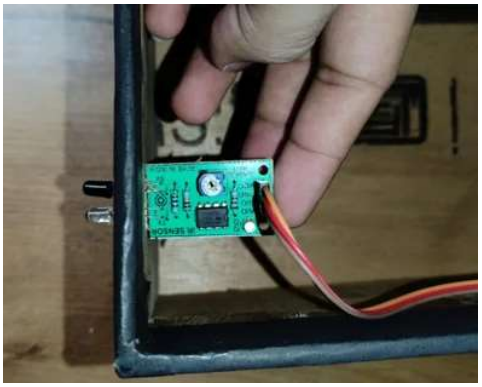
AUTODESK Instructables

Raspberry Pi Based Automated Library Management System

By AnneshuNK in CircuitsRaspberry Pi

Introduction: Raspberry Pi Based Automated Library Management System



In the modern educational landscape, libraries play a pivotal role in the learning process. However, traditional library management methods have proven to be inefficient and are often fraught with challenges, such as mis placed books, long queues, and difficulty in tracking transactions. To address these issues, we embarked on a mission to create a more efficient, automated, and secure library management system.

The Automated Library Management System is designed to revolutionize how libraries function. Its primary go al is to replace manual processes with an innovative, automated solution that enhances the library experience for both students and staff. The project focuses on automation, security, user-friendliness, and scalability as it s core requirements.

The Automated Library Book Management System is developed to meet specific requirements and objectives:

1. The system should automate book issuance and return processes, reducing manual intervention and mi nimizing errors. Automation should be efficient, reducing transaction times.
2. The system should include anti-theft capabilities to prevent unauthorized book removal.
3. Access to the system should be controlled and secure to safeguard library resources and data.
4. The system should maintain a real-time database of book information, including book details and user re cords. Database operations should be fast and accurate.
5. The system should provide intuitive user interfaces for both students and library staff.
6. Users should be able to issue and return books with ease, and staff should manage the system effortles sly.

7. Users should receive real-time notifications about due dates and fines via email. Notifications should be prompt and accurate.

Supplies

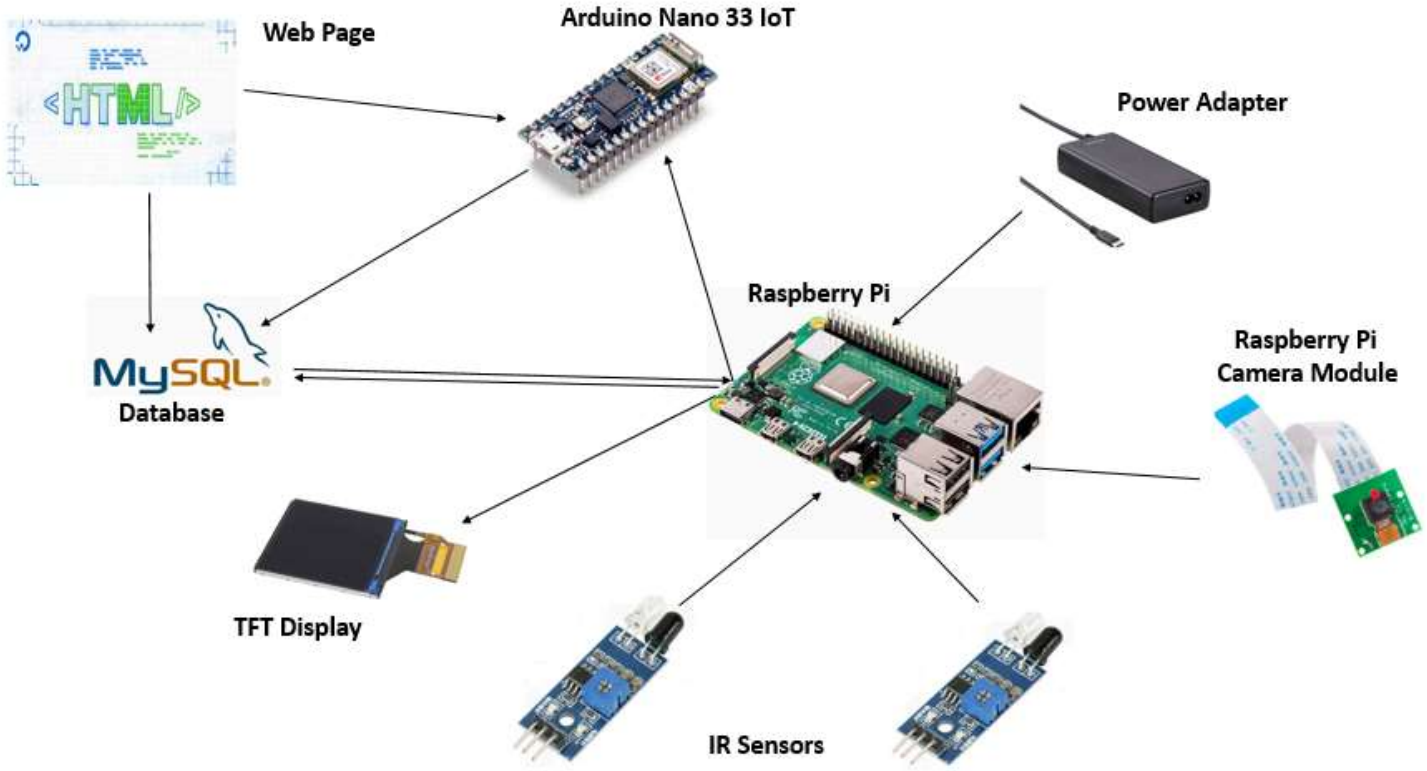
Hardware Requirements:

- Raspberry Pi 4 Model B
- IR Sensors
- Raspberry Pi CAM Module
- TFT Display
- Arduino Nano 33 IoT
- Raspberry Pi Power Adapter
- Arduino UNO (Extra)
- Jumber Wires (Optional)
- Bread Board (Optional)

Software Requirements:

- Arduino IDE
- Raspberry Pi Buster/Bullseye OS
- Python 2.7/3
- MySQL Database
- React (Optional/For Webpage)
- Firebase (Optional)
- Node JS (Optional)
- GitHub Repo

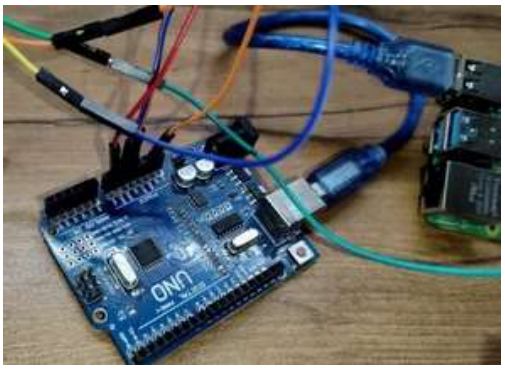
Step 1: Hadware Architechture

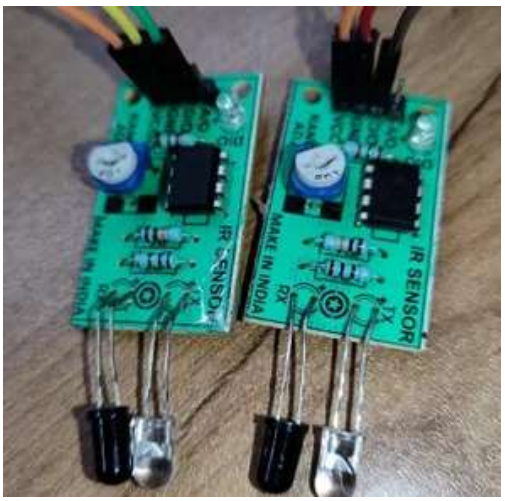


The system architecture is a harmonious integration of diverse hardware and software components, meticulou sly interconnected to craft a fully functional system tailored to the precise needs of both the students as well a s the staff. The system's hardware architecture includes:

1. **Raspberry Pi 4 Model B:** The Raspberry Pi serves as the central processing unit, orchestrating all syst em functions. It connects to the IR sensors, camera module and the TFT Display through it.
2. **IR Sensors:** The IR sensors, connected to the Raspberry Pi's GPIO pins, detect user interactions. Whe n a user waves their hand near the sensors, the system processes the input for book issuance or return.
3. **Raspberry Pi Cam Module**: The Camera module is used for barcode scanning. It's connected to the R aspberry Pi, enabling the system to capture and process barcode information from books and library ID cards.
4. **TFT Display:** The TFT Display provides a user-friendly GUI to show them the steps they need to take in order to issue or return a book.
5. **Arduino Nano 33 IoT:** The Arduino Nano is responsible for adding new users/books to the library itself.
6. **MySQL Database:** The MySQL database is hosted on a server, and the Raspberry Pi communicates wit h it over the internet. It manages book details and user records.

Step 2: Setting Up the Main System





In hardware setup, connecting the hardware in an integrated manner is important to keep trak of what part of t he overall system is performing what action and how it will be sending the data to other components of the sy stem.

1. **Raspberry Pi 4B Model:** The Raspberry Pi serves as the central processing unit and connects to other hardware components through its GPIO (General Purpose Input/Output) pins. GPIO pins can be used fo r various purposes, including input and output.
2. **IR Sensors:**The IR sensors are typically connected to the Raspberry Pi's GPIO pins for digital input. Yo u can use multiple GPIO pins to connect several IR sensors if needed. These sensors are used to detec t user interactions like waving their hand, and the Raspberry Pi reads the input from these sensors throu gh the GPIO pins.
3. **Raspberry Pi Camera Module:**The Camera module can be connected to the Raspberry Pi's dedicated camera interface (CSI - Camera Serial Interface). The camera module usually plugs directly into the ded icated CSI connector on the Raspberry Pi.
4. **TFT Display:** The TFT Display can be connected to the Raspberry Pi via the GPIO pins. There are libra ries available for Raspberry Pi to control TFT displays. You'll typically connect pins for power, ground, an d various data pins to enable the display. The specific GPIO pins used may depend on the display mode l you have.

5. **Arduino Uno (Extra):** The Arduino Uno is used to make up for the lack of power pins the Raspberry Pi after connecting it to the TFT display. You may use any other method to other up you IR sensors.
6. **Arduino Nano 33 IoT:** The Arduino Nano is going to be used like a remote to add new users/books to th e database. It will be separately connected to the device that the user will be using to add data to the da tabase.

*Note: Some of you who are using headless connection with Raspberry Pi directly, you may find it that ater ena bling the **camera module (legacy camera) in the Bullseye OS** the VNC shows **Desktop Not Available**. To r esolve this issue either migrate to Buster OS or use a HDMI display to build the project.*

Step 3: Raspberry Pi Software - Library Installation

```
# Import necessary libraries and modules
import cv2  # OpenCV for capturing video frames
from pyzbar.pyzbar import decode  # Library for barcode decoding
import mysql.connector  # MySQL connector to interact with the database
import time  # For time-based operations
import RPi.GPIO as GPIO  # Raspberry Pi GPIO library
import smtplib  # Library for sending emails
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
import Tkinter as tk  # Tkinter for GUI interface (Note: Tkinter should be replaced with tkinter for Python 3)
from datetime import datetime, date, timedelta  # Import datetime module
```

The very first criteria before jumping into coding is the installation of the below mentioned libraries as we will b e using them to induce the required functionalitie in our system.

1. **cv2 (OpenCV) library:** This library plays an important role in capturing video frames and employing artif icial intelligence to detect barcode presence. To learn how to install OpenCV correctly. Click Here!
2. **pyzbar library:** It's a crucial component for actual barcode scanning and data collection.
3. **mysql.connector library:** This library facilitates seamless communication with the MySQL server, ensu ring smooth data transfer to and from the database.
4. **smtp library:** For various scenarios, the smtp library is used for sending emails to the user about book i ssuance or return along with the fine information if any.
5. **Tkinter library:** This library takes charge of creating an intuitive graphical user interface (GUI) for the T FT display, guiding users through book issuance and return processes.

Note: Some of you may encounter that the OpenCV library is not working with the Camera Module in the Bulls eye OS so you may need to migrate into the Buster OS.

Note: If you are facing problems with downloading the OpenCV library. Click Here!

Step 4: Coding the System

Hopefully by now you are ready with you hardware and have completely installed all the necessary libraries to you Raspberry Pi successfully.

Now you are ready to code. You can build your own logic to give the system specific functionalities of you choi ce or simply copy the code from the attached file that I am providing below.

```
# Connect to the MySQL database
db = mysql.connector.connect(
    host="Your hotname/localhost",
    user="Your user name",
    password="Your MySQL Password",
    database="Your database name"
)
cursor = db.cursor()
```

In the above code snippet you will be placing your own Database creadentials. See the Step 6 to setup the da tabase.

```
# Create an SMTP connection for sending emails
smtp_server = 'smtp.gmail.com'
smtp_port = 587
smtp_username = 'Your gmail account ID'
smtp_password = 'You gmail app password'

smtp_server = smtplib.SMTP(smtp_server, smtp_port)
smtp_server.starttls()  # Use TLS encryption

# Log in to your email account
smtp_server.login(smtp_username, smtp_password)
print('SMTP Server connected.')
```

In the above code snippet you will be placing your gmail ID and app password. Learn How?

Step 5: Setting Up Arduino Remote

Once you are complete with the main code for the system, it is time to setup your Arduino Nano Remote that will be used to store new users/books directly to the database.

You simply need to connect your Nano directly to the device you are using to send the data to the databse (su ppose your computer).

Then you need to write and upload the code to add users/books to the database. You can build you own code or use the code file attached.

Note: The attached code only adds new users.

Step 6: Setting Up the MySQL Database

By now you are almost done with your project, the only part left is the database setup. The first and foremost step to setup the database is the installing the **mariadb/mqsql library** in the Raspberry Pi. We will be using t hese library to access the functions to create a database directly at the Raspberry Pi's terminal.

Once you are done with the installation of the **mariadb/mqsql library**, you need to open a terminal in the Ras pberry Pi and use the commands to create the database.

The document attached below provides the all the commands that are need to create the database and also i nterface with it. Or Click Here!

Finally, after you are done with the database you simplly need to put the databse credential into your system's main code.

You can also access the database directly now by using **myphpadmin** and your Raspberry Pi's server.

With that you are done!

Step 7: Developing a Website (Extra/Optional)

Note: This is not required for building the working system. It is just used to make the process of adding new b ooks/ users to the database user friendly.

To develop the website you can use any web developing platform but the preferred would be React and then using Google Firebase you can send the user data from the web to the Arduino Nano and from the Arduino N ano to the databse.

I have not made the website as it was not necessary but you may do so if you want to make the project truly y ours.

Step 8: Run Your System

After following all the above steps accurately, you will have a fully functioning Automated Library Management System. I have also attached a video explaining and demonstrating how I have made the system. You may ch eck it out to better understand the process of making this project.

This project is made by Anneshu Nag (NK-Works) & Team Embed-Tech.

This article is submitted to **Deakin School of IT & Chitkara University of Enginering and Technology** as p art of my individual assessment for **SIT210 - Embedded Systems Development Project.**

Note: If you have any query and write them down in the comments section. I will try to answer them to the bes t of my capabilities.