Product: UPLIFT.ED

Team: UPLIFT

Student: Michael Aylesbury

Abstract

Team UPLIFT designed the product UPLIFT.ED, referred to as "ED" (Elevation Device). ED is an assistive robot designed primarily to aid mobility impaired consumers reach low down drawers by elevating an inner basket of a cabinet drawer to counter-top height. Working as part of the app design team, my main contributions consisted of designing the initial user interface for the app, implementing the front-end and back-end coding of the app, and designing graphic assets for the team. Throughout the process my team and I learned how to effectively work as a team, learned how to use ROS and Webots, and how to connect the app to Webots using The Standard ROS JavaScript Library.

1. Contribution

My key contribution to the team was working on the app development primarily using React [1]. The task completion criteria requires that users can use the app to control ED by selecting a cabinet to open, which ED then lines up with, opens, and lifts the basket to counter-top level. This requires that the app must be able to communicate with ED through a back-end implementation. I was in close communication with Tech Lead Rafael Anderka, who helped bring everything together in a professional, presentable, industry standard level. I began by designing the initial home page mock-up interface design, and worked with Rafael to develop this with a cabinet render.

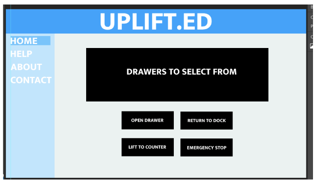


Figure 1. Interface mock-up design

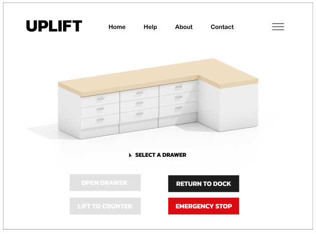


Figure 2. Developed interface mock-up design

The task consisted of implementing four pages- Home, Help, About, and Contact- as well as a navigation bar and drop down menu (represented by the hammer icon).

Following the home page implementation, I created the navigation bar, and began filling out the other pages with content before styling.

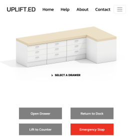
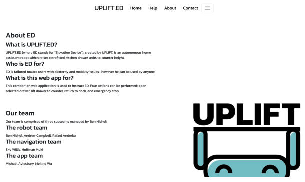


Figure 3. Interface design implemented in React



About ED

What is UPLIFT.ED?

UPLIFT.ED (where ED stands for "Elevation Device"), created by UPLIFT, is an autonomous home assistant robot which raises retrofitted kitchen drawer units to counter height.

Who is ED for?

ED is tailored toward users with dexterity and mobility issues- however he can be used by anyone!

What is this web app for?

This companion web application is used to instruct ED. Four actions can be performed: open selected drawer, lift drawer to counter, return to dock, and emergency stop.

Our team

Our team is comprised of three subteams managed by Ben Nichol:

The robot team

Ben Nichol, Andrew Campbell, Rafael Anderka

The navigation team

Sky Willis, Hoffman Muki

The app team

Michael Aylesbury, Meiling Wu

Figure 4. Beginning to fill out page with content

Then, I focused on the presentation of this page. We initially used a two colour scheme of black and white (later changed to purple and orange), and I added a scrolling animation to the text on every page other than home, fading in and enlarging the upcoming paragraph, while scrolling from the current paragraph (note a Lorem Ipsum Latin generator [2] was used for place holder content for testing).

Figure 5. Initial implementation of scrolling animation

With a functioning first implementation of the front-end design, the focus then moved to the back-end design. I was tasked to set up communication between the app and ED in Webots[3] (which made use of The Robot Operating System (ROS) library[4]). I worked on the app side of the communication, working with Sky Willis, a developer from the Navigation Team, who set up the Webots side of the communication. We initially tried using rosbridge[5], which did not work with our environment due to dependency version mismatches, despite attempts to backdate and fix the compatibility. We instead decided to use The Standard ROS JavaScript Library (roslibjs)[6]. This allowed for successful communication between the app and ED, paving the path for fully integrating our final product.

I also worked on graphic design for the team. I designed the initial logo we used in the first week, and went on to design a visualisation of our team hierarchy and sub-teams.

I created two posters for the fair; one for the app and one for the hardware of ED.

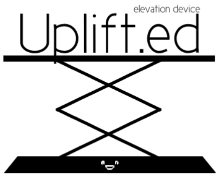


Figure 6. Initial logo design featuring ED chassis primitive

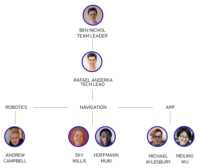
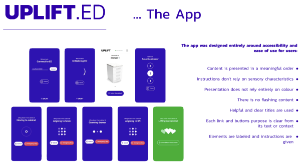


Figure 7. Team hierarchy



The app was designed entirely around accessibility and ease of use for users:

- Content is presented in a meaningful order
- Instructions don't rely on sensory characteristics
- Presentation does not rely entirely on colour
- There is no flashing content
- Helpful and clear titles are used
- Each link and buttons purpose is clear from its text or context
- Elements are labeled and instructions are given

Figure 8. App poster

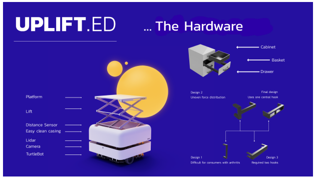


Figure 9. Hardware poster

The hardware poster was designed to show off ED's design with a labelled render, as well as the development of our custom drawers and their handle design. The app poster shows photos of the different states of the app, and displays the accessibility of the app and its design. Finally, I designed a document explaining the styling choices used for our graphic design; the fonts we used, logos we used, and our colour palette.

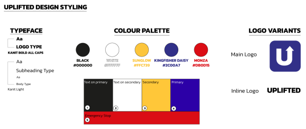
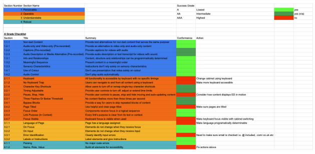


Figure 10. Design styling document

I worked on the risk assessment for ED, and also for the team. The assessment for ED considers the possible hazards that could arise from usage, who and/or what would be affected (and how), as well as risk control. For the team assessment I considered risks in within our team, how they would affect our team, and how we could mitigate these risks.

As ED is an assistive robot for users who may be elderly or suffer from disabilities, accessibility for the app was a huge consideration in its creation and design. The aim was to make the app as simple and accessible as possible, while still reserving functionality and remaining aesthetically pleasing. I researched the WCAG guidelines for accessibility[7], as well as a checklist[8] and compiled a checklist of the guidelines and how we met/did not meet these, and the changes that must be made to meet these.



Section Number	Section Name	Success Grade			
1	Perceivable	A	Lowest	🟩	yes
2	Operable	AA	Intermediate	▩	yes (n/a)
3	Understandable	AAA	Highest	🟥	no
4	Robust				

A Grade Checklist

Section	Title	Summary	Conformance	Action
1.1.1	Non-text Content	Provide text alternatives for non-text content that serves the same purpose	🟩	
1.2.1	Audio-only and Video-only (Pre-recorded)	Provide an alternative to video only and audio-only content	🟩	
1.2.2	Captions (Pre-recorded)	Provide captions for videos with audio	▩	
1.2.3	Audio Description or Media Alternative (Pre-recorded)	Provide audio description or text transcript for videos with sound.	▩	
1.3.1	Info and Relationships	Content, structure and relationships can be programmatically determined	🟩	
1.3.2	Meaningful Sequence	Present content in a meaningful order	🟩	
1.3.3	Sensory Characteristics	Instructions don't rely solely on sensory characteristics	🟩	
1.4.1	Use of Colour	Don't use presentation that relies solely on colour	🟩	
1.4.2	Audio Control	Don't play audio automatically	🟩	
2.1.1	Keyboard	All functionality is accessible by keyboard with no specific timings	🟥	Change cabinet using keyboard
2.1.2	No Keyboard Trap	Users can navigate to and from all content using a keyboard	🟥	Make more keyboard accessible
2.1.4	Character Key Shortcuts	Allow users to turn off or remap single-key character shortcuts	▩	
2.2.1	Timing Adjustable	Provide user controls to turn off, adjust or extend time limits.	▩	
2.2.2	Pause, Stop, Hide	Provide user controls to pause, stop and hide moving and auto-updating content	🟥	Consider how content displays ED in motion
2.3.1	Three Flashes Or Below Threshold	No content flashes more than three times per second	▩	
2.4.1	Bypass Blocks	Provide a way for users to skip repeated blocks of content	▩	
2.4.2	Page Titled	Use helpful and clear page titles	🟥	Make sure pages are titled
2.4.3	Focus Order	Components receive focus in a logical sequence	🟩	
2.4.4	Link Purpose (In Context)	Every link's purpose is clear from its text or context	🟩	
2.4.7	Focus Visible	Keyboard focus is visible when used	🟥	Make keyboard focus visible with cabinet switching
3.1.1	Language of Page	Page has a language assigned	🟥	Make language programatically determinable
3.2.1	On Focus	Elements do not change when they receive focus	🟩	
3.2.2	On Input	Elements do not change when they receive input	🟩	
3.3.1	Error Identification	Clearly identify input errors	🟥	Need to make sure email is checked i.e. @ included, .com/.co.uk etc
3.3.2	Labels or Instructions	Label elements and give instructions	🟩	
4.1.1	Parsing	No major code errors	🟩	
4.1.2	Name, Role, Value	Build all elements for accessibility	🟥	Fix actions above

Figure 11. WCAG accessibility checklist

2. Lessons learned

Throughout the course I gained many skills and learned a lot about myself. I gained a lot of experience using React- it was my first time doing a full implementation using it. Still being relatively new, I initially struggled with using JavaScript in such an environment for creating visual elements and event handlers- but, I found it exciting making progress and seeing the app come to life. If I were to design the app again I would focus early on on preparing the app for distribution on devices other than just a computer. I initially designed the app primarily as a website rather than a cross platform app. I believe this showed my lack of experience, and Tech Lead Rafael Anderka greatly assisted in preparing the app to resize, while retaining functionality, to work effectively on personal computers, tablets, and mobile devices. I learned a lot from his testing for other devices and would be able to effectively implement these changes upon design of another app. I also gained skills using Overleaf[9] for report generation, which will be a greatly helpful skill to have, as well as project management using ClickUp[10]. We heavily used Microsoft Teams[11] for communication, and as well as file sharing. It worked very effectively, and I would consider it as a go-to for future communications.

On one occasion I found myself in a position where I was unable to contribute to a task I agreed to help with, double booking myself with a coursework I had due at the same time. Feeling like I had let the team down, this was one of the biggest lessons that I gleaned from this task- we all had coursework due throughout the semester but that was not an excuse to not complete work. Learning to balance this course with my others was a learning curve for me, but I have come out the other end with fantastic organisational skills, and a better understanding of the time frame I can complete tasks in and how to plan my tasks. Before, I had more of a sporadic unplanned system, without properly taking the time to form a structure.

While I use GitHub[12] regularly for version control, I discovered that I did not have as strong a grasp of its use in a regularly updated team repository. This required me to learn more about the syntax, and delve into stashing and merging code when there were global changes to the project, but also local changes on my system. These skills will be well applied in any and all future development as part of a team.

While initially nervous, I found that I really enjoyed working as part of a team. I was able to effectively contribute, and as a team we kept to the deadlines we made for development. That being said, as a team we struggled with the report deadlines. If we were to do the course over again, we would not underestimate the time writing a sound report takes. I was proud of our communication and preparedness for client meetings; we would meet before hand, anticipating questions that may arise and prepare answers.

We also underestimated how long the website would take to complete. We aimed to (and did) create an informative, accessible website that covered any questions that may arise about the product, and covering technical details. In hindsight this is a large undertaking, and took much longer than we planned it taking. On the final day before submission we had an eight hour Microsoft Teams meeting to complete the website. While we finished with a website we were proud of and fulfilled our aims, had we completed the task earlier we could have polished it further and saved ourselves from the stress of last minute completion. Had any major problems arisen, we would have been in a difficult position that otherwise would have been avoidable.

References

1 https://reactjs.org/ 2 https://loremipsum.io/
3 https://cyberbotics.com/
4 https://www.ros.org/
5 http://wiki.ros.org/rosbridge_suite
6 http://wiki.ros.org/roslibjs
7 https://www.w3.org/WAI/standards-guidelines/wcag/
8 https://www.wuhcag.com/wcag-checklist/
9 https://www.overleaf.com/
10 https://clickup.com/
11 https://www.microsoft.com/en-gb/microsoft-teams/group-chat-software
12 https://github.com/