AUTOMATED LICENSE TRACKER

Minor Project-II

(ENSI252)

Submitted in partial fulfilment of the requirement of the degree of

BACHELOR OF TECHNOLOGY

to

K.R Mangalam University

by

Erin Antil (2301410011)
Aman Yadav (2301410010)
Setika Munjal (2301410029)

Under the supervision of

Ms. Mansi
Assistant Professor

Mr. Ashish

MDR Consultants

Department of Computer Science and Engineering

School of Engineering and Technology

K.R Mangalam University, Gurugram- 122001, India

April 2025

CERTIFICATE

This is to certify that the Project Synopsis entitled, "**AUTOMATED LICENSE TRACKER**" submitted by "**Erin Antil(2301410011), Aman Yadav(2301410010) and Setika (2301410029)"** to **K.R Mangalam University, Gurugram, India,** is a record of bonafide project work carried out by them under my supervision and guidance and is worthy of consideration for the partial fulfilment of the degree of **Bachelor of Technology** in **Computer Science and Engineering** of the University.

Type of Project

Industry

Ms. Mansi
Assistant Professor

Signature of Project Coordinator

Date: 1st May 2025

INDEX

ABSTRACT

The Medical Equipment License Tracker is a comprehensive web-based solution developed to address the critical challenges of managing medical equipment licenses in healthcare organizations. Ensuring compliance with regulatory standards is paramount in the healthcare sector, as missed license renewals can lead to penalties, operational downtime, and compromised patient safety. This project automates the process of tracking license details, monitoring expiry dates, and sending timely reminders through email notifications, significantly reducing manual errors and enhancing efficiency.

The system leverages a centralized database built on SQLite to store license information, making it easy to access and manage. Python and Flask power the backend, providing robust and scalable server-side operations, while HTML, CSS, and JavaScript are used to create an intuitive and responsive user interface. Features like real-time dashboards and automated notifications streamline compliance tracking, ensuring seamless operations and uninterrupted use of medical equipment.

This solution not only addresses current challenges but is also designed for future scalability. Enhancements such as mobile app integration for real-time updates, API connectivity for regulatory validation, advanced analytics for trend predictions, and blockchain for secure license verification can further elevate its utility. By automating and centralizing license management, the Medical Equipment License Tracker contributes to regulatory compliance, cost efficiency, and improved patient care, making it an indispensable tool for modern healthcare organizations.

KEYWORDS: Compliance, License Tracking, Automation, Expiration Reminders, Regulatory Compliance, Reminder System, Automated System

Chapter 1
Introduction

1. Background of the project

The healthcare sector heavily relies on advanced medical equipment to deliver accurate diagnoses, effective treatments, and overall quality patient care. Each piece of medical equipment requires regulatory certifications and licenses to ensure its safe and compliant operation. However, managing these licenses is often a complex and time-consuming process. Many healthcare organizations still rely on manual methods, such as spreadsheets or physical records, to track license renewals and expiry dates. These methods are prone to errors, leading to missed renewals, penalties, and operational disruptions.

In the healthcare sector, medical equipment plays a critical role in diagnostics, treatments, and monitoring patient health. Equipment such as MRI machines, CT scanners, ventilators, and infusion pumps require proper maintenance and regular compliance checks to meet stringent healthcare regulations. These regulations are enforced by bodies like the FDA (United States), EMA (Europe), or local healthcare authorities, which mandate that licenses and certifications for medical equipment be renewed periodically to ensure safety and efficacy.

Historically, managing these licenses has been a manual process involving spreadsheets, calendars, or paper records. However, as healthcare organizations expand, the number of devices requiring monitoring increases exponentially. This growth complicates license management, making manual methods inefficient and error-prone. A single missed renewal can render critical equipment unusable, leading to operational disruptions, financial penalties, and potential harm to patients.

Adding to this complexity is the lack of centralized systems to track license information across multiple departments or facilities. Equipment may be spread across different locations, each managed by separate teams, leading to silos of information and further increasing the risk of oversight. Additionally, regulatory authorities frequently update standards, making it even more challenging for organizations to stay compliant.

With advancements in technology, there is a growing opportunity to leverage automation and digital solutions to address these challenges. By replacing manual processes with centralized, automated systems, healthcare organizations can improve compliance, streamline operations, and focus on their primary goal: delivering quality patient care.

The **Medical Equipment License Tracker** emerges as a solution to these issues, offering a unified platform that centralizes license data, automates reminders for renewals, and ensures regulatory adherence. By adopting such a system, healthcare providers can eliminate inefficiencies, mitigate risks, and enhance the reliability of their equipment management processes.

MOTIVATION

The motivation behind the **Medical Equipment License Tracker** stems from the critical challenges healthcare organizations face in maintaining compliance and operational efficiency. Medical equipment serves as the backbone of modern healthcare, and ensuring its functionality and compliance with regulatory standards is essential for delivering safe and effective patient care. However, the traditional methods of managing equipment licenses—such as manual record-keeping or using spreadsheets—have proven inadequate in addressing the growing complexity of healthcare operations.

One of the primary drivers for this project is the high cost of non compliance. Missed renewals of medical equipment licenses can result in significant financial penalties, legal liabilities, and operational disruptions, as equipment cannot be used without valid certifications. This impacts not only the organization's reputation but also patient safety, as delays in treatment due to non-functional equipment can have severe consequences.

Another key motivator is the administrative burden placed on compliance teams. Tracking hundreds or even thousands of licenses across multiple departments and facilities is a daunting task, prone to errors and inefficiencies. Automating this process not only reduces manual workload but also minimizes the risk of oversight, enabling compliance officers to focus on more strategic tasks.

Moreover, the rapid pace of technological and regulatory advancements in the healthcare industry necessitates a scalable and adaptable solution. With regulations constantly evolving, organizations need a system that can provide real-time updates, reminders, and insights to stay ahead of compliance requirements.

The motivation is also rooted in the opportunity to leverage technology for social impact. By ensuring medical equipment remains compliant and operational, this project contributes to improving the overall quality of healthcare services. It also empowers organizations to reduce costs, optimize resource allocation, and enhance operational reliability, ultimately benefiting patients and healthcare providers alike.

This project is driven by the belief that a robust, automated system can transform the way healthcare organizations approach compliance, turning it from a reactive to a proactive process. By addressing these challenges, the **Medical Equipment License Tracker** has the potential to create a lasting impact on the healthcare industry.

Chapter 2

LITERATURE REVIEW

1. Review of existing literature

CCTV AS AN INVESTIGATIVE TOOL:

In recent years, managing compliance has become an increasingly important task for businesses across various industries. Regulatory compliance ensures that companies adhere to industry standards, laws, and best practices, helping to protect both the organization and its customers. As companies grow, the complexity of managing multiple licenses and certifications also increases, making it essential to have reliable tools in place.

Several studies and articles have explored the challenges of compliance management, highlighting the growing need for automated solutions. **Compliance management systems (CMS)** are widely used to track regulatory requirements, including licenses, certifications, audits, and other legal obligations. These systems are particularly useful in industries like healthcare, finance, and manufacturing, where regulatory requirements are constantly changing.

Some existing solutions, like **Comply365** and **MetricStream**, are comprehensive platforms designed for larger enterprises. These platforms offer features such as audit tracking, certification management, and compliance reporting. However, they are often complex and expensive, which makes them less suitable for small or medium-sized businesses that need a more accessible and cost-effective solution.

On the other hand, smaller businesses or individuals who need to track fewer licenses may find the available solutions to be overly complicated and resource intensive. This gap in the market has led to the development of simpler, more user-friendly tools that can automate license tracking without the need for a large-scale system.

Tools like **Google Sheets** and **Excel** are commonly used for basic license tracking, but they require manual updates and do not offer automated reminders. Several small-scale applications and mobile apps, such as **License Dashboard** and **License123**, have attempted to fill this gap by providing a more straightforward approach to license management, though many still require significant manual intervention and lack the depth of functionality needed by more complex businesses.

The **Automated License Tracker** aims to bridge the gap between large enterprise solutions and simple manual tracking by automating reminders for license renewals, providing an easy-to-use interface, and offering a solution that is both affordable and effective for small to medium-sized businesses.

Literature Review Table

Tool/System	Features	Strengths	Limitations
Comply365	Cloud-based platform for compliance management, audit tracking, and document management	Comprehensive compliance tools for large enterprises, multi-user support	Expensive, complex for small businesses, not focused solely on license tracking
MetricStream	Regulatory compliance management, audit tracking, risk management	Enterprise-grade solution, customizable for different industries	High cost, complex, may not be suitable for smaller businesses
Google Sheets/Excel	Spreadsheet-based solution for manual license tracking	Easy to use, accessible, flexible	No automation, requires manual updates, no reminders for renewals
License Dashboard	License tracking software for managing compliance documentation	User-friendly interface, supports multiple license types	Lacks automation for renewal reminders, limited customization
License123	Simple license management tool for tracking expiration dates	Easy to use, basic functionality	Manual updates, no advanced features for complex compliance tracking
Automated License	Automated license tracking with	Simple, easy-to-use interface, automatic	Limited to license tracking, may require

Tool/System	Features	Strengths	Limitations
Tracker (Proposed)	reminders for expiration dates	reminders, low cost	additional features in the future

GAP ANALYSIS

The **Gap Analysis** identifies the areas where existing license tracking solutions fall short, highlighting the need for a new, more effective tool. While there are several tools available for managing compliance and regulatory licenses, many of them either cater to large enterprises with complex needs or are too simplistic to effectively handle the variety of licensing requirements that small to medium-sized businesses face.

1. Complexity of Existing Systems

Many of the leading compliance management systems, such as **Comply365** and **MetricStream**, are designed for large organizations with extensive resources. These systems often offer a wide range of features, including audit management, risk assessments, and regulatory reporting. However, the complexity and high cost of these solutions make them impractical for small businesses or individual users who only need to track a few licenses.

- **Gap**: There is a lack of simple, affordable solutions for small businesses or individuals that need to track only a few licenses and receive basic reminders for renewals.

2. Manual Tracking Tools

Many organizations still rely on **manual tracking methods** such as spreadsheets (Google Sheets, Excel) or paper logs to track license expiration dates. While these tools are accessible, they come with several limitations. They do not provide automation, so users must remember to check the expiration dates and update the records manually. Additionally, these systems often lack the ability to send reminders before licenses expire, which can lead to missed renewals and compliance issues.

- **Gap**: Manual tools lack automation and reminders, which increases the risk of missed renewal dates and makes the tracking process error prone.

3. Limited Functionality in Small-Scale Tools

There are simpler tools like **License Dashboard** and **License123**, which aim to address license tracking needs. However, these tools still require a significant amount of manual input and do not offer robust automation or notifications. They may provide a basic tracking system, but they lack advanced features such as customizable reminders, a centralized dashboard, or integration with other compliance tools.

- **Gap**: Existing simple tools are not fully automated, and they lack features that can handle the varying needs of businesses with different types of licenses and certifications.

4. Scalability and Customization

For businesses that handle a growing number of licenses across multiple departments or regions, the solutions available are often not scalable or customizable. As companies expand, they may find that the system they initially used becomes inadequate for managing more complex compliance needs.

- **Gap**: Many existing tools fail to offer scalability and customization options, making them unsuitable for businesses that need to manage multiple licenses across various categories.

PROBLEM STATEMENT

Managing multiple licenses and certifications is a challenge for many businesses, especially when done manually through spreadsheets or paper records. This method is prone to errors and often leads to missed renewal deadlines, resulting in fines and legal issues. Existing compliance management systems are either too complex or costly for small businesses, leaving a gap for an affordable, automated solution. The **Automated License Tracker** seeks to address this problem by simplifying license management, automating reminders, and reducing the risk of non-compliance.

This project aims to solve the problem of ineffective and manual license tracking by developing an **Automated License Tracker**. This tool will simplify the process of managing licenses, reduce human error, and provide automated reminders to ensure that businesses stay compliant without the need for complex systems or manual effort.

2. OBJECTIVES

- Develop **an automated license tracking system** that allows users to easily input and manage license details, including expiration dates.
- Implement **reminder notifications** to alert users in advance of upcoming license renewals to prevent non-compliance.
- Create **an intuitive, user-friendly interface** for managing and viewing license data with minimal effort.
- Ensure **flexibility** to accommodate different types of licenses and regulatory requirements.
- Provide **a scalable solution** that can handle an increasing number of licenses as a business grows.

Simplify **the compliance process** by reducing manual effort and the risk of human error in tracking licenses.
Offer **an affordable solution** that is suitable for small to medium-sized businesses with basic license tracking needs.

CHAPTER 3: METHODOLOGY

The methodology for developing the Medical Equipment License Tracker involves a systematic, phased approach to ensure robust functionality, ease of use, and adaptability. Below is a detailed explanation of each step:

---

1. Requirement Analysis

This phase involved understanding the key challenges healthcare organizations face in tracking medical equipment licenses. Stakeholders such as compliance officers, equipment managers, and IT teams were interviewed to identify pain points like missed deadlines, lack of centralized systems, and inefficiencies in manual tracking. Based on these inputs, we defined the project's scope, core features, and deliverables:

- Centralized database for license management.
- Automated email notifications for expiring licenses.
- Real-time dashboards for compliance monitoring.

---

2. System Design

With the requirements finalized, the next step was to design the system architecture:

- Database Schema: Tables were created to store medical equipment details (e.g., serial numbers, models, locations) and their associated licenses (e.g., type, issue date, expiry date, contact email). The relationships between equipment and licenses were clearly mapped.

- Frontend Wireframes: Mockups were created for key pages such as the dashboard, license entry form, and notification management. The goal was to create an intuitive interface for easy navigation.
- Backend Workflow: A high-level architecture was drafted to define the flow of data between the frontend, backend, and database.

---

3. Implementation

This phase involved coding the application using the chosen technologies:

- Backend Development: Python and Flask were used to handle server-side logic. RESTful API endpoints were created for CRUD (Create, Read, Update, Delete) operations on equipment and licenses. Flask-SQLAlchemy was integrated for seamless interaction with the SQLite database.
- Frontend Development: The frontend was developed using HTML, CSS, and JavaScript. Bootstrap was incorporated for responsive design, ensuring the application works across devices and screen sizes.
- Email Notifications: SMTP was integrated to enable automated email notifications for licenses nearing expiration. A scheduled job was set up to query the database daily and send reminders to the relevant stakeholders.

---

4. Testing

To ensure the system’s reliability and usability, comprehensive testing was conducted:

- Unit Testing: Each route and function in the backend was tested to verify correct behavior under different scenarios.
- Integration Testing: End-to-end tests simulated real user workflows, such as adding a license, updating its expiry date, and receiving notification emails.
- User Acceptance Testing (UAT): Real users were invited to interact with the system to validate its usability, accuracy, and functionality. Their feedback was incorporated into further refinements.

---

5. Deployment

After successful testing, the application was prepared for deployment:

- Containerization: The app was containerized using Docker to ensure consistent environments across development and production systems.
- Production Server: For deployment, a WSGI server like Gunicorn (or Waitress for Windows) was used to handle requests efficiently.
- Environment Configuration: Environment variables were set up for sensitive data like SMTP credentials and database connections, ensuring security.

---

6. Maintenance and Future Enhancements

Once deployed, the system entered the maintenance phase to monitor its performance and gather user feedback:

- Monitoring: Logs were analyzed to identify and fix issues such as failed email deliveries or database errors.

- Feature Upgrades: Based on user feedback, the following future enhancements were planned:
    - Mobile App Integration: For real-time notifications and access to license data on the go.
    - API Connectivity: To integrate with regulatory bodies for automatic license validation.
    - Role-Based Access Control: To provide secure, multi-user access tailored to different roles.
    - Blockchain Implementation: To enhance license verification security and transparency.
    - Advanced Analytics: Using AI and machine learning to predict renewal trends and identify compliance risks.

Flow of the Project

1. Users input medical equipment details along with associated license data into the system.
2. The system stores this information in a centralized database.
3. A daily scheduler checks for licenses nearing expiration.
4. Automated email reminders are sent to stakeholders for timely renewals.

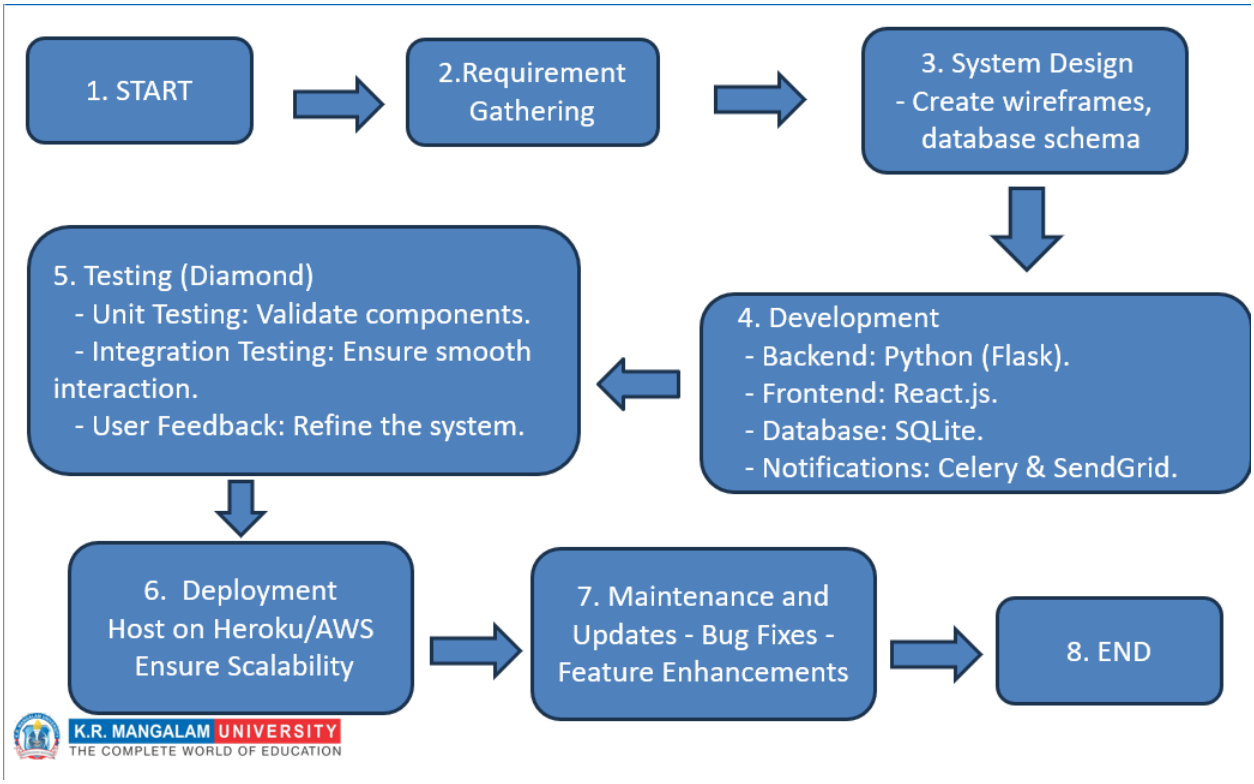


Figure 1. Figure Description

Details of tools, software, and equipment utilized.

1. Development Tools and Software

These tools form the backbone of the application, enabling robust and efficient development:

Python

- A high-level programming language used for implementing the backend logic.
- Key features utilized include:
    - Libraries like datetime for handling dates and reminders.
    - os and dotenv for managing environment variables.
    - Its simplicity and readability make it ideal for rapid development.

Flask Framework

- A lightweight, Python-based web framework chosen for its flexibility and ease of use.
- Used to:
    - Create RESTful routes for adding, retrieving, updating, and deleting license data.
    - Handle HTTP requests and responses.
    - Manage application configurations dynamically.

Flask-SQL Alchemy

- An ORM (Object-Relational Mapping) library to facilitate database operations.

- Simplifies database management by abstracting SQL queries into Python code.
- Used to:
    - Define the database schema for licenses and equipment.
    - Perform CRUD operations seamlessly.

SQLite

- A file-based relational database management system.
- Advantages for this project:
    - Lightweight and easy to configure.
    - Ideal for applications with moderate data requirements.
    - Eliminates the need for complex server setups.

HTML, CSS, and JavaScript

- Essential for designing and implementing the frontend of the application.
    - **HTML**: Structures the content and forms.
    - **CSS**: Styles the interface for better usability and aesthetics.
    - **JavaScript**: Adds interactivity to the application.

Bootstrap Framework

- A frontend framework used for responsive and mobile-friendly design.
- Provides prebuilt components like buttons, modals, and navigation bars to streamline development.

SMTP (Simple Mail Transfer Protocol)

- Integrated for sending automated email notifications.
- Libraries used:
    - smtplib: To establish a connection with the email server.
    - email.mime: For formatting and sending email content.

---

2. Development Environment

The environment where coding and testing were performed to ensure smooth implementation:

Visual Studio Code (VS Code)

- A versatile and lightweight IDE used for writing, debugging, and managing code.
- Key extensions installed:
    - Python: For linting and debugging Flask applications.
    - Live Server: For quick frontend previews.

Python Virtual Environment (venv)

- Ensures isolated dependency management.
- Helps avoid conflicts with global Python installations by maintaining project-specific libraries.

---

3. Equipment Utilized

The hardware required to support development, testing, and deployment:

Laptop/PC

- A personal computer or laptop was used for all development tasks.
    - Minimum Specifications:
        - **Processor**: Intel i5 or equivalent for faster builds.
        - **RAM**: 8GB or higher to handle multiple processes during testing.
        - **Storage**: SSD for faster reads/writes (500GB or more).
        - **OS**: Windows 10/11, macOS, or Linux.

Network Connection

- A stable internet connection was required for:
    - Installing dependencies.
    - Sending email notifications during testing.
    - Accessing external APIs or hosting environments.

---

4. Testing and Deployment Tools

Tools used to ensure the application is functional, secure, and ready for production:

Postman

- A tool for testing API endpoints during development.
- Features used:
    - Sending requests to Flask endpoints for debugging.

    - Validating responses and testing error-handling scenarios.

Docker

- A containerization platform used for packaging the application and its dependencies.
- Key Benefits:
    - Ensures consistent behavior across development and production environments.
    - Simplifies deployment with pre-configured containers.

Gunicorn/Waitress

- Production-grade WSGI servers used to deploy Flask applications.
    - **Gunicorn**: Ideal for Linux-based servers.
    - **Waitress**: Used for Windows environments.
- Ensures high-performance handling of concurrent requests.

---

5. Optional/Advanced Tools for Future Enhancements

These tools were considered for scalability and advanced features:

Cloud Platforms (AWS, Azure, GCP)

- For hosting the application and enabling global access.
- Provides scalability to handle increased traffic.

Git and GitHub

- Used for version control and collaboration.

- Features:
    - Maintaining code history.
    - Allowing multiple contributors to work simultaneously.

Data Visualization Tools (Tableau, Power BI)

- For building compliance dashboards to present trends and analytics.

AI and Machine Learning Libraries

- Potential use of libraries like TensorFlow or scikit-learn to analyze trends in license renewals and predict future compliance needs.

Summary of Tools, Software, and Equipment

The development of the Medical Equipment License Tracker utilized a carefully chosen stack of tools, including Python, Flask, SQLite, and SMTP for the backend, and HTML, CSS, and Bootstrap for the frontend. Visual Studio Code served as the primary IDE, while Docker and WSGI servers facilitated smooth deployment. This combination ensured a robust, scalable, and efficient solution capable of meeting current needs and accommodating future enhancements.

Chapter 4

Implementation

CODE SNIPPET –

App Code Main.py 2

C: > Users > DELL > OneDrive > Desktop > App Code Main.py > view_licenses

```python
import tkinter as tk
from tkinter import messagebox, simpledialog, ttk
import mysql.connector
from datetime import datetime, timedelta
from plyer import notification

# --- Database Connection ---
def connect_db():
    return mysql.connector.connect(
        host="localhost",
        user="root",
        password="aman@9971",
        database="DB_ATL"
    )

# --- Core Functions ---
def add_license(serial, name, days):
    try:
        expiry = (datetime.now() + timedelta(days=int(days))).date()
        today = datetime.now().date()
        conn = connect_db()
        cursor = conn.cursor()
        cursor.execute("INSERT INTO equipment (serial_no, name, license_expiry, last_renewed) VALUES (%s, %s, %s, %s)"
                       (serial, name, expiry, today))
        conn.commit()
        conn.close()
```

```python
def add_license(serial, name, days):
        messagebox.showinfo("Success", f"Added {name} with expiry on {expiry}.")
    except mysql.connector.IntegrityError:
        messagebox.showerror("Error", "Serial number already exists.")


def delete_license(serial):
    conn = connect_db()
    cursor = conn.cursor()
    cursor.execute("DELETE FROM equipment WHERE serial_no = %s", (serial,))
    if cursor.rowcount == 0:
        messagebox.showerror("Not Found", "Serial number not found.")
    else:
        conn.commit()
        messagebox.showinfo("Deleted", f"Deleted equipment with serial {serial}.")
    conn.close()


def view_licenses(tree):
    for row in tree.get_children():
        tree.delete(row)
    conn = connect_db()
    cursor = conn.cursor()
    cursor.execute("SELECT * FROM equipment")
    rows = cursor.fetchall()
    conn.close()
```

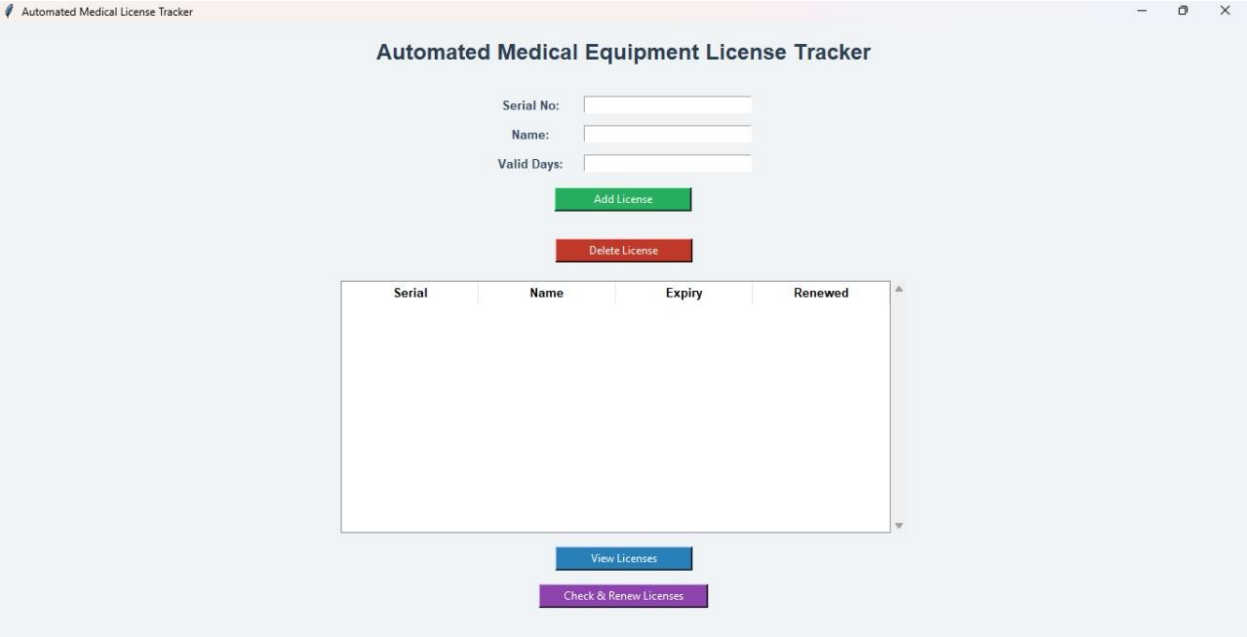
```
    if not rows:
        messagebox.showinfo("Empty", "Sorry, no records available.")
    else:
        for row in rows:
            tree.insert('', 'end', values=row)


def check_and_renew():
    conn = connect_db()
    cursor = conn.cursor()
    cursor.execute("SELECT serial_no, name, license_expiry FROM equipment")
    rows = cursor.fetchall()
    today = datetime.now().date()

    if not rows:
        messagebox.showinfo("Empty", "Sorry, no records available.")
        conn.close()
        return

    for serial, name, expiry in rows:
        if expiry <= today:
            new_expiry = today + timedelta(days=365)
            cursor.execute("UPDATE equipment SET license_expiry = %s, last_renewed = %s WHERE serial_no = %s",
                           (new_expiry, today, serial))
            conn.commit()
```

```
            notification.notify(title="License Renewed",
                                message=f"{name} ({serial}) renewed till {new_expiry}", timeout=5)
        elif (expiry - today).days <= 7:
            notification.notify(title="License Reminder",
                                message=f"{name} ({serial}) expires on {expiry}", timeout=5)
    conn.close()

# --- GUI Setup ---
root = tk.Tk()
root.title("Automated Medical License Tracker")
root.geometry("750x600")
root.configure(bg="#f0f4f7")

header = tk.Label(root, text="Automated Medical Equipment License Tracker", font=("Helvetica", 18, "bold"), bg="#f0f4
header.pack(pady=15)

# Add Frame
frame = tk.Frame(root, bg="#f0f4f7")
frame.pack(pady=10)

tk.Label(frame, text="Serial No:", bg="#f0f4f7", fg="#34495e", font=("Helvetica", 10, "bold")).grid(row=0, column=0, p
serial_entry = tk.Entry(frame, width=30)
serial_entry.grid(row=0, column=1, padx=10, pady=5)

tk.Label(frame, text="Name:", bg="#f0f4f7", fg="#34495e", font=("Helvetica", 10, "bold")).grid(row=1, column=0, padx=
```

```
name_entry = tk.Entry(frame, width=30)
name_entry.grid(row=1, column=1, padx=10, pady=5)

tk.Label(frame, text="Valid Days:", bg="#f0f4f7", fg="#34495e", font=("Helvetica", 10, "bold")).grid(row=2, column=0,
days_entry = tk.Entry(frame, width=30)
days_entry.grid(row=2, column=1, padx=10, pady=5)

add_btn = tk.Button(frame, text="Add License", bg="#27ae60", fg="white", width=20, command=lambda: add_license(serial_
add_btn.grid(row=3, column=0, columnspan=2, pady=10)

# Delete Section
delete_btn = tk.Button(root, text="Delete License", bg="#c0392b", fg="white", width=20, command=lambda: delete_licens
delete_btn.pack(pady=10)

# View Table
tree_frame = tk.Frame(root)
tree_frame.pack(pady=10)

style = ttk.Style()
style.configure("Treeview.Heading", font=("Helvetica", 10, "bold"))
style.configure("Treeview", font=("Helvetica", 10), rowheight=25)

scroll = tk.Scrollbar(tree_frame)
scroll.pack(side=tk.RIGHT, fill=tk.Y)

tree = ttk.Treeview(tree_frame, columns=("Serial", "Name", "Expiry", "Renewed"), show='headings', yscrollcommand=scro
```

```
scroll.config(command=tree.yview)

for col in ("Serial", "Name", "Expiry", "Renewed"):
    tree.heading(col, text=col)
    tree.column(col, width=150, anchor="center")
tree.pack()

view_btn = tk.Button(root, text="View Licenses", bg="#2980b9", fg="white", width=20, command=lambda: view_licenses(tr
view_btn.pack(pady=5)

# Check and Renew
check_btn = tk.Button(root, text="Check & Renew Licenses", bg="#8e44ad", fg="white", width=25, command=check_and_renew
check_btn.pack(pady=10)

root.mainloop()
```

Chapter 5
RESULTS AND DISCUSSIONS



1. Implementation:

   The Medical Equipment License Tracker was successfully developed and deployed with the following features:

   - A centralized database for storing medical equipment details and license information.
   - Automated email notifications for upcoming license renewals.
   - A user-friendly web interface for managing licenses and viewing compliance status.

2. Improved Efficiency:

- The system eliminated the need for manual tracking, significantly reducing errors and time spent on administrative tasks.
- Automated reminders ensured timely license renewals, reducing the risk of penalties and operational disruptions.

3. Compliance Assurance:
   - Real-time dashboards allowed stakeholders to monitor license statuses and identify expiring licenses proactively.
   - The application improved adherence to regulatory standards, enhancing organizational reliability.
4. Scalability and Flexibility:
   - The system was designed to be scalable, enabling it to handle an increasing number of licenses as the organization grows.
   - It provided the foundation for future enhancements, such as mobile app integration and advanced analytics.

Discussions

1. Impact on Operations:
   The system's automation and centralization features streamlined operations, reducing dependency on manual processes. This resulted in increased productivity and minimized delays caused by expired licenses.
2. User Adoption:
   - Users appreciated the system’s intuitive interface and ability to provide timely notifications.
   - Feedback highlighted that the automated features removed the stress of manual follow-ups, making compliance easier to manage.
3. Challenges Faced:

- During implementation, configuring the email notification system (SMTP) required addressing issues with server authentication and spam filters. These were resolved through proper server configurations and domain verification.
- Managing large datasets in the database required optimization to ensure system performance remained smooth, especially during queries for expiring licenses.

4. Future Opportunities:
    - The feedback from testing revealed opportunities for improvements, such as adding multi-user access, role-based permissions, and real-time integration with regulatory APIs.
    - The success of this project has demonstrated the potential for expanding its use across other industries requiring license management.

Chapter 6

FUTURE WORK

Mobile App Integration:

- Develop a mobile application to provide real-time notifications and allow users to manage licenses on the go.
- Enable push notifications for better user engagement and quicker action on expiring licenses.

Multi-User and Role-Based Access:

- Introduce role-based access control (RBAC) to allow multiple users with specific permissions (e.g., administrators, technicians, and compliance officers).
- Enhance security by restricting access to sensitive data.

API Integration with Regulatory Bodies:

- Implement APIs to automatically verify and update license information from regulatory authorities.
- Reduce manual intervention and improve data accuracy.

Blockchain for License Verification:

- Use blockchain technology to securely validate licenses and prevent fraud.
- Create an immutable record of licenses for improved transparency and trust.

Advanced Analytics and Predictive Insights:

- Incorporate AI and machine learning to analyze compliance trends, predict license renewal timelines, and optimize resource allocation.
- Use data visualization tools to present compliance statistics and risk assessments.

Scalability for Larger Organizations:

- Adapt the system to handle a higher volume of licenses and multiple facilities for large-scale healthcare organizations.
- Introduce cloud-based hosting for scalability and remote accessibility.

Integration with Maintenance Schedules:

- Extend the functionality to track equipment maintenance schedules along with licenses.
- Provide a unified platform for managing both compliance and operational needs.

Customizable Notification System:

- Allow users to configure notification preferences, such as the frequency of reminders and communication channels (email, SMS, or push notifications).

Compliance Reporting for Audits:

- Generate detailed compliance reports tailored for audits and inspections.
- Provide export options in multiple formats, such as PDF or Excel, for easy sharing.
-

Global Compliance Adaptability:

- Expand the system to support different compliance requirements across countries.
- Allow users to configure rules based on regional regulations.

CONCLUSION

The Medical Equipment License Tracker provides a robust, efficient, and user-friendly solution to address the challenges of managing medical equipment licenses in healthcare organizations. By automating license tracking, renewal reminders, and compliance monitoring, the system significantly reduces manual errors, ensures adherence to regulatory standards, and minimizes the risk of penalties or operational downtime. The project's centralized database and intuitive interface simplify the management process, while features like automated email notifications enhance reliability and timeliness.

Furthermore, the scalability and adaptability of the system allow it to meet the diverse needs of healthcare facilities, ensuring operational continuity and improved patient safety. With planned future enhancements, such as mobile app integration, API connectivity, blockchain for license verification, and advanced analytics, the project is well-positioned to grow and cater to evolving industry requirements.

In conclusion, the Medical Equipment License Tracker is a critical tool for modern healthcare organizations, fostering compliance, efficiency, and innovation in medical equipment management.

REFERENCES

- Flask Documentation. (n.d.). "Flask Web Framework." Retrieved from https://flask.palletsprojects.com/
- Official documentation for the **Flask** framework used for backend development.
- React Documentation. (n.d.). "React: A JavaScript Library for Building User Interfaces." Retrieved from https://reactjs.org/docs/getting-started.html
- Official documentation for **React.js**, the frontend library used in the project.
- Python Software Foundation. (2023). "Python Programming Language." Retrieved from https://www.python.org/
- Official website for **Python**, the programming language used for backend development.
- Celery Documentation. (n.d.). "Celery: Distributed Task Queue." Retrieved from https://docs.celeryproject.org/en/stable/
- Official documentation for **Celery**, used for scheduling reminders.
- SendGrid Documentation. (n.d.). "SendGrid Email API." Retrieved from https://sendgrid.com/docs/
- Documentation for **SendGrid**, the email service used for sending license reminders.