Secure Mailbox with Mobile Connectivity

Neehar Sawant, Roshun Navin, Avadh Patel

ECE 445
Team 26
Problem: Mail theft from unsecured mailboxes

Solution: Automatically locking mailbox with mobile control and notifications
High Level Requirements

1. The mailbox must **automatically lock within 10 seconds** of the door being closed if mail is placed inside and there is no schedule set to leave it unlocked.

2. The mailbox must **lock and unlock within 5 seconds of pressing the corresponding button** on the application.

3. The mailbox must **send a notification within 30 seconds of an action being made on the mailbox**. This includes opening and closing the mailbox as well as whether mail is present.
Block Diagram
Original Mailbox Design

Magnetic Contact Sensors

Servo Motor Lock

Ultrasonic Sensors
Completed Design
ESP32 with Built in WiFi

- ESP32-WROOM processes mailbox logic and sensor data
- Allows for mobile application communication
- Surface mounted on PCB
Power

- 10,000mAh battery bank delivers 5V to PCB via micro USB
- 5V delivered to ultrasonic sensors and servo motor
- 3.3V delivered to ESP32-WROOM
Mobile Application UI

- Gives user access to manual lock and unlock
- Set a schedule for time where mailbox is unlocked
- Real time status of door, lock, schedule, and mail
Arduino IoT Cloud

- Arduino IoT Cloud allows anyone connected to the internet to interact with hardware
- Functions will run if cloud variables are changed (such as locking the door)

```c
/*
   Since LockDoor is READ_WRITE variable, onLockDoorChange() is executed every time a new value is received from IoT Cloud.
*/

void onLockDoorChange() {
   // Add your code here to act upon LockDoor change
   if(lock_door == 1){
      lockDoor();
   } else{
      unlockDoor();
   }
}
```
Mobile Notifications

- IFTTT allows notifications to be sent to anyone with the app
- Web request from software -> IFTTT web server -> user’s phone

```java
void sendMessage(String message){
    http.begin("https://maker.ifttt.com/trigger/door/with/key/keyvalue?value1=\"+message\);
    http.GET();
    http.end();
}
```
Project Video
### Quantitative Analysis

<table>
<thead>
<tr>
<th>Action</th>
<th>Time (s)*</th>
</tr>
</thead>
<tbody>
<tr>
<td>automatically locking mailbox when mail is present</td>
<td>0.35</td>
</tr>
<tr>
<td>locking/unlocking mailbox from mobile app</td>
<td>0.48</td>
</tr>
<tr>
<td>updating mobile app dashboard</td>
<td>0.86</td>
</tr>
<tr>
<td>receiving push notification for mailbox updates</td>
<td>2.21</td>
</tr>
<tr>
<td>locking/unlocking mailbox from schedule</td>
<td>0.74</td>
</tr>
</tbody>
</table>

*average of 15 attempts

<table>
<thead>
<tr>
<th>Ultrasonic</th>
<th>Distance (cm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower bound</td>
<td>2.35</td>
</tr>
<tr>
<td>Upper bound</td>
<td>18.93</td>
</tr>
</tbody>
</table>
Challenges

- MQTT protocol and notifications being blocked by WiFi network
- Connectivity issues between AWS IoT and ESP32
- Soldering WROOM and SMD components
- 3D printing design
- Placing ultrasonic sensors
Conclusions

What We Learned

- Cloud Applications
- Arduino C controlling sensors
- Wireless connectivity protocols
- PCB Design
- PWM signals

What We Would Do Differently

- Adding button for manual lock
- 3D print the top of the mailbox separately
- Housing for sensors/motor in mailbox
Further Improvements

- Metal structure instead of 3D printed plastic
- Camera for live feeds and increased range of detecting mail
- More robust locking mechanism
- Potentially use solar power and a larger mAh rechargeable battery
- Ensure waterproofing
Thank you!

Q&A