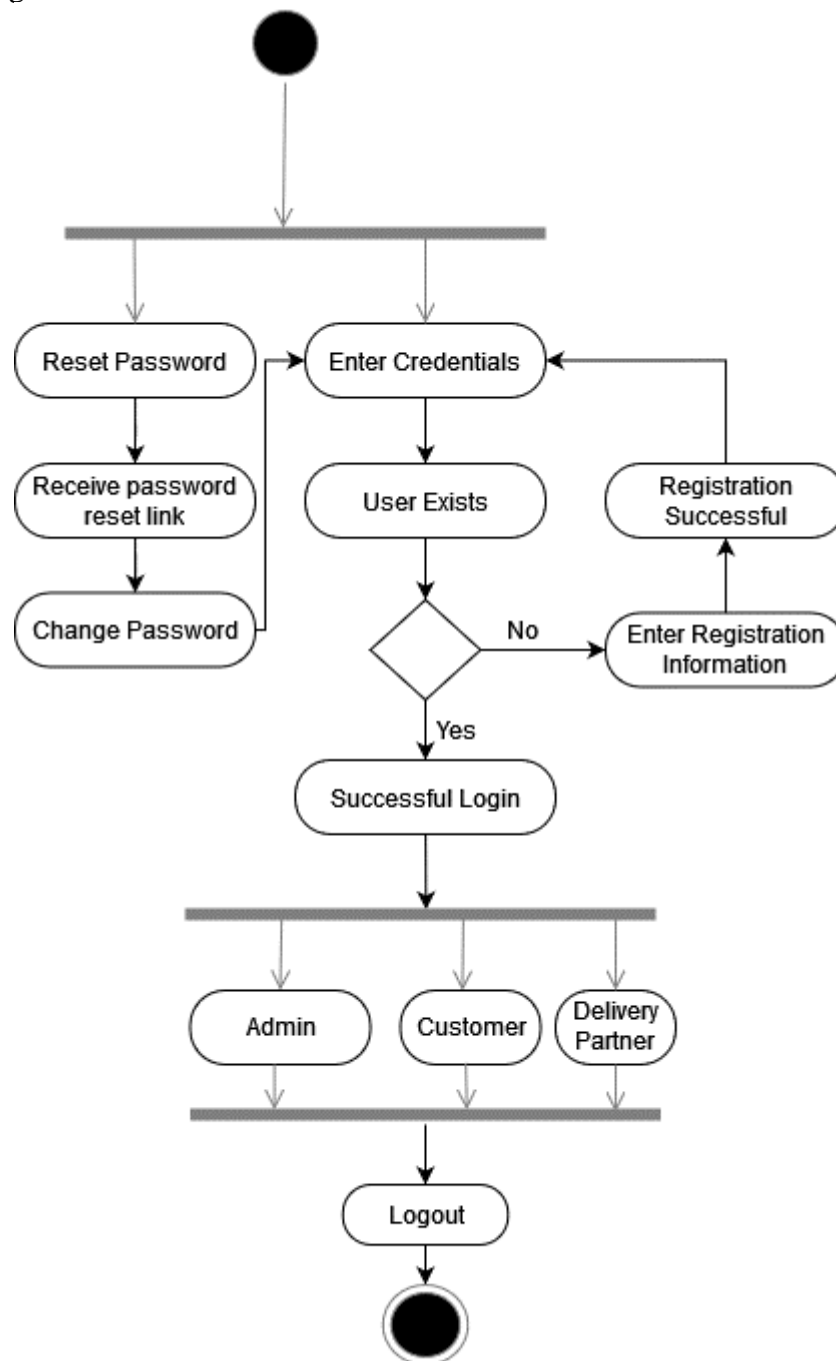


Delivery Use Case Diagram:

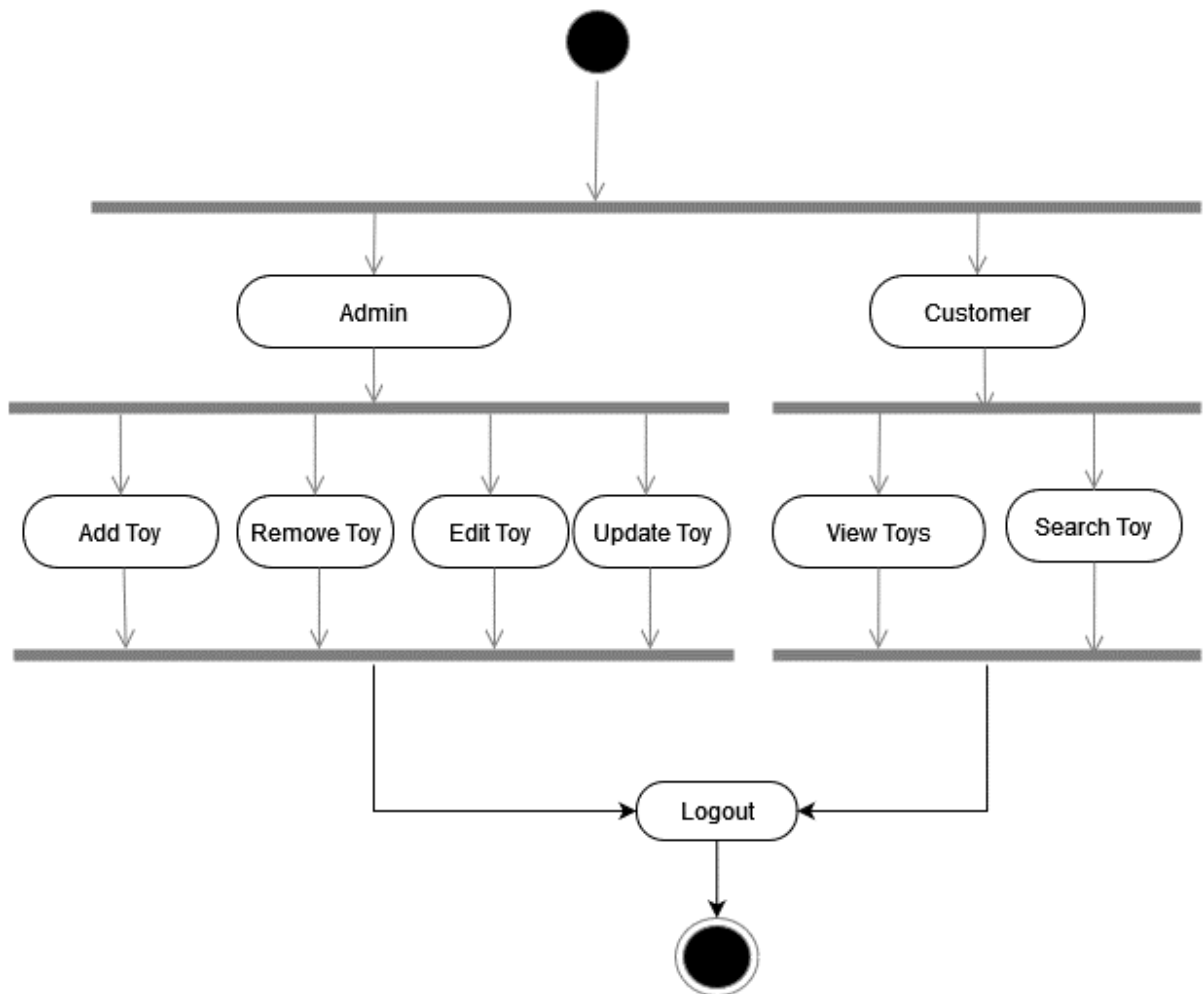


3.4 Activity Diagram

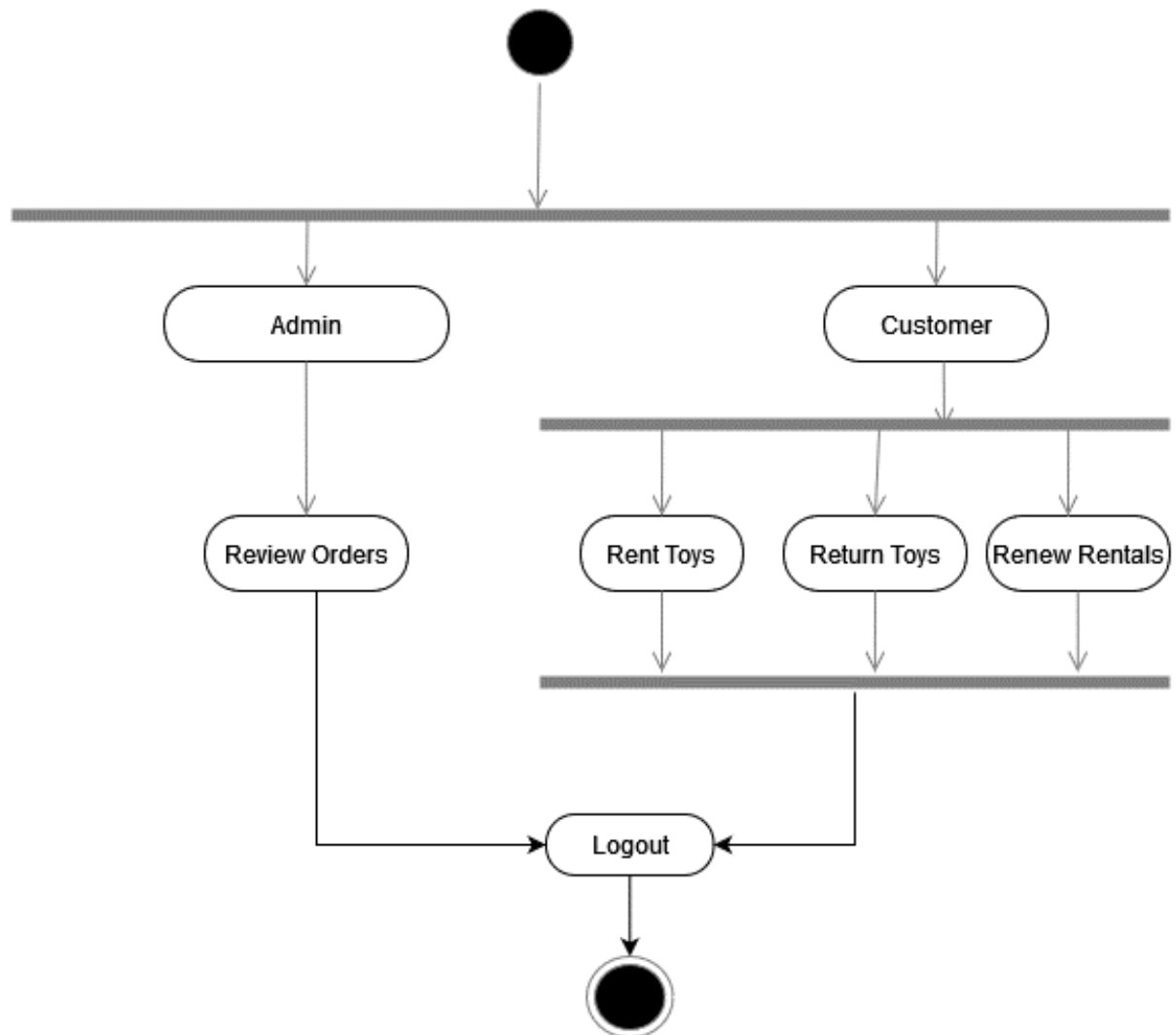
Profile Activity Diagram:



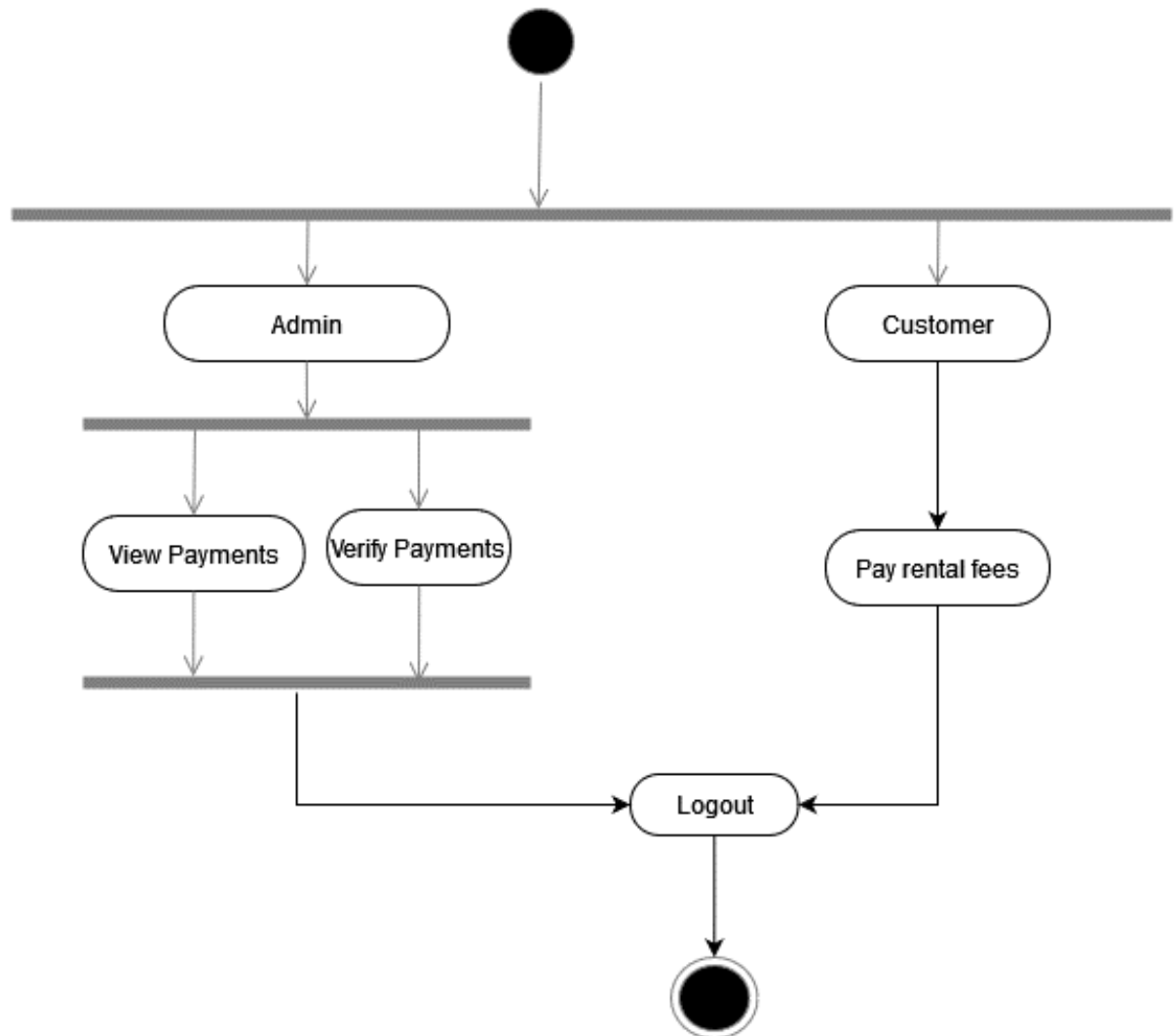
Toys Activity Diagram:



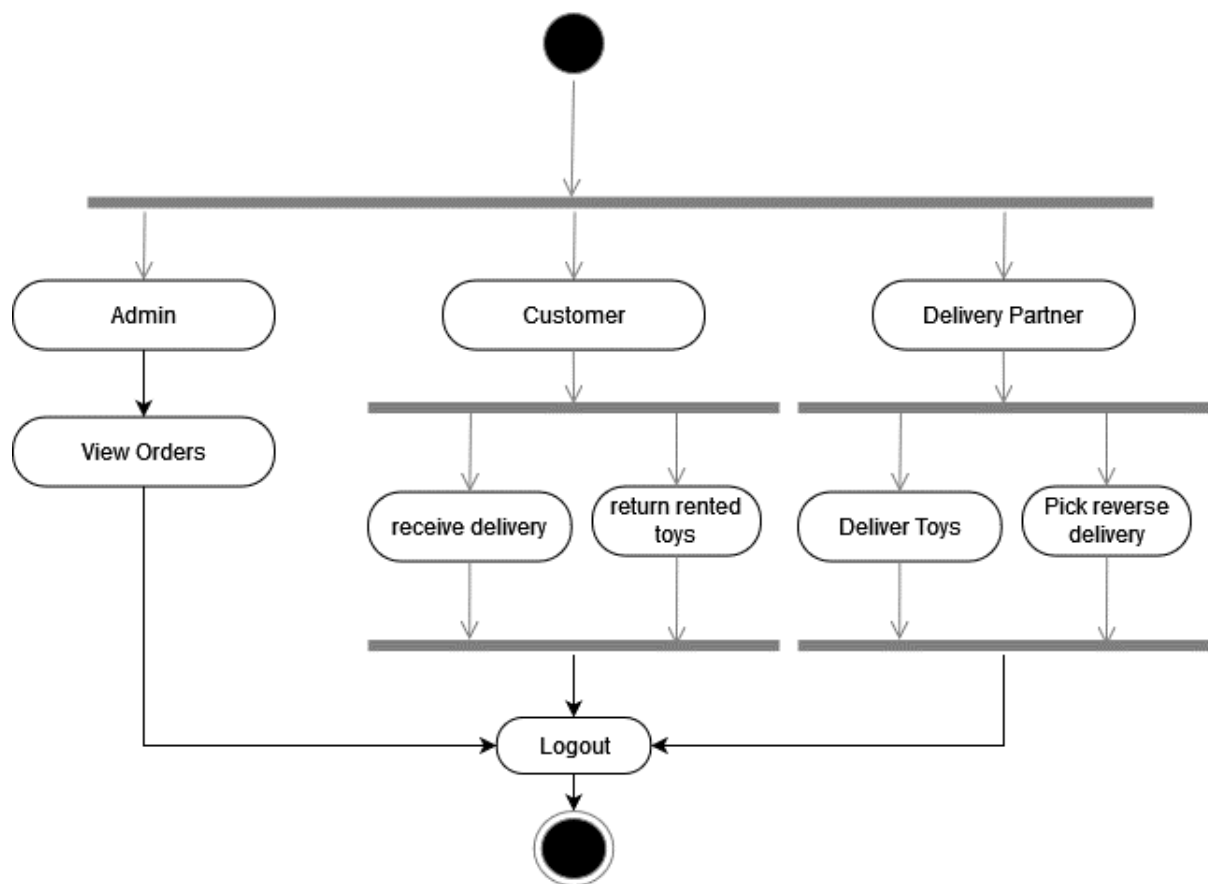
Rental Activity Diagram:



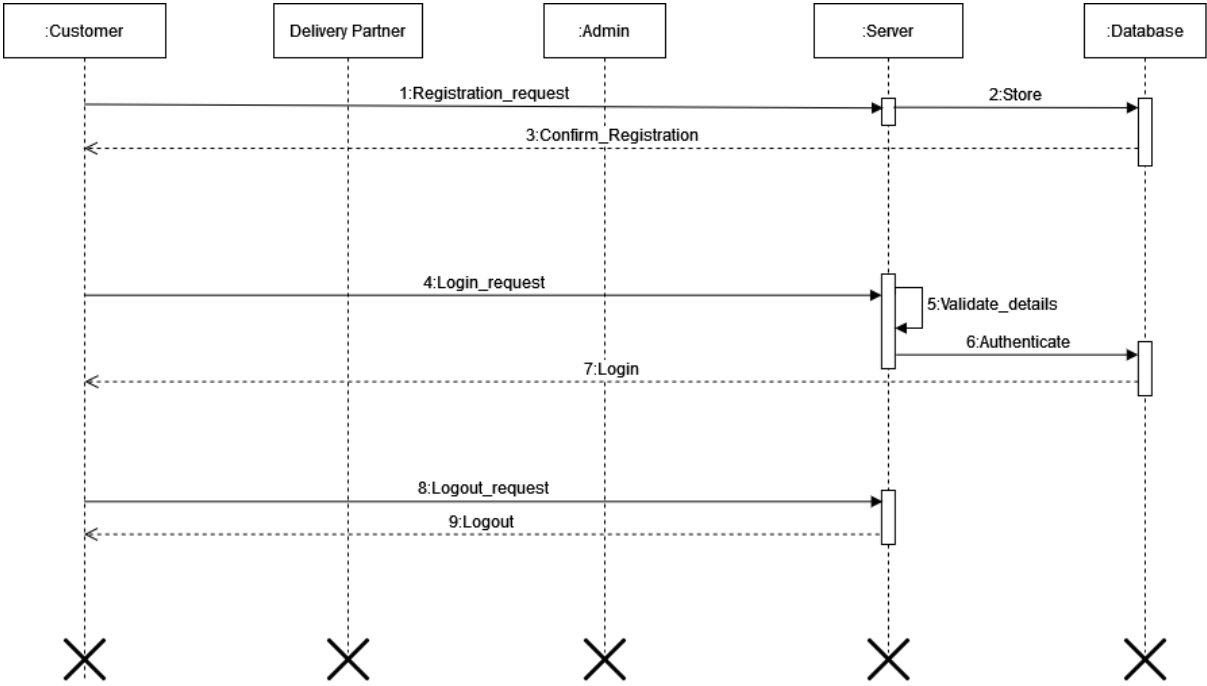
Payment Activity Diagram:



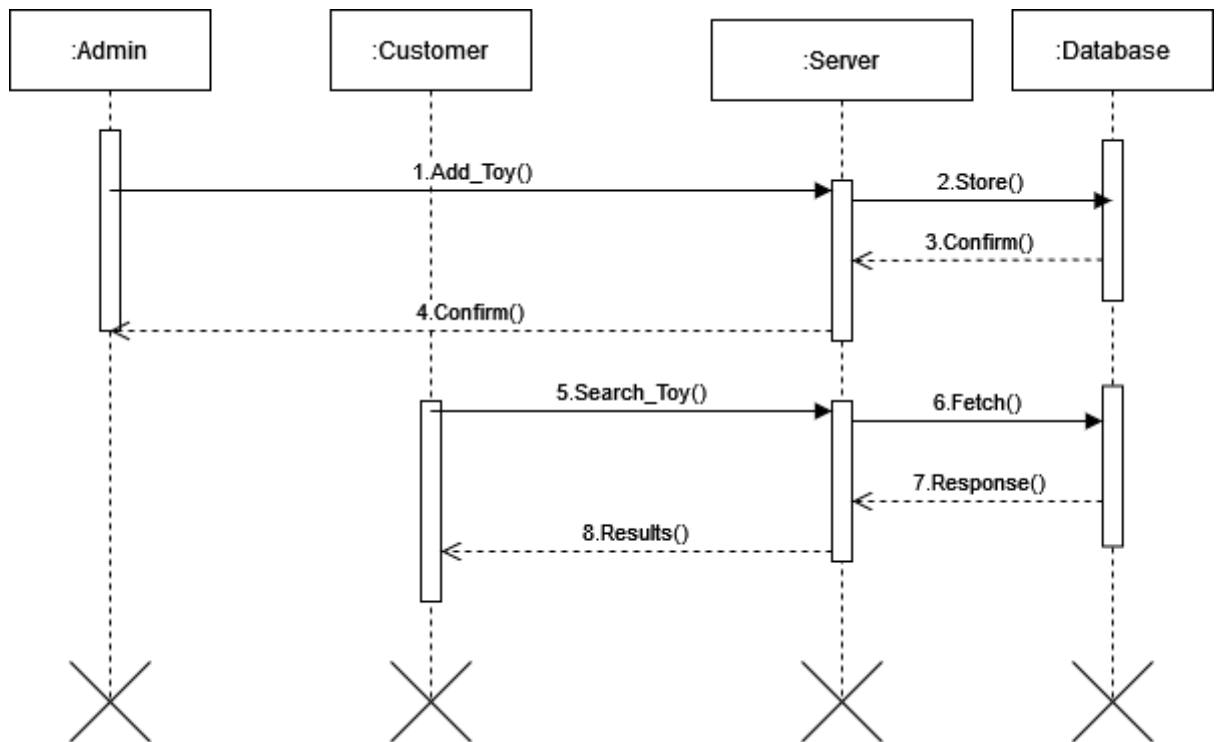
Delivery Activity Diagram:



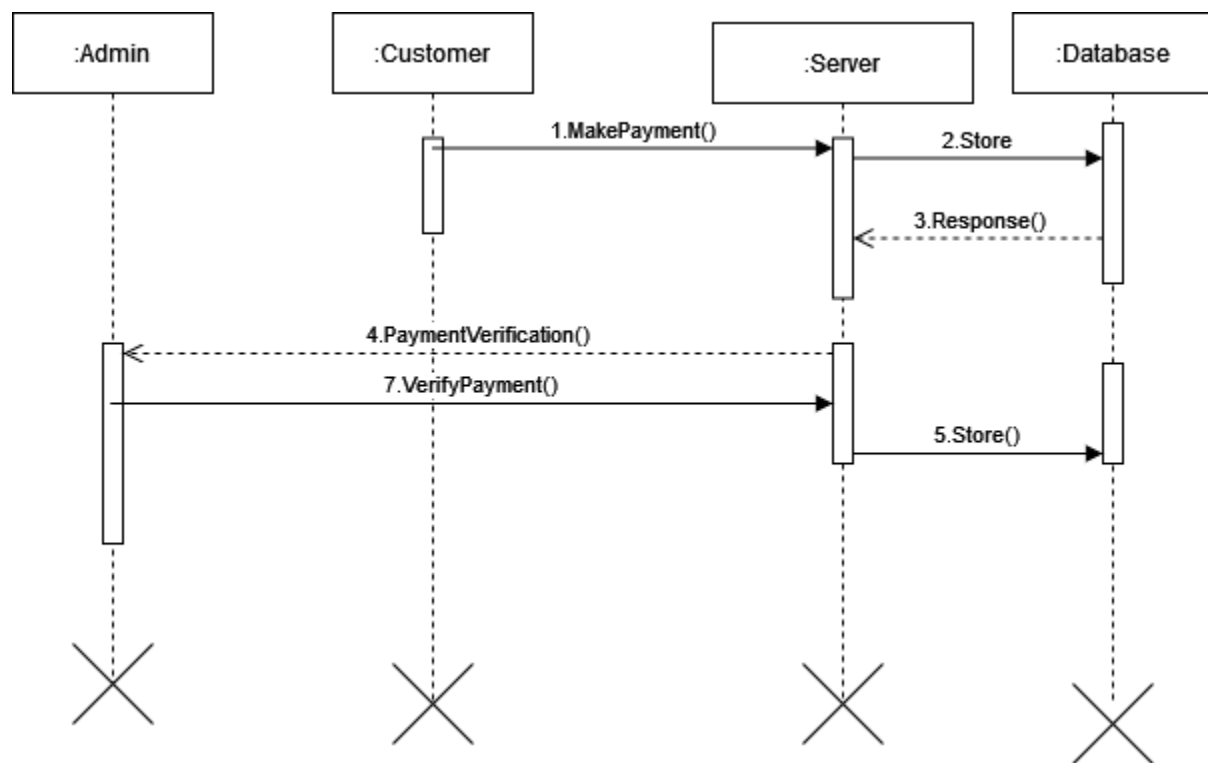
3.5 Sequence Diagram
Profile Sequence Diagram:



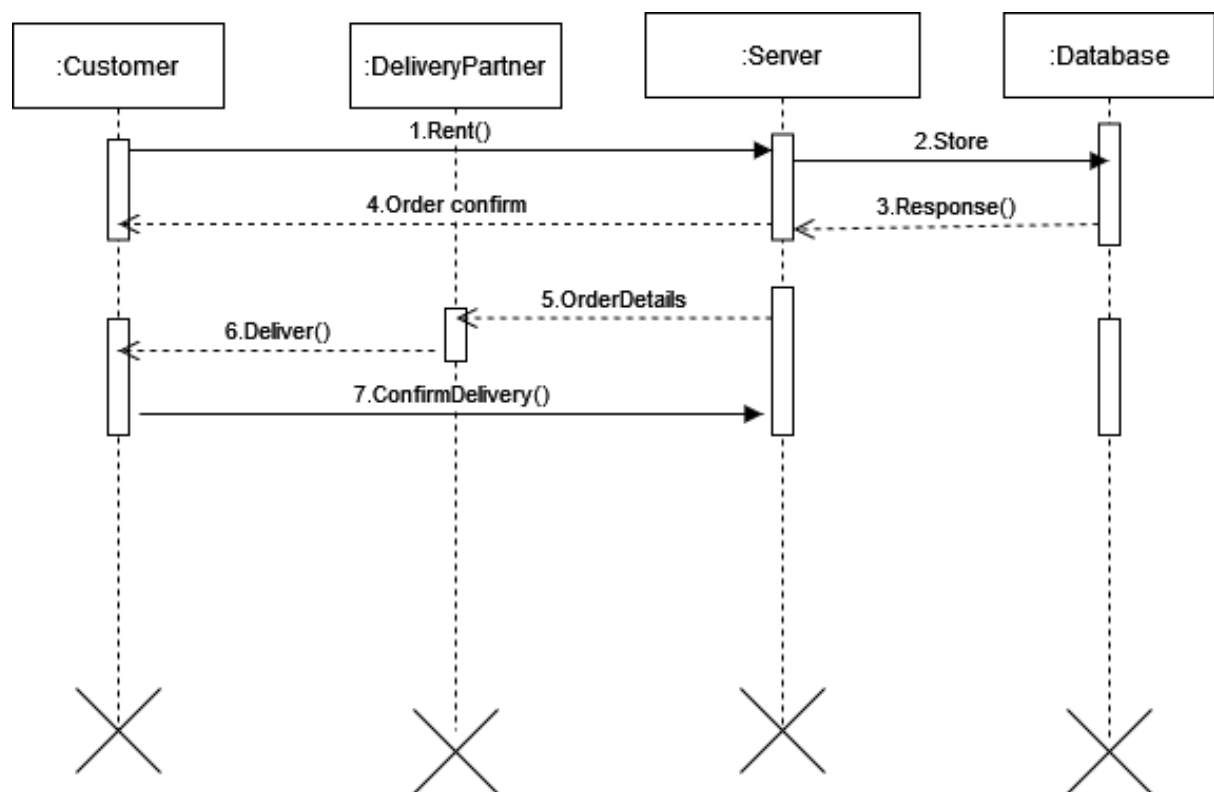
Toys Sequence Diagram:



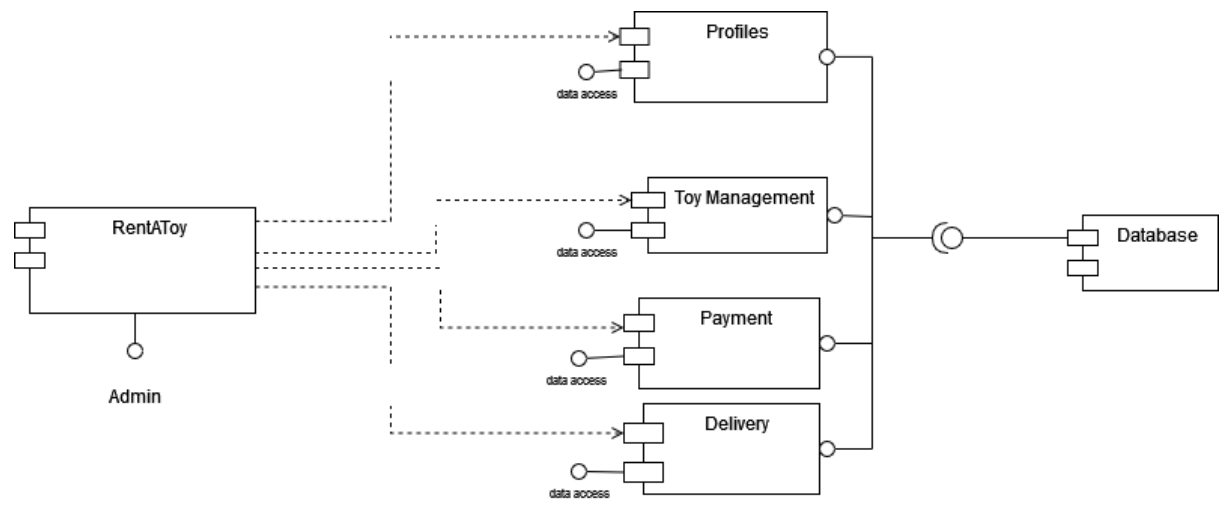
Payment Sequence Diagram:



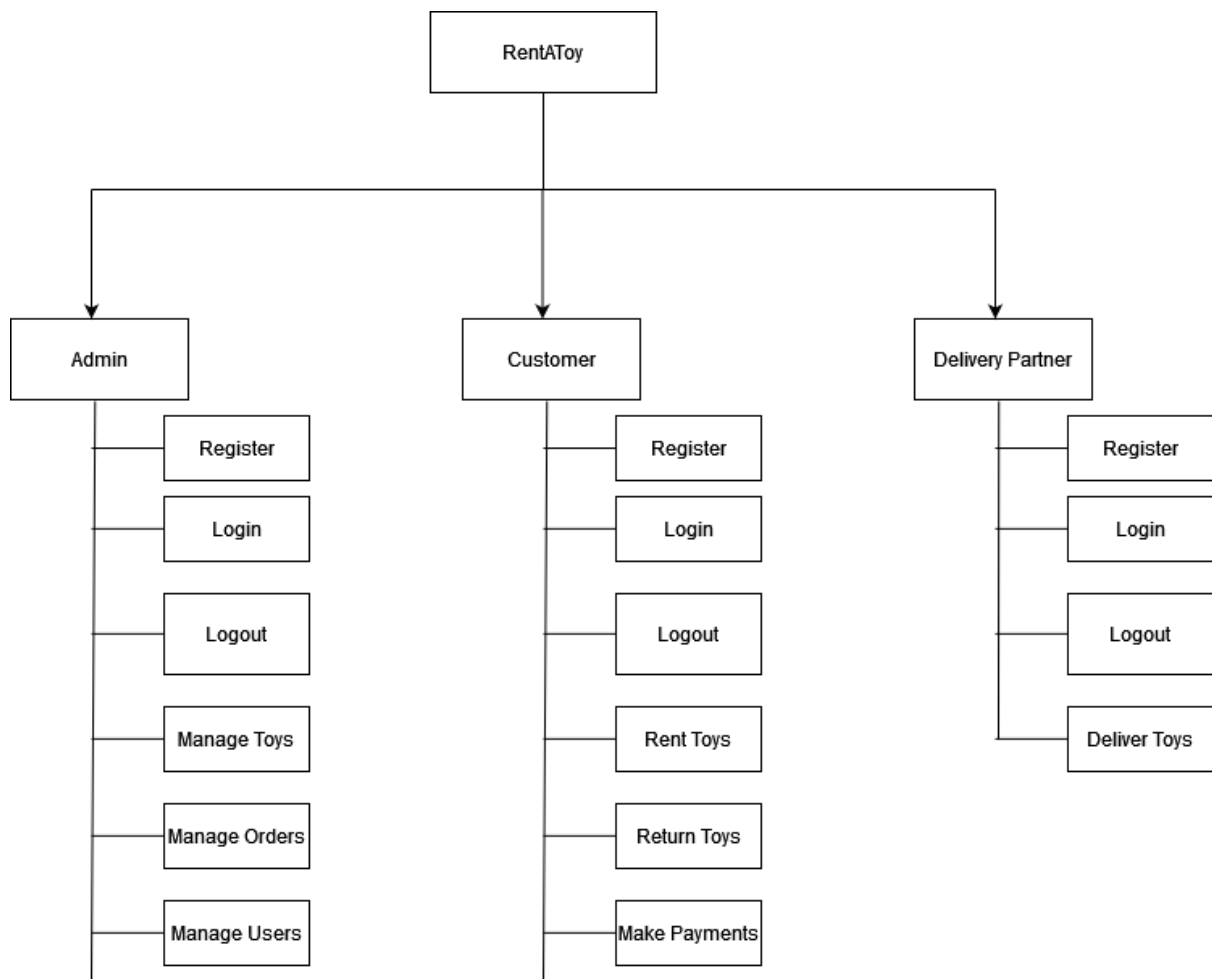
Delivery Sequence Diagram:



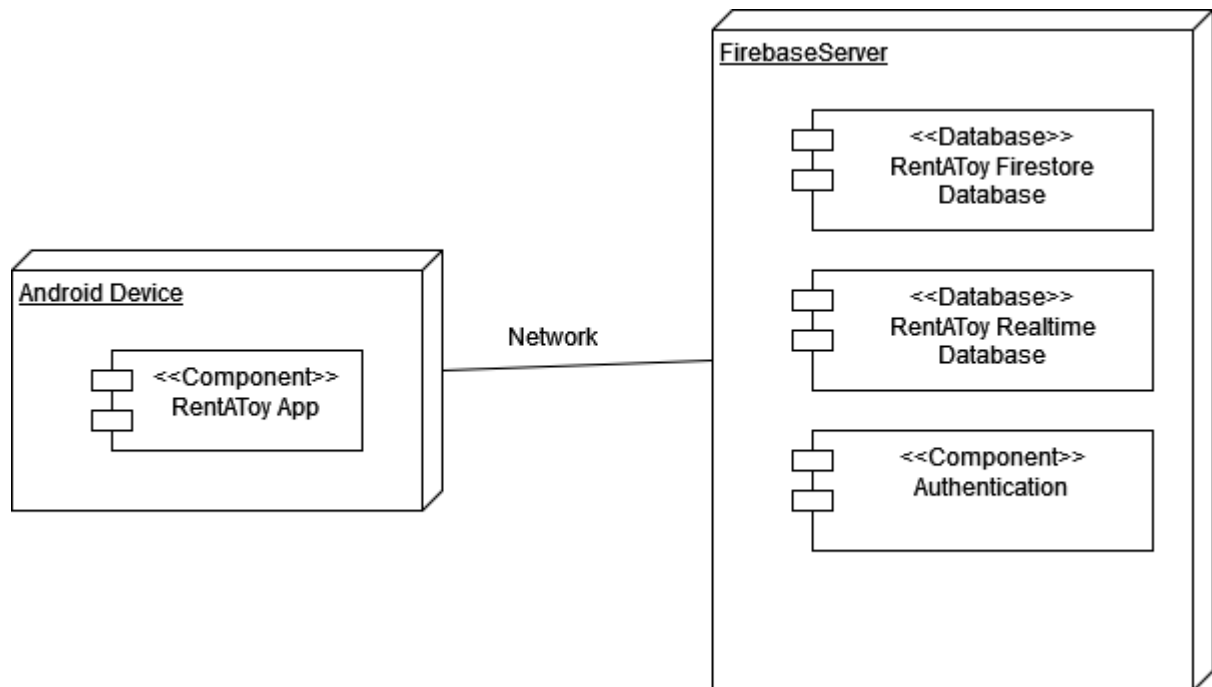
3.6 Component Diagram



3.7 Module and Hierarchy Diagram



3.8 Deployment Diagram



3.9 Table Design

Table Name: Admin

Sr.No	Column Name	Data Type	Constraints
1	AdminID	INT	PRIMARY KEY, AUTO_INCREMENT
2	FullName	VARCHAR(50)	NOT NULL
3	LoginName	VARCHAR(20)	UNIQUE, NOT NULL
4	Contact	VARCHAR(15)	
5	Email	VARCHAR(50)	UNIQUE, NOT NULL
6	Password	VARCHAR(20)	NOT NULL

Table Name: Toy

Sr.No	Column Name	Data Type	Constraints
1	ToyID	INT	PRIMARY KEY, AUTO_INCREMENT
2	ToyName	VARCHAR(50)	NOT NULL
3	ToyRentalFees	DECIMAL(10,2)	NOT NULL
4	ToyCategory	VARCHAR(20)	
5	isAvailable	BOOLEAN	
6	ToyAgeGroup	VARCHAR(10)	

Table Name: Customer

Sr.No	Column Name	Data Type	Constraints
1	CustomerID	INT	PRIMARY KEY, AUTO_INCREMENT
2	FullName	VARCHAR(50)	NOT NULL
3	LoginName	VARCHAR(20)	UNIQUE, NOT NULL
4	Contact	VARCHAR(15)	
5	Email	VARCHAR(50)	UNIQUE, NOT NULL
6	Address	VARCHAR(100)	
7	Password	VARCHAR(20)	NOT NULL

Table Name: Payment

Sr.No	Column Name	Data Type	Constraints
1	PaymentID	INT	PRIMARY KEY, AUTO_INCREMENT
2	OrderID	INT	FOREIGN KEY (OrderID)
3	rentalFees	DECIMAL(10,2)	NOT NULL

Sr.No	Column Name	Data Type	Constraints
4	DateOfPayment	DATE	NOT NULL

Table Name: Order

Sr.No	Column Name	Data Type	Constraints
1	OrderID	INT	PRIMARY KEY, AUTO_INCREMENT
2	ToyID	INT	FOREIGN KEY (ToyID)
3	CustomerID	INT	FOREIGN KEY (CustomerID)
4	DeliveryAddress	VARCHAR(100)	NOT NULL
5	RentalFees	DECIMAL(10,2)	NOT NULL
6	DueDate	DATE	
7	closed	BOOLEAN	DEFAULT false

Table Name: DeliveryPartner

Sr.No	Column Name	Data Type	Constraints
1	PartnerID	INT	PRIMARY KEY, AUTO_INCREMENT
2	Name	VARCHAR(50)	NOT NULL
3	LoginName	VARCHAR(20)	UNIQUE, NOT NULL
4	Contact	VARCHAR(15)	
5	Email	VARCHAR(50)	UNIQUE, NOT NULL
6	DateOfBirth	DATE	
7	Password	VARCHAR(20)	NOT NULL

3.10 Data Dictionary

Table: Admin

Column Name	Data Type	Constraints	Description
AdminID	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each admin.
FullName	VARCHAR(50)	NOT NULL	Full name of the admin.
LoginName	VARCHAR(20)	UNIQUE, NOT NULL	Unique login name for the admin.
Contact	VARCHAR(15)		Contact number of the admin.
Email	VARCHAR(50)	UNIQUE, NOT NULL	Email address of the admin.
Password	VARCHAR(20)	NOT NULL	Password for admin authentication.

Table: Toy

Column Name	Data Type	Constraints	Description
ToyID	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each toy.
ToyName	VARCHAR(50)	NOT NULL	Name of the toy.
ToyRentalFees	DECIMAL(10,2)	NOT NULL	Rental fees associated with the toy.
ToyCategory	VARCHAR(20)		Category to which the toy belongs.
isAvailable	BOOLEAN		Indicates whether the toy is available for rent.
ToyAgeGroup	VARCHAR(10)		Age group for which the toy is suitable.

Table: Customer

Column Name	Data Type	Constraints	Description
CustomerID	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each customer.
FullName	VARCHAR(50)	NOT NULL	Full name of the customer.
LoginName	VARCHAR(20)	UNIQUE, NOT NULL	Unique login name for the customer.
Contact	VARCHAR(15)		Contact number of the customer.
Email	VARCHAR(50)	UNIQUE, NOT NULL	Email address of the customer.
Address	VARCHAR(100)		Address of the customer.
Password	VARCHAR(20)	NOT NULL	Password for customer authentication.

Table: Payment

Column Name	Data Type	Constraints	Description
PaymentID	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each payment.
OrderID	INT	FOREIGN KEY (OrderID)	Reference to the associated order.
rentalFees	DECIMAL(10,2)	NOT NULL	Rental fees associated with the payment.
DateOfPayment	DATE	NOT NULL	Date when the payment was made.

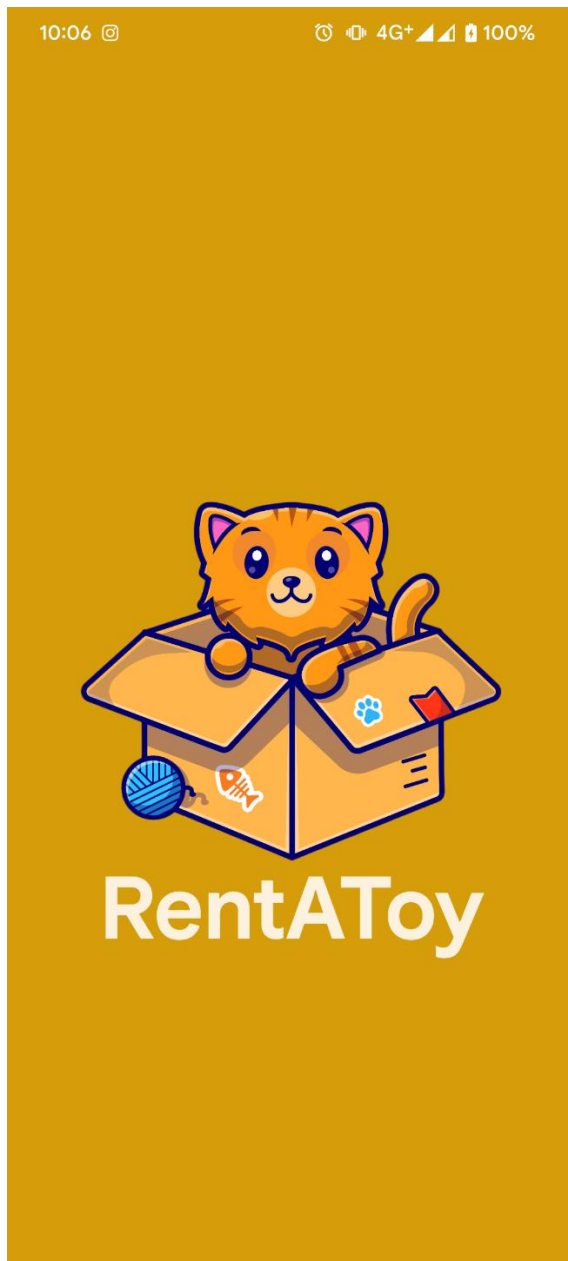
Table: Order

Column Name	Data Type	Constraints	Description
OrderID	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each order.
ToyID	INT	FOREIGN KEY (ToyID)	Reference to the toy in the order.
CustomerID	INT	FOREIGN KEY (CustomerID)	Reference to the customer placing the order.
DeliveryAddress	VARCHAR(100)	NOT NULL	Address where the order needs to be delivered.
RentalFees	DECIMAL(10,2)	NOT NULL	Rental fees associated with the order.
DueDate	DATE		Due date for returning the rented toy.
closed	BOOLEAN	DEFAULT false	Indicates whether the order is closed.

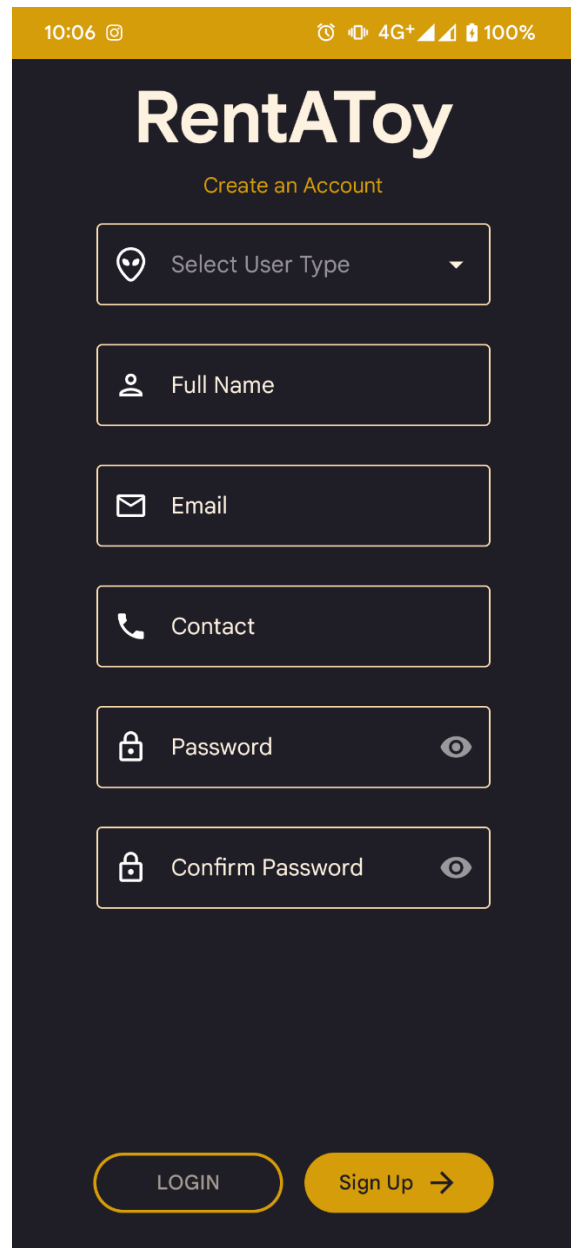
Table: DeliveryPartner

Column Name	Data Type	Constraints	Description
PartnerID	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each delivery partner.
Name	VARCHAR(50)	NOT NULL	Name of the delivery partner.
LoginName	VARCHAR(20)	UNIQUE, NOT NULL	Unique login name for the delivery partner.
Contact	VARCHAR(15)		Contact number of the delivery partner.
Email	VARCHAR(50)	UNIQUE, NOT NULL	Email address of the delivery partner.
DateOfBirth	DATE		Date of birth of the delivery partner.
Password	VARCHAR(20)	NOT NULL	Password for delivery partner authentication.

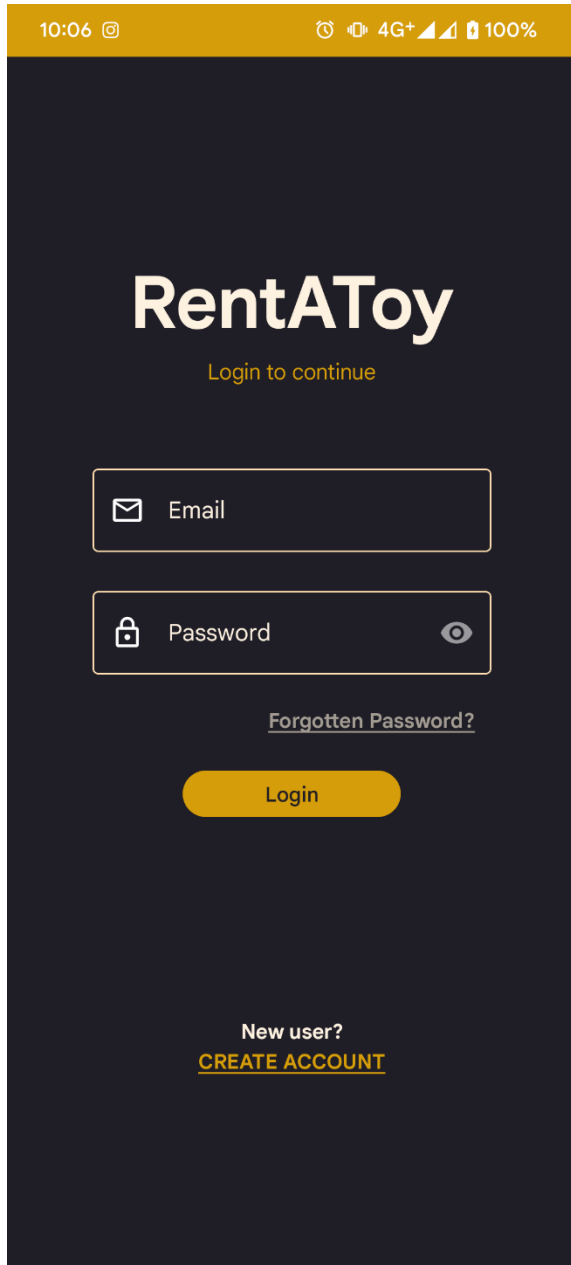
3.11 Sample Input and Output Screens



Splash Screen

The registration page has a dark blue background. At the top, a status bar shows the time as 10:06, signal strength, 4G+ connectivity, and 100% battery. Below the status bar, the "RentAToy" logo is displayed in white, followed by the text "Create an Account" in a smaller, orange font. The page contains several input fields, each with a white icon on the left and a white label: "Select User Type" (with a location pin icon and a dropdown arrow), "Full Name" (with a person icon), "Email" (with an envelope icon), "Contact" (with a telephone icon), "Password" (with a lock icon and a toggle eye icon), and "Confirm Password" (with a lock icon and a toggle eye icon). At the bottom, there are two buttons: a white "LOGIN" button and an orange "Sign Up →" button.

Registration Page



10:06 4G+ 100%

RentAToy

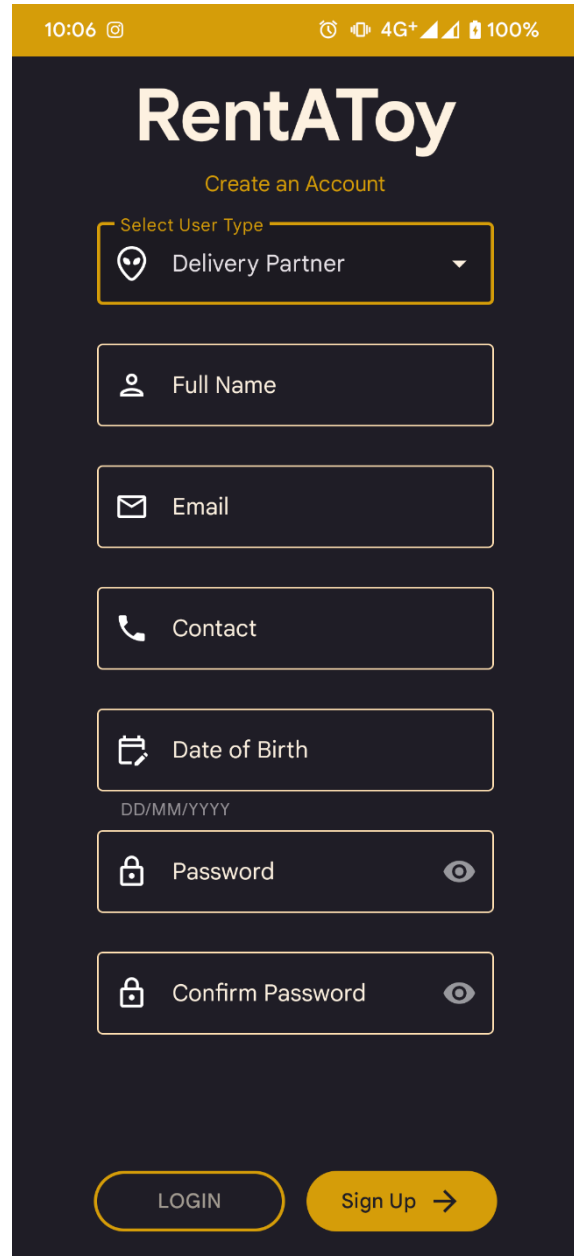
Login to continue

[Forgotten Password?](#)

Login

New user?
[CREATE ACCOUNT](#)

Login



10:06 4G+ 100%

RentAToy

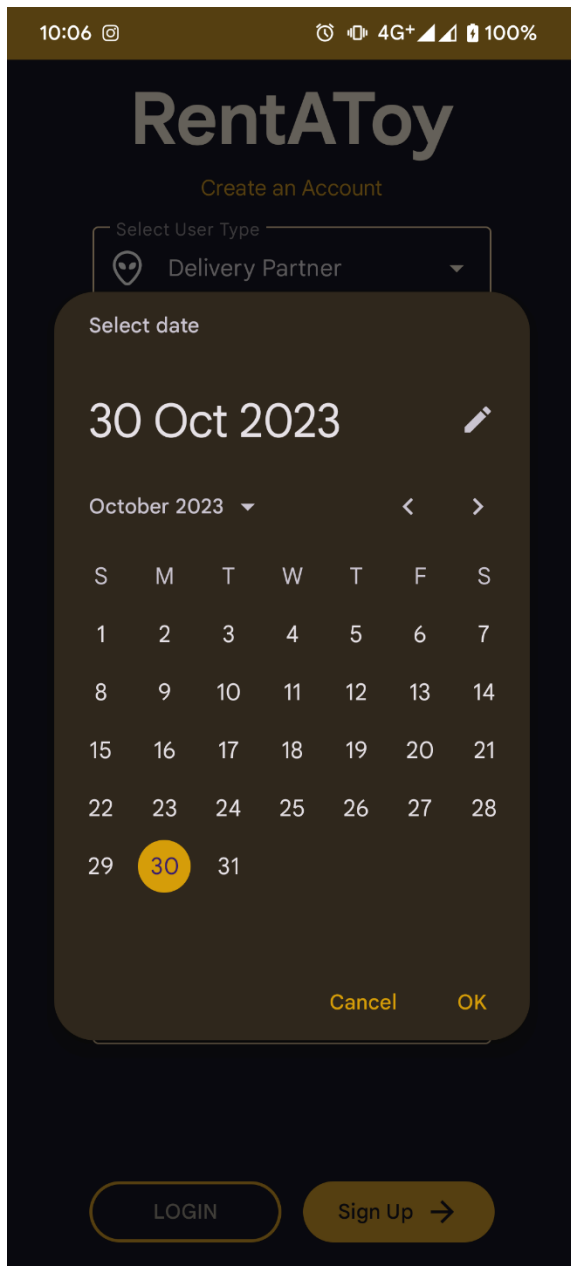
Create an Account

Select User Type
Delivery Partner

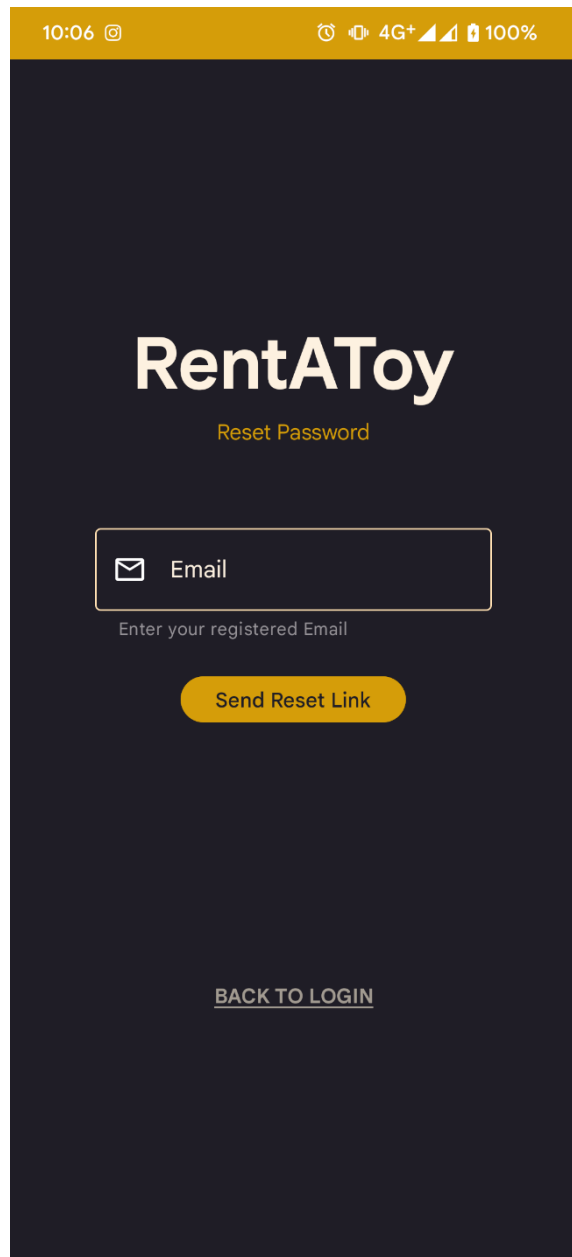
DD/MM/YYYY

LOGIN Sign Up →

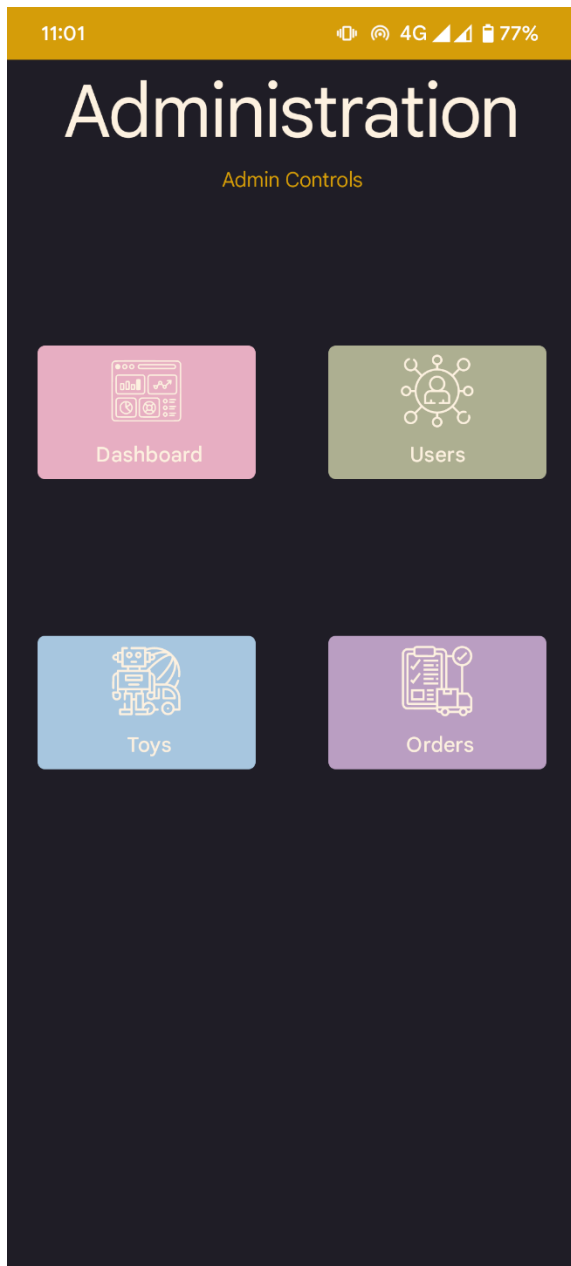
Registration Page Type Selection



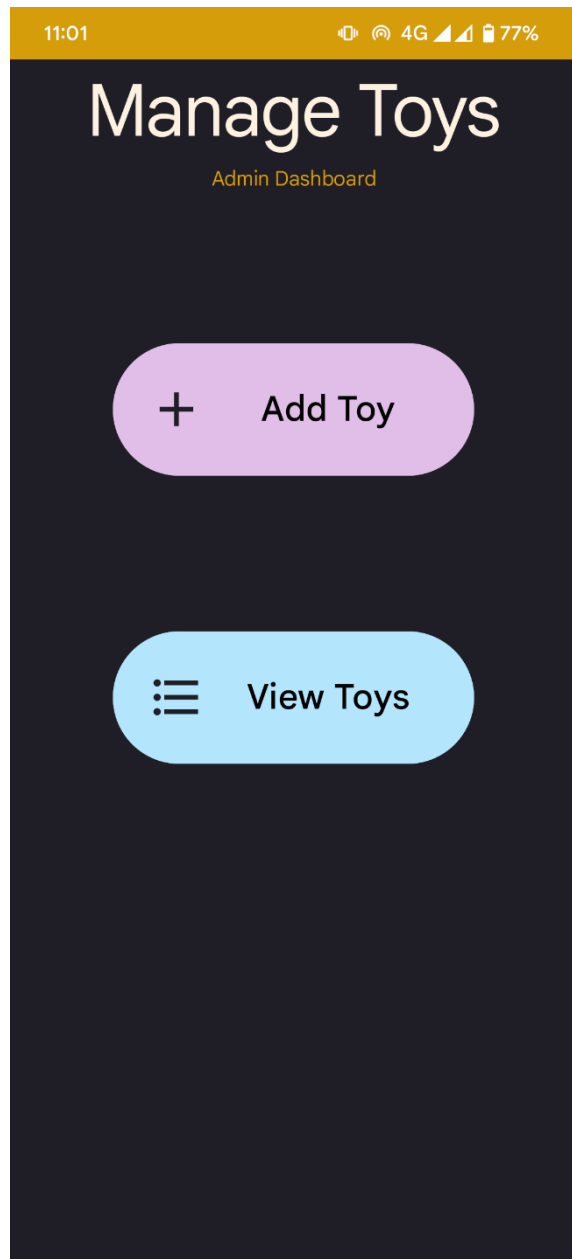
Date Picker



Reset Password



Admin Dashboard



Manage Toys

11:03

77%

Manage Toys

Add a toy

19

Toy Name

Remote Controlled Car

Toy Rental Fees

₹ 50

In rupees

Toy Category

Remote Controlled

Available

☒
True

True by default

Age Group

School-Age [6-12 years]

Toy Description

A toy car controlled using a remote control

Image Added

Add Toy

Add toy

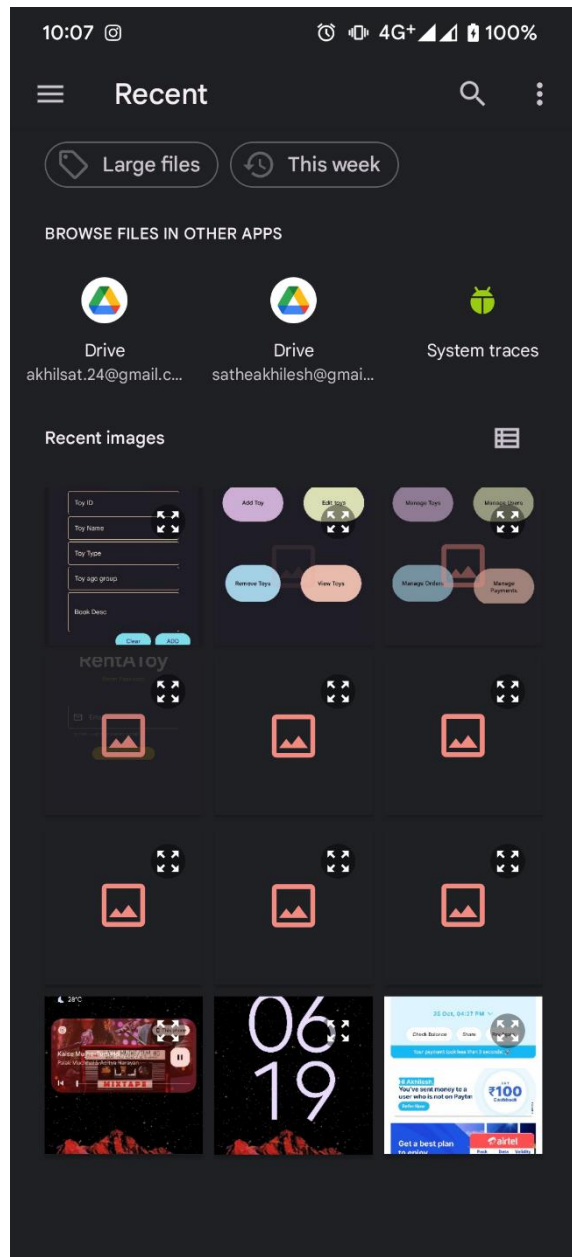
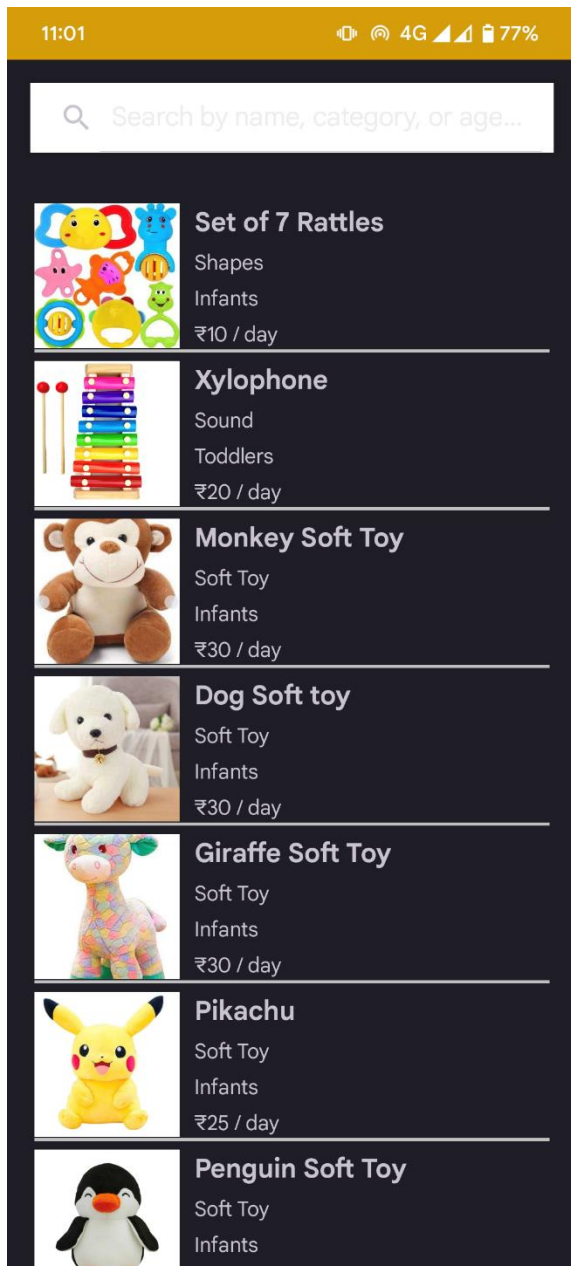
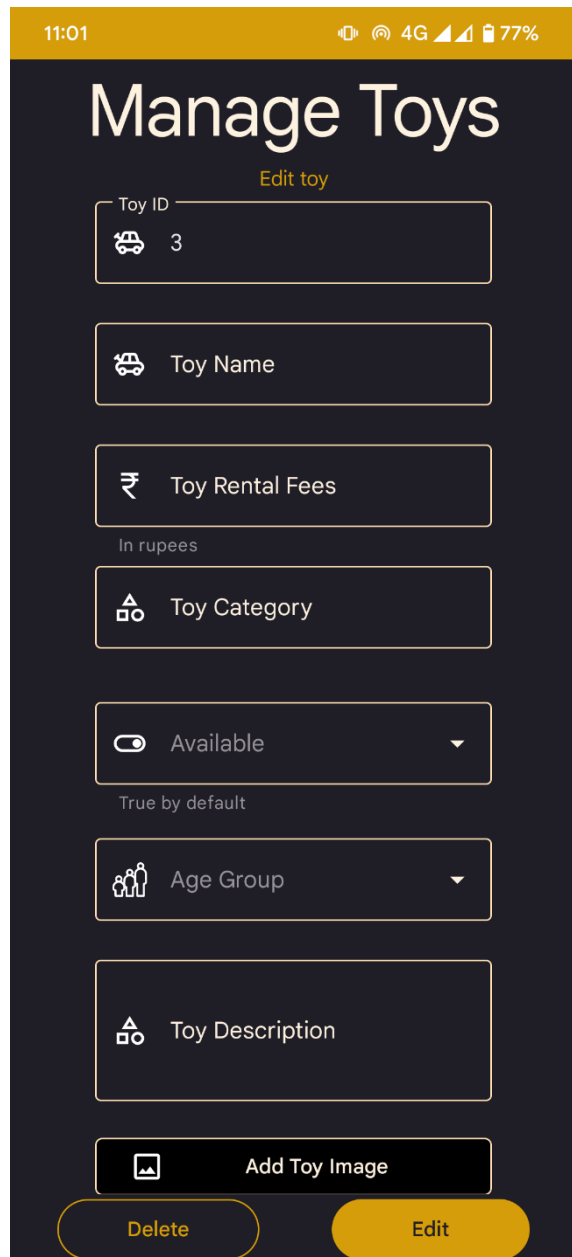


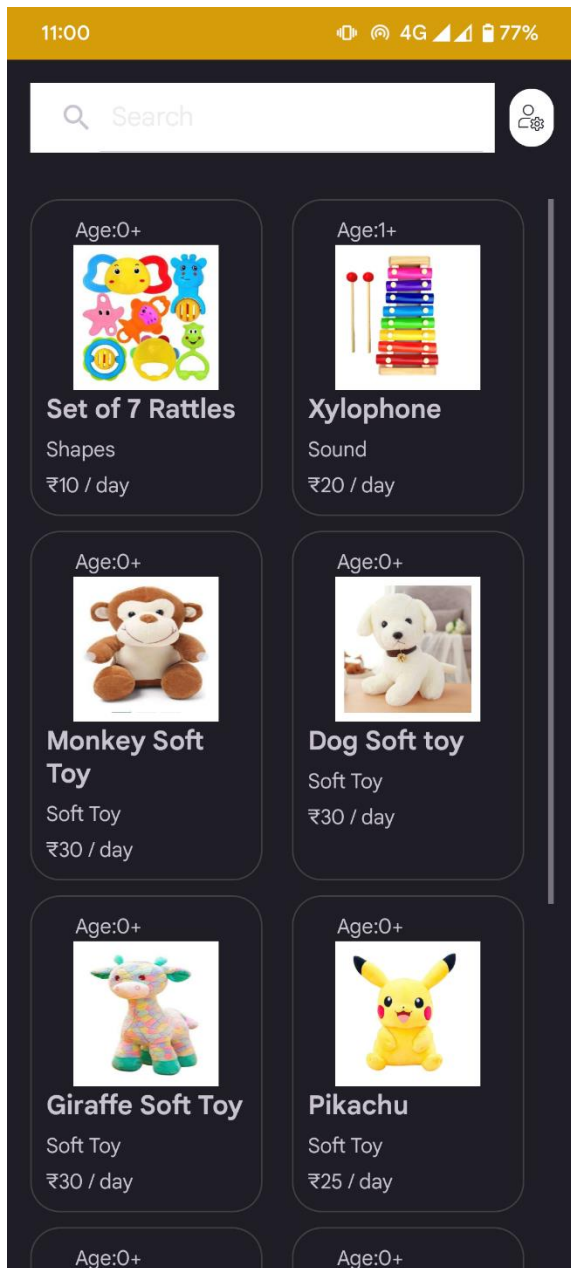
Image Picker



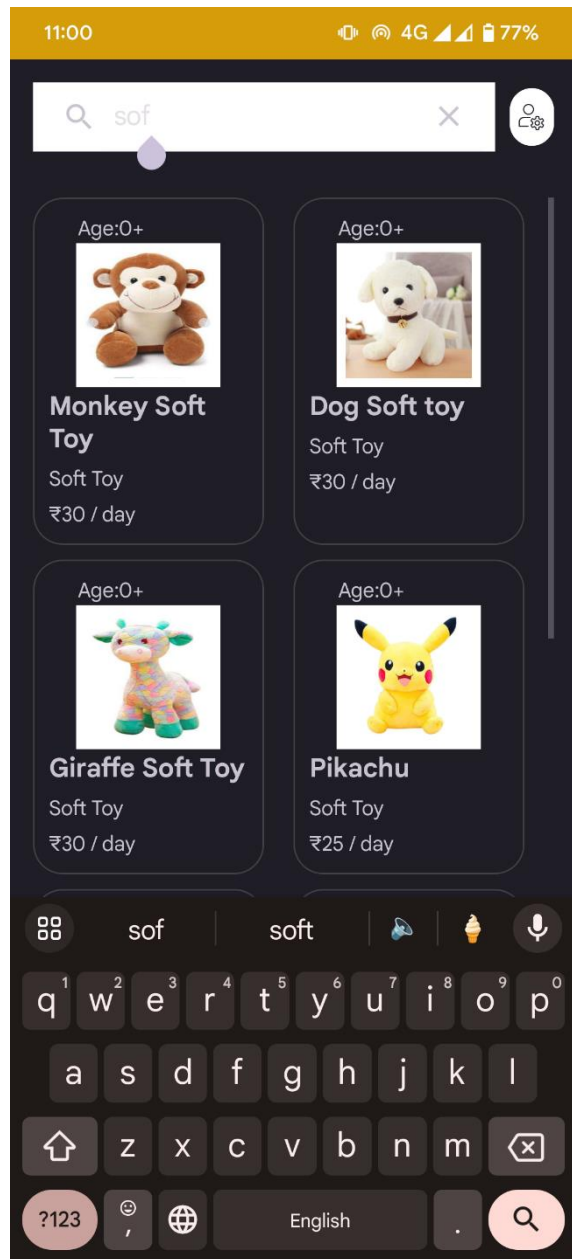
Admin View Toys



Admin:Edit/Delete Toy



User



User:Search

CHAPTER 4: CODING Sample code

Signup page

```
package com.rentatoy
```

```
import android.content.Intent
import android.os.Bundle
import android.util.Log
import android.view.View
import android.widget.AdapterView
import android.widget.AutoCompleteTextView
import android.widget.Button
import android.widget.EditText
import android.widget.TextView
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import com.google.android.material.datepicker.MaterialDatePicker
import com.google.android.material.textfield.TextInputLayout
import com.google.firebase.Firebase
import com.google.firebase.auth.FirebaseAuth
import com.google.firebase.auth.auth
import com.google.firebase.database.FirebaseDatabase
import java.util.Calendar
```

```
class Signup : AppCompatActivity() {
    private lateinit var loginButton: TextView
    private lateinit var email: EditText
    private lateinit var password: EditText
    private lateinit var confPassword: EditText
    private lateinit var fullNameEdt: EditText
    private lateinit var phoneEdt: EditText
    private lateinit var addressEdt: EditText
    private lateinit var dobEdt: EditText
    private lateinit var signupButton: Button

    private lateinit var auth: FirebaseAuth

    private lateinit var userTypeSpinner: AutoCompleteTextView
    private lateinit var userTypeInputLayout: TextInputLayout
    private lateinit var fullNameLayoutEdt: TextInputLayout
    private lateinit var emailLayoutEdt: TextInputLayout
    private lateinit var phoneLayoutEdt: TextInputLayout
    private lateinit var passLayoutEdt: TextInputLayout
    private lateinit var addressLayoutEdt: TextInputLayout
    private lateinit var dobLayoutEdt: TextInputLayout
```

```

private lateinit var confPassLayoutEdt: TextInputLayout

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_signup)

    val sharedPrefs=getSharedPreferences("RentAToy",MODE_PRIVATE)
    val mEditor = sharedPrefs.edit()
    //Firebase
    auth = Firebase.auth

    //EditText
    email = findViewById(R.id.edtEmail)
    password = findViewById(R.id.edtPass)
    confPassword = findViewById(R.id.edtConfPass)
    fullNameEdt = findViewById(R.id.edtFullName)
    phoneEdt = findViewById(R.id.edtPhone)
    addressEdt = findViewById(R.id.edtAddress)
    fullNameEdt = findViewById(R.id.edtFullName)

    //buttons
    loginButton = findViewById(R.id.txtLoginBtn)
    signupButton = findViewById(R.id.btnSignUp)
    dobEdt = findViewById(R.id.edtDOB)
    var userType = ""
    //InputLayout
    fullNameLayoutEdt = findViewById(R.id.edtFullNameLayout)

    passLayoutEdt = findViewById(R.id.edtPassTxtInpLayout)
    confPassLayoutEdt = findViewById(R.id.edtConfPassLayout)
    userTypeInputLayout = findViewById(R.id.inputLayoutUserType)
    emailLayoutEdt = findViewById(R.id.edtEmailLayout)
    phoneLayoutEdt = findViewById(R.id.edtPhoneLayout)
    addressLayoutEdt = findViewById(R.id.edtAddressLayout)
    dobLayoutEdt = findViewById(R.id.edtDOBLayout)

    //Spinner
    userTypeSpinner = findViewById(R.id.spinnerUserType)
    val userTypeList = arrayOf("Admin", "Customer", "Delivery Partner")

    val userTypeArrayAdapter = ArrayAdapter(this,
R.layout.layout_spinner_item, userTypeList)
    userTypeSpinner.setAdapter(userTypeArrayAdapter)

```

```

dobLayoutEdt.visibility = View.GONE
addressLayoutEdt.visibility = View.GONE
userTypeSpinner.setOnItemClickListener { parent, _, position, _ ->

    //Toast.makeText(baseContext, item, Toast.LENGTH_SHORT).show()
    when (parent.getItemAtPosition(position).toString()) {
        "Admin" -> {
            dobLayoutEdt.visibility = View.GONE
            addressLayoutEdt.visibility = View.GONE
            userType = "A"
        }

        "Customer" -> {
            dobLayoutEdt.visibility = View.GONE
            addressLayoutEdt.visibility = View.VISIBLE
            userType = "C"

        }

        "Delivery Partner" -> {
            userType = "D"
            addressLayoutEdt.visibility = View.GONE
            dobLayoutEdt.visibility = View.VISIBLE
        }
    }

}

userTypeInputLayout.setEndIconOnClickListener {
    userTypeSpinner.showDropDown()
}

loginButton.setOnClickListener {

    val intent = Intent(this, Login::class.java)
        .apply {
            flags = Intent.FLAG_ACTIVITY_NEW_TASK or
Intent.FLAG_ACTIVITY_CLEAR_TASK
        }
        startActivity(intent)
    }
dobEdt.setOnFocusChangeListener { view, _ ->
    if (view.isFocused) {
        view.performClick()
    }
}

```

```

    }
}
dobEdt.setOnClickListener {

    val datePicker =
        MaterialDatePicker.Builder.datePicker()
            .setTitleText("Select date")

            .setSelection(MaterialDatePicker.todayInUtcMilliseconds())
            .build()

    datePicker.show(
        supportFragmentManager,
        ""
    )
    datePicker.addOnPositiveButtonClickListener { selectedDateMillis -
>
        // Convert the selected date from milliseconds to a Calendar
instance
        val selectedDate = Calendar.getInstance()
        selectedDate.timeInMillis = selectedDateMillis

        // Now you can access various components of the selected date
        val year = selectedDate.get(Calendar.YEAR)
        val month = selectedDate.get(Calendar.MONTH) // Note: Months
are zero-based
        val day = selectedDate.get(Calendar.DAY_OF_MONTH)
        val date = "$day/$month/$year"
        // Do something with the selected date, for example, print it

        dobEdt.setText(date)
    }

}
signupButton.setOnClickListener {
//        Toast.makeText(baseContext, "Signup button pressed",
Toast.LENGTH_SHORT).show()
        val emailTxt = email.text.toString()
        val passTxt = password.text.toString()
        val confPassTxt = confPassword.text.toString()
        val fullNameTxt = fullNameEdt.text.toString()
        val phoneTxt = phoneEdt.text.toString()
        val addressTxt = addressEdt.text.toString()
        val dobTxt = dobEdt.text.toString()

        if (userType == "") {
            userInputLayout.error = "User Type is required"

```

```

    }
    else if (fullNameTxt.isEmpty()) {
        userTypeInputLayout.error = null
        fullNameLayoutEdt.error = "Required *"
    }
    else if (emailTxt.isEmpty()) {
        fullNameLayoutEdt.error = null
        emailLayoutEdt.error = "Required *"
    }
    else if
(!android.util.Patterns.EMAIL_ADDRESS.matcher(emailTxt).matches()) {
        fullNameLayoutEdt.error = null
        emailLayoutEdt.error = "Enter a valid email"
    }
    else if (phoneTxt.isEmpty()) {
        emailLayoutEdt.error = null
        phoneLayoutEdt.error = "Required *"
    }

    else if (phoneTxt.length < 10 || phoneTxt.length > 10) {
        emailLayoutEdt.error = null
        phoneLayoutEdt.error = "Invalid phone"
    }
    else if (userType == "C" && addressTxt.isEmpty()) {
        phoneLayoutEdt.error = null
        addressLayoutEdt.error = "Required *"
    }

    else if (userType == "D" && dobTxt.isEmpty()) {
        phoneLayoutEdt.error = null
        dobLayoutEdt.error = "Required *"
    }

    else if (passTxt.isEmpty()) {

        addressLayoutEdt.error = null
        phoneLayoutEdt.error = null
        dobLayoutEdt.error = null
        passLayoutEdt.error = "Required *"

    } else if (passTxt.length < 6) {
        phoneLayoutEdt.error = null
        passLayoutEdt.error = "Password is too short"
        Toast.makeText(this, "Password is too short",
Toast.LENGTH_SHORT).show()
    } else if (confPassTxt.isEmpty()) {

```

```

passLayoutEdt.error = null
confPassLayoutEdt.error = "Required *"

} else if (passTxt != confPassTxt) {
    passLayoutEdt.error = null
    confPassLayoutEdt.error = "Passwords do not match."
} else {

    confPassLayoutEdt.error = null
    val tag = this.localClassName
    auth.createUserWithEmailAndPassword(emailTxt, passTxt)
        .addOnCompleteListener(this) { task ->

        if (task.isSuccessful) {
            // Sign in success, update UI with the signed-
in user's information

            Log.d(tag, "createUserWithEmail:success")
            if (userType == "") {
                userTypeInputLayout.error = "Please select
a user type"
            }

            val user = auth.currentUser

            val userData = hashMapOf(
                "uid" to user?.uid.toString(),
                "userType" to userType,
                "name" to fullNameTxt,
                "contact" to phoneTxt,
                "email" to user?.email
            )

            when (userType) {
                "C" -> userData["address"] = addressTxt
                "D" -> userData["dob"] = dobTxt
            }

            FirebaseDatabase
                .getInstance()
                .getReference("Users")
                .child(user?.uid.toString())
                .setValue(userData)
                .addOnSuccessListener {
                    Log.d("Auth", "User Added
Successfully")

```



```

//                                val intent = Intent(this,
AdminHome::class.java)
//                                startActivity(intent)
//
mEditor.putString("userType",userType)
//                                }
//                                "C"->{
//                                val intent = Intent(this,
UserHome::class.java)
//                                startActivity(intent)
//
mEditor.putString("userType",userType)
//                                }
//                                "D"->{
//                                val intent = Intent(this,
UserHome::class.java)
//                                startActivity(intent)
//
mEditor.putString("userType",userType)
//                                }
//                                }
//                                }
//                                }
//                                .addOnFailureListener { e ->
//                                Log.e("Auth", "error", e)
//                                }

} else {
    // If sign in fails, display a message to the
user.

    Log.w(tag, "createUserWithEmail:failure",
task.exception)

    Toast.makeText(
        baseContext,
        "Authentication failed.",
        Toast.LENGTH_SHORT
    ).show()

}

}

}

}

```

CHAPTER 5: Testing

5.1 Test Cases

Test Case ID	Test Case Scenario	Test Case Description	Test Data	Expected Results	Actual Results	Pass/Fail
TC001	Successful Login	Valid email and password for a customer user	Valid email and password	User is logged in, redirected to the `UserHome` activity	User successfully logged in, redirected to `UserHome`	Pass
TC002	Invalid Email Format	Invalid email format entered	Invalid email format	Display error message for invalid email address	Error message displayed: "Invalid Email Address"	Pass
TC003	Empty Email Field	Empty email field submitted	Empty email field	Display error message indicating that the email field is required	Error message displayed: "Required *"	Pass
TC004	Empty Password Field	Empty password field submitted	Empty password field	Display error message indicating that the password field is required	Error message displayed: "Required *"	Pass
TC005	Password Less Than 6 Characters	Password less than 6 characters entered	Short password	Display error message indicating that the password is too short	Error message displayed: "Password is too short"	Pass
TC006	Incorrect Email or Password	Valid email but incorrect password	Valid email and incorrect password	Display authentication failed message	Error message displayed: "Authentication failed."	Pass
TC007	Reset Password	Click on "Reset Password" link	N/A	Navigate to the `ResetPassword` activity	Redirected to the `ResetPassword` activity	Pass
TC008	Navigate to Sign Up	Click on "Sign Up" link	N/A	Navigate to the `Signup` activity	Redirected to the `Signup` activity	Pass
TC009	Successful Login for Admin	Valid email and password for an admin user	Valid admin email and password	User is logged in, redirected to the `AdminHome` activity	User successfully logged in, redirected to `AdminHome`	Pass
TC010	Login with Non-existing Account	Valid email and password for an account that doesn't exist	Valid email and password	Display authentication failed message	Error message displayed: "Authentication failed."	Pass

CHAPTER 6: LIMITATIONS OF SYSTEM

Some limitations of the RentAToy project are as follows:

- While the Toy Rental App offers numerous benefits, it also has some limitations:
- Availability of toys may vary by location.
- Dependence on third-party payment gateways.
- Limited control over toy condition after rental.
- Potential challenges with logistics and delivery services

CHAPTER 7: PROPOSED ENHANCEMENTS

The RentAToy project has potential for further enhancements and expansions, including:

- To further improve the Toy Rental App, potential future enhancements may include:
- Integration with augmented reality to enhance the toy browsing experience.
- Expansion to other platforms (iOS, web).
- Collaborations with toy manufacturers and retailers.
- Loyalty programs and discounts for frequent users.
- User-generated content, such as toy reviews and recommendations.

CHAPTER 8: CONCLUSION

RentAToy is a promising venture aiming to address the current challenges parents face in accessing a variety of toys for their children. The feasibility study indicates strong market demand for a user-friendly toy rental platform. The technical, operational, and financial aspects are well-considered, showcasing the viability of the business model. With multiple revenue streams and a focus on awareness and user acquisition, RentAToy is positioned to fill a significant gap in the market, offering parents a modern and convenient solution for affordable and diverse toy rentals.

CHAPTER 9: BIBLIOGRAPHY

1. Android Developers. "Android Developers."
<https://developer.android.com/docs>.
2. Google. "Firebase Documentation."
<https://firebase.google.com/docs>.
3. Material Design. "Text fields - Material Design."
<https://m3.material.io/components/text-fields/overview>.
4. Stack Overflow. "Stack Overflow."
<https://stackoverflow.com/>.
5. Freepik. "Freepik."
<https://www.freepik.com/>.
6. Google Fonts. "Google Fonts."
<https://fonts.google.com/>.
7. Mozilla Developer Network. "Mozilla Developer Network (MDN)."
<https://developer.mozilla.org/en-US/docs/>.
8. Firebase. "Get Started with Firebase for Android - Documentation." .
<https://firebase.google.com/docs/android/setup>.
9. Android Developers. "Kotlin Programming Language".
<https://developer.android.com/kotlin>.
10. Firebase. "Firebase Authentication in Android - Documentation."
<https://firebase.google.com/docs/auth/android/start>.
11. Firebase. "Cloud Firestore for Android - Documentation."
<https://firebase.google.com/docs/firestore>.
12. Firebase. "Firebase Realtime Database for Android - Documentation."
<https://firebase.google.com/docs/database/android/start>.