# Product: CleanCAPS
## Team: Group 6


### Abstract

CleanCAPS is an innovative robotic keyboard cleaner designed to address hygiene concerns in high-traffic, high-touch, or shared keyboard environments in an interactive manner. This demo presents a substantially revised system, reflecting key advances in hardware, software, and integration. The original robotic arm and air-pump cleaning mechanism have been replaced with a rail-mounted cleaning head equipped with a vacuum and two rotating brushes, significantly improving debris removal. The cleaning head simulates ultraviolet disinfection using downward-facing LED strips. The TurtleBot operates autonomously, navigating between keyboards and aligning the cleaning head. The object detection model has been fine-tuned using a custom dataset to support accurate keyboard identification. A RESTful server facilitates communication between the vision system and robot control logic. In addition, a web application and administrative dashboard have been developed. These allow users to summon the robot, monitor cleaning progress, and access real-time data including system status, timestamps, and robot location.


## 1. Project management update

According to our proposal, we set the following tasks to be our goals for each of the demos:

### 1.1. Goals

**Demo 1**

- Cleaning - LED and Fan/Pump Function (achieved).
- Cleaning - Mounting design for cleaning head (achieved).
- Vision - Setup Restful Service. (achieved)
- Vision - Keyboard vs Non-Keyboard Classifier (achieved).
- Movement - Basic Payload Along 3 axes (partly achieved).

**Demo 2**

- Cleaning - Cleaning head expansion exploration with vacuum and brushes (achieved).
- Cleaning - Show significant keyboard cleaning (partly achieved).
- Vision - Classifier improvement with CNN partly achieved).
- Vision - Extract cleaning axes to guide the arm movement (not achieved).
- Movement - Environment mapping (achieved).
- Movement - Move head to keyboard position (achieved).

**Demo 3**

- Cleaning - Demonstrate consistently reduced particles (partly achieved).
- Vision - Improve Accuracy (achieved).
- Vision - Test Validation (achieved).
- Movement - Consolidate collaboration with Vision team (achieved).
- Movement - Travel over multiple keyboards in sequence (achieved).
- Web application - Telemetry app for data retrieval and robot control (achieved).

### 1.2. Deviations

Following feedback from Demo 1, we realised our original vision of CleanCAPS being fully autonomous was unrealistic in an office environment, nor was it achievable within the remaining time frame. We realigned CleanCAPS towards a more interactive approach, with the user (e.g., employee or student) requesting CleanCAPS when they want their keyboard cleaned. At the same time, an admin (e.g., janitorial staff or floor manager) monitors and schedules cleans. As a result, we adapted or removed various goals from our final product, while goals to create an admin dashboard and webapp were included.

Feedback from the user study in Demo 2 enabled us to further refine our goals by affirming that the webapp was a good addition but needed improvement, as shown by Figure 8 in the Appendix. Following Demo 1 and 2, there were some concerns about the spread of germs, dust, and debris on the desk by the air-blowing. This led to a major shift in our vision for the project, and we moved to exploring more effective cleaning mechanisms, such as a vacuum and brushes. The word cloud (Figure 9) shows suggestions given by our survey takers during Demo 2, showing what they would like to see in an autonomous keyboard cleaner.

**Partly-achieved justifications**

- **Movement - Basic Payload Along 3 axes (partly achieved).**

  Instead of the rail system outlined in the proposal, the team shifted towards ROS2-based autonomous movement with a Turtlebot so that CleanCAPS would be able to service keyboards across multiple desks and a Robotic arm to manoeuvre the cleaning head to support the height of keyboards. However, the robotic arm proved to be unreliable, and the 50-gram weight constraint was challenging to work around. Thus, we instead redesigned our cleaning head to extend from the top level of the Turtlebot, allowing for a larger cleaning head but restricting CleanCAPS to keyboards at a fixed height.

- **Cleaning - Show significant keyboard cleaning and demonstrate consistently reduced particles (partly achieved)**. Deviation from the original cleaning head plan left us with limited time to design, print, order, and integrate the necessary parts into our system. While we successfully demonstrated partial keyboard cleaning, the results were less than satisfactory, particularly due to the vacuum not being powerful enough for our use case.

- **Vision - Classifier improvement with CNN (partly achieved)** Our goal was to enhance the classifier using a Convolutional Neural Network (CNN) to improve accuracy and robustness. However, due to time constraints and limited human resources -partly because the team had to focus on developing the admin dashboard for the project- we were unable to fully implement and optimize the CNN model. While initial experimentation was conducted, further tuning and evaluation would be required to achieve the desired performance.

- **Vision - Extract cleaning axes to guide the arm movement. (not achieved)** We encountered numerous issues with the arm's functionality, which led us to reassess its feasibility. As a result, we decided to abandon this approach and shift focus to developing a new cleaning head, which we felt would be more achievable for our use case. The downside of this approach is that the flexibility with which we can clean keyboards was significantly reduced.

### 1.3. Organisation

At the beginning of the project, the team was divided into three main groups: Vision, Cleaning Mechanism, and Movement. Each team was responsible for their own subsystem. However, as development progressed and the interdependence between components increased, this structure became inefficient. After Demo 1, the team shifted to a task-based approach, assigning individual responsibilities such as webapp development, TurtleBot movement, and hardware wiring. This allowed team members to work in parallel, reducing delays and improving accountability.

To support this new structure, the team introduced daily task lists, check-ins, and used Trello to track progress and reassign tasks when needed. Having some members in the same physical space also allowed for faster face-to-face problem-solving. For code collaboration, GitHub was used with a feature-branch strategy to avoid conflicts and ensure tested, stable integrations. This reorganization led to clearer communication, faster development, and ultimately enabled the successful delivery of a fully integrated system for the final demo.

### 1.4. Team Members

**Team Contributions**

- **Arin (250h)** – Managed the GitHub repo, developed both versions of the keyboard classifier, handled data collection and testing, built the admin dashboard, and led QA and data visualisation efforts.

- **Guillermo (250h)** – Handled wiring and soldering for both cleaning heads, optimized Pi–Arduino communication, implemented the classifier, integrated the system with the TurtleBot, and tested rail precision.

- **Jieyi (100h)** – Handled motor control and contributed to TurtleBot and rail movement.

- **Kenji (270h)** – Designed and built key hardware, including the vacuum, cleaning heads, rail system, and TurtleBot mount, set up ROS navigation, supported web app integration, and helped with presentations.

- **Matthew (200h)** - Set up object detection with a Pi camera, integrated it with Arduino, and built a web app with backend and QR code support for real-time cleanliness tracking.

- **Mira (250h)** – Managed the project, led all demos and presentations, coordinated the team, handled documentation, and contributed to cleaning head design, hardware assembly, and wiring and soldering.

- **Punom (120h)** – Set up the ROS workspace, developed custom nodes for autonomous TurtleBot navigation, and designed key hardware components including the vacuum pump.

- **Sophia (120h)** – Designed visuals, tested the classifier, collected and annotated data, built an evaluation function, and created the admin UI and 3D cleaning head mount.

### 1.5. Budget

Table 1 displays how the monetary budget of £100 has been spent so far. Estimates were taken for the 3D printing costs and the brushes we ordered. We did not spend our entire monetary budget.

| Item | Cost/Unit (£) | Quantity | Cost (£) |
|---|---|---|---|
| PLA Filament | 0.025/g | 1419g | 35.48 |
| PVA Filament | 0.115/g | 18g | 2.07 |
| Brushes | 10.86 | 2 | 21.72 |
| **Total** | | | **60.67** |

*Table 1.* The spending of the £100 monetary budget.

Table 2 shows how the 10 hours of allocated technician time has been used. We spent 505 minutes of technician time, totaling 9 hours.

| | 3D prints (15min) | Hole Drilling (5min) | Cutting (5min) | Repairs(35 min) | Time Usage (min) |
|---|---|---|---|---|---|
| Demo 1 | 2 | 1 | 1 | - | 40 |
| Demo 3 | 29 | 0 | 6 | 1 | 500 |
| **Total** | **31** | **1** | **7** | **1** | **540** |

*Table 2.* Technician time used

### 1.6. Post-mortem

#### 1.6.1. What went well:

1. **Revised Team Structure**

   After Demo 1, switching from team-based work to individual task ownership significantly improved efficiency. By assigning clear responsibilities for the webapp, dashboard, movement system, wiring, and vision, we enabled parallel progress and eliminated dependence on each other.

2. **Effective Daily Planning and Check-ins**

   Our use of daily check-ins and Trello task boards helped keep everyone accountable and focused. Especially under time constraints, this lightweight project management approach ensured consistent progress and quick resolution of blockers.

3. **Hardware Simplification**

   Moving away from the robotic arm to a rail-mounted cleaning head simplified both control and integration. This change made the cleaning system more stable and reliable, which in turn improved overall system consistency.

4. **Integrated and Functional Web Interfaces**

   The development of the WebApp and Admin Dashboard gave our project a complete, user-facing system. These tools elevated the robot from a prototype to a product concept with real deployment potential.

5. **Autonomous Movement and Detection**

   Achieving reliable keyboard detection and autonomous TurtleBot alignment marked a key milestone. The movement system consistently reached target positions with a low error margin of 2.82 cm, meeting our precision goals.

#### 1.6.2. What Went Wrong

1. **Inconsistent Member Engagement** A significant challenge faced by the team was the inconsistency in member engagement, with some individuals unable to complete their assigned responsibilities within the expected timeframe. This affected the team's overall progress and required a redistribution of tasks to ensure project objectives were achieved.

2. **Project Direction Changes After Demo Feedback** Another major setback was the shift in our project's direction following the feedback received after Demo 1 and Demo 2. This feedback prompted us to re-evaluate key aspects of our design, leading to substantial modifications. We decided to increase user interaction within the system, and, following Demo 2, we completely redesigned the cleaning head mechanism. This included developing a rail-based system with a vacuum component, replacing the original air pump design. Additionally, we removed the robotic arm, as it presented significant challenges in terms of weight limitations and feasibility within our given timeframe. These changes, while ultimately beneficial, required considerable time and effort to implement.

## 2. Quantitative analysis and testing

Quantitative analysis was split into 5 sections: keyboard classifier, navigation, web app, and dashboard, cleaning head, and end-to-end quality assurance.

### 2.1. Keyboard Classifier

**Justification:** The keyboard classifier is responsible for detecting the presence and the position of keyboards on a desk and it serves as a core component to our project. Testing its performance under different conditions ensures that the system works reliably.

**Evaluation:** The following results show the performance of our vision model:

| Metric | Value |
|---|---|
| Total Number of Images | 150 (Keyboard: 75, Other: 75) |
| True Positives (TP) | 53 |
| False Positives (FP) | 0 |
| True Negatives (TN) | 73 |
| False Negatives (FN) | 24 |
| Precision | 1.000 |
| Recall | 0.688 |
| F1 Score | 0.815 |
| Accuracy | 0.840 |

*Table 3.* Performance metrics for keyboard classification model

Our vision model is Resnet-50 (He et al., 2015)trained on the COCO-keyboards (Lin et al., 2015) dataset. It is reliable across most data classes; however, it consistently underperforms on one specific case: images where keyboards are only partially visible, positioned at unusual angles, and captured from a tilted field of view. This systematic limitation contributes to the model's lower-than-expected recall.

The precision of our classifier was impressive and showed that every time the model predicted that an object was a keyboard, it was correct. An F1 score of 0.815 is good, however, the recall lowered it. Accuracy was also good, where the model was correct 84% of the time on all its predictions.

## 2.2. Navigation Quality Assurance

The quality of CleanCAPS's navigation was measured by the time taken to move to a requested keyboard and the accuracy of the Turtlebot's movement.

### 2.2.1. Navigation Time

The navigation time was calculated by requesting two keyboards on opposite sides of a small mock office space and measuring the time to navigate to both and then the time to return home (see figure 7 in the appendix).

**Justification:** To not disrupt the work of a user, CleanCAPS must be able to navigate to a keyboard anywhere in the office in under 90 seconds. Furthermore, our user study reinforced the importance of low navigation time (see figure 9 in the appendix for word cloud with "speed" and "high movement", and a mean satisfaction of 3.41/5 on how fast the robot takes to get to a keyboard in figure 1 in the appendix). Accordingly, a lower navigation time would increase user satisfaction in CleanCAPS.

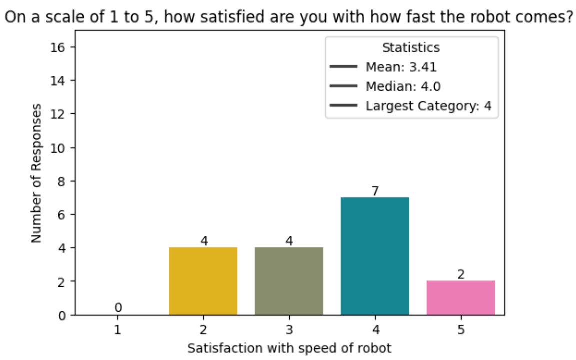


*Figure 1.* Robot Speed Satisfaction

**Evaluation:**

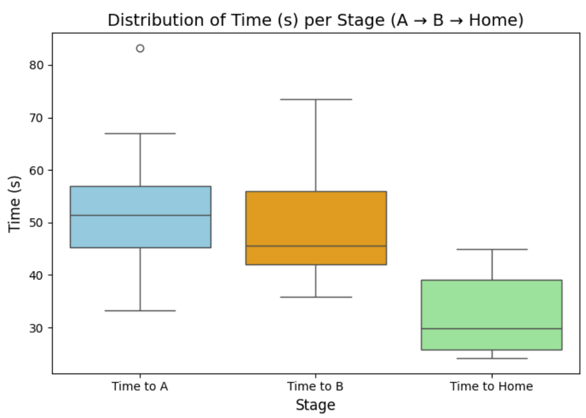


*Figure 2.* Time taken for robot to reach its requested destinations

Over the 10 trials, there was a mean time taken to A of 52.72s and 49.9s for B. This fulfills the above goal. However, we notice that the navigation is inconsistent, with standard deviations of 14.51 and 11.62.

### 2.2.2. Accuracy

The accuracy of CleanCAPS's navigation was measured by setting a goal at measured distances and calculating the difference with CleanCAPS's position. The error radius should be less than 0.005m.

**Justification**: CleanCAPS's movement must be accurate when navigating to requested keyboards and during its cleaning routine. Inaccurate navigation could result in no keyboard being detected by the camera and thus, a failed request.

**Evaluation:**

| Distance (m) | Average Error Radius (m) |
|---|---|
| 1.000 | 0.022 |
| 2.000 | 0.035 |

*Table 4.* Average Radius Error of Turtlebot movement by Distance

In table 4 we can see an average error radius significantly smaller than the target goal. Thus the goal was achieved.

## 2.3. Web App and Dashboard Quality Assurance

### 2.3.1. User Focused Design

**Justification:**

During Demo 2, we asked survey respondents what suggestions they had for the web app. Their responses are summarised in the word cloud in figure 8. Based on this feedback, we implemented several key improvements. One concern was that the user interface displayed that a keyboard had been cleaned before the robot had actually completed the task — or even while it was still in progress. To address this, we adjusted the system's status reporting to reflect real-time updates more accurately. Respondents also commented that the dashboard only showed a generic "waiting" message for the robot. In response, we introduced more detailed status indicators, such as "waiting," "moving," "cleaning," or "cleaned," to give users a clearer understanding of the robot's activity.

**Evaluation:**

We used this feedback to improve the design of both interfaces, and the results from Google Lighthouse show that they are easy to use and built to a good standard. See table 5.

## 2.4. Cleaning Head Quality Assurance

A series of tests were conducted focusing on the accuracy, durability, and performance of the cleaning head. These

| Category | Accessibility | Best Practices | SEO |
|---|---|---|---|
| WebApp | 80 | 96 | 82 |
| Admin Dashboard | 86 | 100 | 100 |

*Table 5.* Lighthouse scores for WebApp and Admin Dashboard.

tests are necessary to assess the feasibility on the clean-head in real offices or libraries where it will be running for multiple hours a day on a variety of keyboards.

### 2.4.1. Cleaning Head Extension Accuracy

From the pitch diameter of the pinion in the rack and pinion extension mechanism, we predict that the cleaning head should extend 98.17 mm. Thus, the head should extend and retract with at most 5 mm difference from 98.17 mm so that the cleaning mechanism still covers the keyboard.

**Justification**: The extension and retraction of the cleaning head must be accurate to ensure correct placement on the cleaning head over the keyboard.

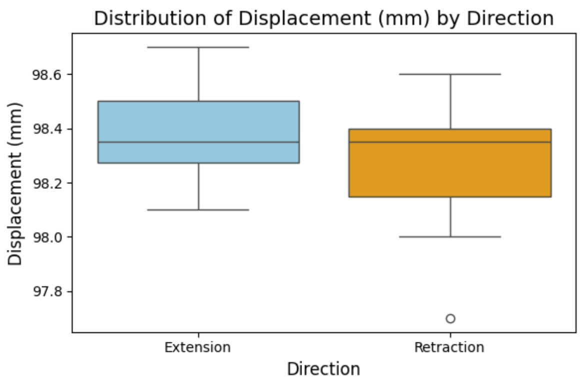


*Figure 3.* Distribution of Displacement by Direction

**Evaluation**: The mean displacement of the cleaning head of 98.38mm for extraction and 98.28mm for retraction. All trials were within 5mm from the predicted, thus successfully satisfying the goal.

### 2.4.2. Durability

**Justification**: The cleaning head must withstand deterioration from hours of daily use to minimize repair costs. To simulate this long-term usage within a short test, we ran the extraction and retraction mechanism of the cleaning head 50 times.

**Evaluation**: After the 50 trials, there was no obvious wear and the cleaning head still extended within 5 mm of 98.17mm. This test displays the reliability and longevity of the mechanism.

### 2.4.3. Cleaning head performance

**Justification:** To evaluate the real-world performance of the cleaning head, we conducted a test to see how many of the 10 pieces of plastic we put on the keyboard were dislodged and removed. Two sets of simulated particles were used: larger debris to simulate debris-like food chunks and smaller debris to simulate debris-like hair, dust, and crumbs. This is important to understand the reliability and the limits of the system.

**Evaluation:** The results showed that the cleaning head is more effective at removing smaller debris. On average, it scored 3.9 out of 10 for smaller debris and 3.1 out of 10 for larger debris. The median scores were 4.0 and 3.0, respectively, with standard deviations of 1.1 and 1.29, indicating slightly more consistency with smaller particles. This can be seen in figure 4.

```
Cleaning Head Effectiveness
Debris Type: Mean - Median - Std - 95% CI
Smaller debris: (3.9, 4.0, 1.1, 0.682)
Bigger debris: (3.1, 3.0, 1.29, 0.8)
```

*Figure 4.* Results for cleaning head performance

While the scores were satisfactory, there is room for improvement. We acknowledge that our current prototype's vacuum employs a weak DC fan, but this is what was available. However, this test showed that our robot can handle smaller debris better than larger debris that tends to be pushed to the side.

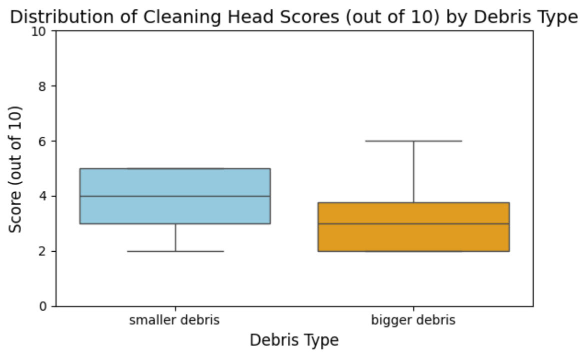


*Figure 5.* Cleaning Head Scores by debris type

### 2.4.4. End to End Quality Assurance

**Justification:** This test is meant to verify how the system runs from start to finish. This test confirms the successful integration of the entire system, from the web app back-end to the final step of cleaning. This is essential to see if all our components are integrated correctly. We checked 1) One request for one keyboard. 2) Two requests for the same keyboard. 3) One request for each of the keyboards.

**Evaluation:** All three tests were completed.

- In Test 1, the robot received the request, navigated, and cleaned the keyboard as expected.
- In Test 2, the system correctly registered two separate requests and cleaned the same keyboard twice.

- In Test 3, the robot handled multiple independent requests, identifying and cleaning each keyboard in sequence.

These results confirmed that CleanCAPS operates as a fully integrated system, capable of handling multiple requests, processing them in order, and performing consistent cleaning.

#### 2.4.5. Battery lifetime

**Justification:** To understand how long CleanCAPS can operate on a single charge, we ran a battery test on the TurtleBot. This was important to evaluate how much cleaning the robot could realistically complete in one session without needing to recharge, especially in larger rooms with multiple keyboards.

**Evaluation:** We let the TurtleBot run through typical tasks: moving around, detecting keyboards, and activating the cleaning mechanism. After 3 tests, the battery lifetime averaged at 1 hour and 35 minutes before it ran out.

## 3. Estimated System Budget

Tables 6, 8 and 7 show that in its current form, one CleanCAPS unit costs £1529.8.

| Product | Cost/Unit (£) | Quantity | Total (£) |
|---|---|---|---|
| Brushes | 10.86 | 2 | 21.72 |
| LED lights | 4.16 | 2 | 8.32 |
| DC fan | 3.16 | 1 | 3.16 |
| Tubing | 51.86/10m | 10cm | 0.519 |
| Arduino Mega 2560 | 40.55 | 1 | 40.55 |
| Grove Relay | 2.99 | 1 | 2.99 |
| 16-Channel PWM Driver | 15.94 | 1 | 15.94 |
| Micro servo Motor | 3.92 | 2 | 7.84 |

*Table 6.* Cleaning Head Budget.

| Product | Cost/Unit (£) | Quantity | Total (£) |
|---|---|---|---|
| TurtleBot | 1275.8 | 1 | 1275.8 |
| Misc (Wires, Glue, Screws, Tape...) | 10 | 1 | 10 |
| Raspberry Pi 3 | 30 | 1 | 30 |
| Grove Shield | 5.58 | 1 | 5.58 |
| Powerbank | 15 | 1 | 15 |
| AA Rechargeable Batteries | 1.2 | 19 | 22.8 |

*Table 7.* General.

| Product | Cost/Unit (£) | Quantity | Total (£) |
|---|---|---|---|
| RSPro Silver Aluminium Profile Strut 20x20x1000 | 6.28 | 35cm | 2.198 |
| RSPro Silver Aluminium Profile Strut 40x40x2000 | 16.50 | 2 x 24 cm | 7.92 |
| Angle joints for 40mm RSPro square struts | 5.70 | 1 | 5.70 |
| Angle joints for 20mm RSPro square struts | 0.66 | 2 | 1.32 |
| Astrosyn Y129 Stepper Motor | 27.57 | 1 | 27.57 |
| Sparkfun Bipolar Microstepping Driver | 24.85 | 1 | 24.85 |

*Table 8.* Camera stand and Rail.

### 3.1. Platform Pricing

During Demo 2, the system was hosted using Cloudflare Workers, which provided robust performance but proved to be unfeasible for continued development. As a result, we migrated to Firebase, which offered a more cost-effective solution for hosting our web services.

### 3.2. Wages

Using estimations of salaries for junior software engineers (Glassdoor, 2024c), robotics engineers (Glassdoor, 2024b), and product managers in the UK (Glassdoor, 2024a), we interpolate an average hourly wage of £14.71/h for each time member. We spent a total of 1560 hours. Totalling £22,947.6 in wages for the members. Thus, accounting for labour costs, monetary budget usage, and the cost of the robotic unit, our total system's budget would be £24538.07.

### 3.3. Commercialisation

CleanCAPS, in its current form, serves as a strong proof of concept with clear potential for future commercialisation. While the prototype is not yet ready for direct deployment, it successfully demonstrates the feasibility of autonomous keyboard cleaning in shared environments. To make this product satisfy the vision of our team, we will need to scale the product significantly. This would involve replacing the TurtleBot with a taller, more powerful mobile base capable of navigating real-world obstacles such as chairs, cables, and dynamic desk layouts. Enhanced sensors, cameras, and more advanced SLAM would enable real-time environmental adaptation, making the product suitable for dynamic office and institutional settings. CleanCAPS in its current form offers a promising starting point for developing a product that enhances hygiene in shared workspaces.

# A. Appendix

## A.1. Competition

The current market for keyboard cleaning is dominated by manual tools such as compressed air cans, brushes, and gel putties. While these products can help remove surface debris, they rely entirely on user effort and offer inconsistent results. More importantly, they provide no disinfection, making them unsuitable for environments where hygiene is a priority.

Mini portable vacuum cleaners offer improved dust removal, often including small brushes for crevices. However, they are still handheld and require careful operation. Improper use can potentially damage keyboards, and the cleaning process remains time-consuming and user-dependent.

The most advanced competitor is the Vioguard Defender (Vioguard, 2025), a self-sanitizing keyboard with a built-in UV-C light. While effective at disinfecting surfaces, it is limited in scope: it works with only one specific keyboard, cannot remove dust or debris, and costs around £2000 per unit.

In contrast, CleanCAPS offers a fully autonomous, scalable solution. It combines debris removal with UV-simulated disinfection, navigates between keyboards without user input, and supports real-time interaction through a web app and admin dashboard. These features position CleanCAPS as a practical and innovative alternative filling the gap in the market.

## A.2. UV Regulations and Compliance

The LED strips currently used in CleanCAPS are WS2813 models, which emit light within the 467-622 nm (Shenzhen Normand Electronic Co., 2016) wavelength range. This range falls entirely within the visible light spectrum and is considered safe for human exposure under UK regulations, specifically The Control of Artificial Optical Radiation at Work Regulations 2010(Government, 2010). Our current lighting setup, with the LED lights angled downwards and are shielded to avoid direct exposure to users. does not pose any UV-related safety risks. An essential feature we implemented is that when the object detection model detects with 95% confidence a human, it will turn off to ensure that the detected human is not exposed to UV light (simulated by the LED). This percentage can be decreased to ensure more prompt turning off of the mechanism when a human is detected in case actual UV-C light is used.

If there was a potential to expand into using UV, the commercially available UV light strips of comparable size to our LED module emit at 395-400 nm (Gove, 2010), which falls within the UV-A range. Under the current set up of our robot like the downward-facing, enclosed light sources and the system design of short exposure periods, these UV-A strips are considered safe.

This however will not be the case if the system were to use shorter wavelengths (namely UV-C), stricter compliance would be required. Observe figure 6 that shows the exposure limits (ELs) in accordance to IEC 62471(CENELEC, 2008) for irradiance at the surface of the skin or cornea. The system would fall under the "Actinic UV" hazard classification. Any exposure to light in the 200-400 nm range must be limited to below 30/t $W/m^2$.

## A.3. Summary of exposure limits (ELs) from IEC 62471 table

**Table 5.4** Summary of the ELs for the surface of the skin or cornea (irradiance based values)

| Hazard Name | Relevant equation | Wavelength range nm | Exposure duration sec | Limiting aperture rad (deg) | EL in terms of constant irradiance $W \cdot m^{-2}$ |
|---|---|---|---|---|---|
| Actinic UV skin & eye | $E_S = \sum E_\lambda \cdot S(\lambda) \cdot \Delta\lambda$ | 200 – 400 | < 30000 | 1,4 (80) | 30/*t* |
| Eye UV-A | $E_{UVA} = \sum E_\lambda \cdot \Delta\lambda$ | 315 – 400 | ≤1000<br>>1000 | 1,4 (80) | 10000/*t*<br>10 |
| Blue-light small source | $E_B = \sum E_\lambda \cdot B(\lambda) \cdot \Delta\lambda$ | 300 – 700 | ≤100<br>>100 | < 0,011 | 100/*t*<br>1,0 |
| Eye IR | $E_{IR} = \sum E_\lambda \cdot \Delta\lambda$ | 780 –3000 | ≤1000<br>>1000 | 1,4 (80) | $18000/t^{0,75}$<br>100 |
| Skin thermal | $E_H = \sum E_\lambda \cdot \Delta\lambda$ | 380 – 3000 | < 10 | 2π sr | $20000/t^{0,75}$ |

*Figure 6.* Summary of exposure limits (ELs) from IEC 62471 for irradiance at the surface of the skin or cornea.

## A.4. Area map

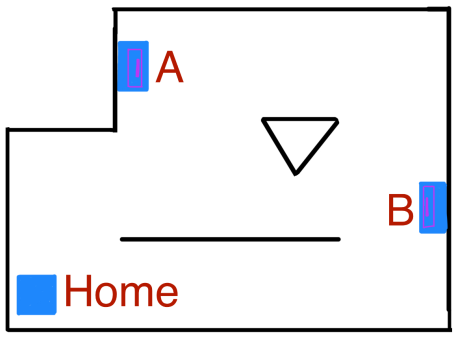


*Figure 7.* Area mapping

## A.5. User Suggestions for the product word cloud

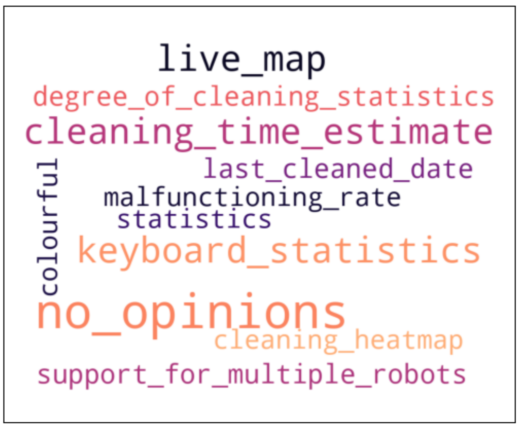


*Figure 8.* User Suggestions for the product

### A.6. User suggestions for the cleaning head word clouds

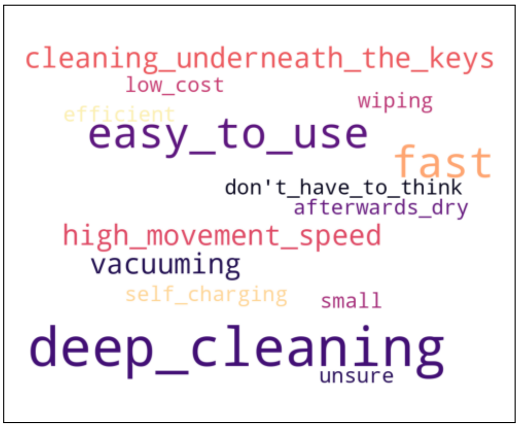


*Figure 9.* User suggestions for the cleaning head

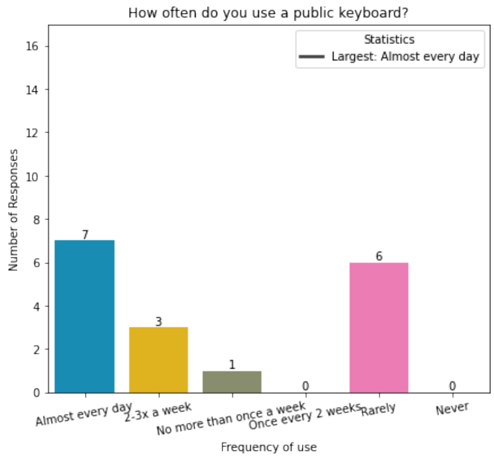


*Figure 10.* How often do you use a public keyboard?

### A.7. Error of Turtlebot movement table

| Trial ($n$) | Expected Distance ($m$) | Distance Moved ($m$) | Deviation ($m$) |
|---|---|---|---|
| 1 | 1 | 0.945 | 0.055 |
| 2 | 1 | 0.985 | 0.015 |
| 3 | 1 | 0.987 | 0.013 |
| 4 | 1 | 0.985 | 0.015 |
| 5 | 1 | 0.993 | 0.007 |
| 6 | 1 | 0.983 | 0.017 |
| 7 | 1 | 0.970 | 0.030 |
| 8 | 1 | 0.980 | 0.020 |
| 9 | 1 | 0.979 | 0.021 |
| 10 | 1 | 0.978 | 0.022 |
| 11 | 2 | 1.991 | 0.009 |
| 12 | 2 | 1.962 | 0.038 |
| 13 | 2 | 1.955 | 0.045 |
| 14 | 2 | 1.969 | 0.031 |
| 15 | 2 | 1.951 | 0.049 |
| 16 | 2 | 1.964 | 0.036 |
| 17 | 2 | 1.969 | 0.031 |
| 18 | 2 | 1.966 | 0.034 |
| 19 | 2 | 1.959 | 0.041 |
| 20 | 2 | 1.968 | 0.032 |

*Table 9.* Error of Turtlebot movement

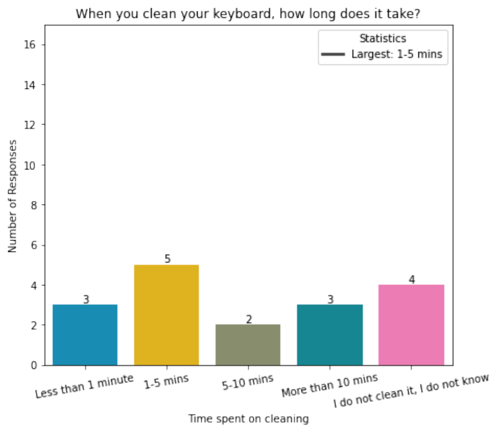


*Figure 11.* How long does it take to clean you keyboard?

### A.8. Demo 2 User Survey Analysis

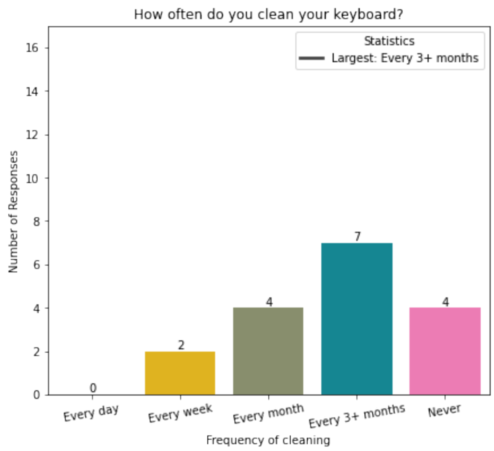


*Figure 12.* How often do you clean your keyboard?

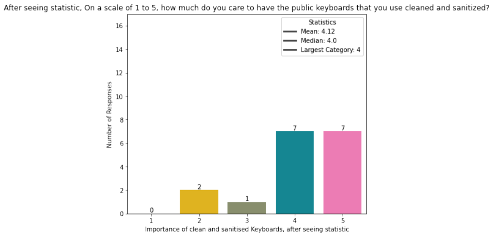


*Figure 13.* How much do you care to have the public keyboards that you use cleaned and sanitised?

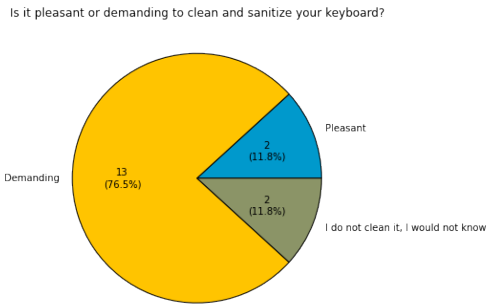


*Figure 14.* Is it pleasant or demanding to clean and sanitise your keyboard?

## A.9. Testing Data Analysis

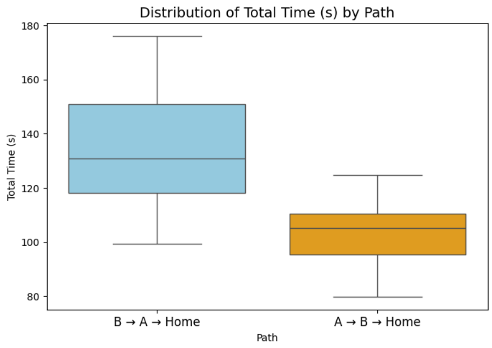


*Figure 15.* Distribution of Total Time (s) by Path

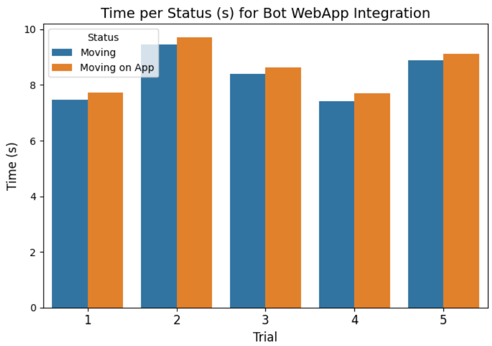


*Figure 16.* Time Per Status

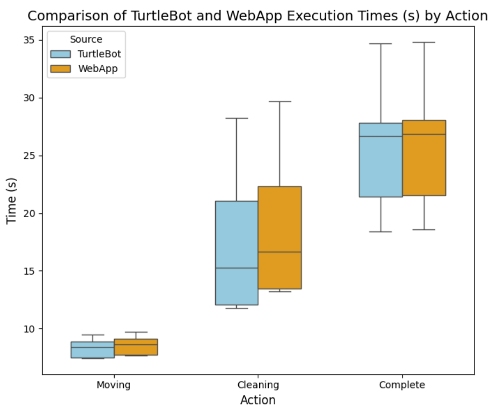


*Figure 17.* Comparison Of Turtlebot and Webapp Execution Times (s) by Action

## References

CENELEC. Photobiological safety of lamps and lamp systems, 2008. URL https://www.lisungroup.com/wp-content/uploads/2020/02/EN62471-2008-Standard-Free-Download.pdf?srsltid=AfmBOopKEaoqKzqAZEsT6I_QnIPxlMzy2B-TmHtTmqCq_kjxBsyncX-V. Accessed: 10 Mar. 2025.

Glassdoor. Junior product manager salary estimates, 2024a. URL https://www.glassdoor.co.uk/Salaries/junior-product-manager-salary-SRCH_KO0,22.htm. Accessed: 14 Feb. 2025.

Glassdoor. Junior robotics engineer salary estimates, 2024b. URL https://www.glassdoor.co.uk/Salaries/junior-robotics-engineer-salary-SRCH_KO0,24.htm. Accessed: 14 Feb. 2025.

Glassdoor. Junior software engineer salary estimates, 2024c. URL https://www.glassdoor.co.uk/Salaries/junior-software-engineer-salary-SRCH_KO0,24.htm. Accessed: 14 Feb. 2025.

Gove, UK. The control of artificial optical radiation at work regulations 2010, 2010. URL https://www.legislation.gov.uk/uksi/2010/1140/contents. Accessed: 10 Mar. 2025.

Government, UK. The control of artificial optical radiation at work regulations 2010, 2010. URL https://www.legislation.gov.uk/uksi/2010/1140/contents. Accessed: 10 Mar. 2025.

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

Lin, Tsung-Yi, Maire, Michael, Belongie, Serge, Bourdev, Lubomir, Girshick, Ross, Hays, James, Perona, Pietro, Ramanan, Deva, Zitnick, C. Lawrence, and Dollár, Piotr. Microsoft coco: Common objects in context, 2015. URL https://arxiv.org/abs/1405.0312.

Shenzhen Normand Electronic Co., Ltd. Ws2813: Intelligent control integrated led light source, 2016. URL http://www.normandled.com/upload/201605/WS2813%20LED%20Datasheet.pdf. Accessed: 10 Mar. 2025.

Vioguard. Vioguard defender, 2025. URL https://www.vioguard.com/product/vioguard-defender/#:~:text=DefenderâĎć%20uses%20the%20germicidal,and%20disposal%20of%20biohazard%20waste. Accessed: 10 Mar. 2025.