|  |  |  |
| --- | --- | --- |
| **Speaker** | **Slides** | **Script** |
|  | Slide 1 | Good afternoon everybody, we’re group B23 and we’ll be presenting how we went through the design process to create Elevaider |
|  | Slide 2 | The first step we took in designing was to use the problem refinement loop |
| Hiruni  ------------  Gianluca | Slide 3 | **Needs:** For this, we conducted client meetings to find statements that we could convert into a list of needs. The main needs we focused on were reducing the need to touch surfaces, being usable with 1 hand *(since the other hand is usually preoccupied with cane, etc)*, notifies audible or by vibration, has a simple interface and is affordable. From these needs, we developed our problem statement.  **Problem statement:** Our problem required a device that would help people with low or no visibility locate accessible buttons. This would prevent them from having to touch public surfaces to find the button’s location and reduce the risks associated with it. |
| Thuy-Vi | Slide 4 | **Benchmarking:** We then researched other products that tackled these needs, such as Key2Access, SeeingAI, and Portal. These products we benchmarked against are products that ease daily activities, like crossing at a crosswalk or finding accessibility buttons, for people with low vision.  **Metrics:** We then came up with the following metrics to measure each product against. By benchmarking, we can lay out the parameters for our own product. |
| Gabriel | Slide 5 | **Target Specifications:** After developing our list of metrics, we created our target specification, shown here, by setting specific values for each metric of the ideal value, assuming the best possible results, and a set of values that we set as an acceptable limit.  This process allowed for more flexibility in evaluation of our design and made sure we wouldn’t get bogged down by a single given metric not achieving the exact right value. |
| Hiruni | Slide 6 | Now with these values in hand, we moved on to the persistence loop |
| Hiruni | Slide 7 | Initially we were going to go with a remote design. This would have been an external device placed near the button (as seen in the diagram below) and an app would tell the device to press the button.We decided against this idea because of feasibility issues with needing legal permission to place devices on public or private property.  So we went back to the problem refinement loop, to narrow down our scope to make the problem more manageable for 4 months. Our new scope focused solely on elevator buttons and would be more software based.    With this in mind, the navigation option was created. It’s an app that uses the user’s phone camera to detect buttons. Then it would give directions using either an external phone attachment like a joystick, or give vibrational or audible cues. |
| Gianluca  ------------  Thuy-Vi | Slide 8 | **Technical**: each of us had to research on Swift syntax, coding in Swift using the Xcode IDE, and learning how to train and test machine learning models with Core ML.  **Economic**: we would have to consider a fee to enroll in Apple’s Developer Program to publish the app on either the app store or TestFlight.  **Legal**: we had to do research on the patents, copyrights, and trademarks of other similar applications.  **Operating:** In terms of operational feasibility, there weren’t too many challenges. We’re all working remotely so we established a mostly reliable method of communication through Discord. One challenge we did face was with the resources we had. We needed to develop an IOS app, which can only be developed using a MacOS, and the team only has access to two. So, we needed to account for the fact that the whole team couldn’t contribute to the coding.  **Scheduling:** As for scheduling, all of us are full time students so we have to take into consideration time for our other courses. So, we came with a schedule to have 2 team meetings each week, Fridays for assigning tasks and Mondays for check-ins. |
| Gabriel  -----------  Tony | Slide 9 | **Navigation with Joystick**  One of our original concepts incorporated a physical device that would have a “joystick” attached and would be connected to the phone app. The basic idea was that the phone would locate the button using some sort of image recognition software and then the joystick would physically demonstrate the correct direction to go by pointing in that direction.  There were a few major problems with this solution:  First, developing an app with object recognition and locating abilities to find the button,a device capable of moving in a specific direction, and a bluetooth connection to connect them was, especially knowing what we do now, way too much  Second, There was no practical way for the person to both carry this and they’re phone around while only using 1 hand and we wanted it to be 1-handed  Third, It wouldn’t be a very good indicator anyway because it could be interpreted ambiguously depending on the person’s perception of where it is pointing.  **Navigation with audio**:   1. Another concept we came up with was audio notification. When the camera of our device or phone detects a button, an audio cue (like “a button is in view”) will play to notify the user. 2. Originally our plan was going to be based on the location of the button in the camera feed, it would have guided the user with specific instructions towards the button of interest 3. The only issue is that our client wanted to have a device instead of holding their phone out in public |
| Tony | Slide 10 | 1. Guide to button, camera    1. This would allow us to focus on software development only instead of separating and making hardware as well 2. Decisions made    1. Focus on elevator buttons    2. Focus on LRT button specifically |
| Hiruni  ---------  Gianluca | Slide 11  *\*\*add pictures of the testing phases\*\** | * **Low fidelity:** We used the low fidelity prototype to check our assumption that audible cues would be a good form of navigation. To do so, we had a test user turn on their phone camera and then blind folded them. The goal was for them to locate a door handle using just the audible cues we would give them when we could see the handle on screen. The user was able to find it in roughly 2 mins so our assumption was proven. * **Medium fidelity:** We then moved on to the medium fidelity prototype where we focused on creating our first model to recognize buttons. We then tested this model with our images taken of LRT buttons. Unfortunately, It didn’t recognize all the test images so we knew we had to retrain the model for the high fidelity prototype. * **High fidelity**: Our high fidelity testing consisted of our final Core ML machine learning model and the app’s functionality tested locally on an iOS device. |
| Gianluca | Slide 12 | Our demo, as the video will demonstrate, displays the app functionality including opening the app, scanning the environment, and LRT button recognition with audible notification to the user |
| Thuy-Vi | 13 | This project gave us real world experience, or in other words, plenty of challenges |
| Thuy-Vi  ---------  Gabriel | Slide 14 | **Lessons learned 1:** Our first major problem we encountered was feasibility. Our initial concept, a device installed on a button to push it, was simply not possible within our time frame, and would require multiple approval processes with the government and with businesses in order to install these kinds of devices.  So after a discussion with our professor and project managers, we realized we needed to develop a product that won’t be installed on the button itself. So we went through the ideation process again, and came up with our current concept, Elevaider.  From this, we learned how to better scope projects and how to set more feasible goals.  **Lessons learned 2:**  Initial model couldn't recognize elevator buttons accurately:  A significant issue we encountered during the early development of our final prototype was that our model couldn't recognize elevator buttons accurately. Though we’d fed it several hundred images of buttons, the issue stemmed from the fact that not only were some of the pictures too far away or at an odd angle so the ai couldn’t recognize the button in them, but also the large amount of variation between models of elevator buttons made for inconsistent parameters to track.  Our solution was to narrow the scope of our design to work exclusively with the LRT model elevator button and only use our clearest images of the button to avoid any confusion. This did mean our design was limited only to the LRT, but the LRT is a frequently used form of transportation around key locations in the city and as a result it should still be helpful and useful to a large enough number of people. |
| Tony | 15 | Future work includes expanding to other buttons and improving accuracy   1. Expanding to other buttons   -Larger variety of elevator, accessibility, crosswalk buttons  -Being able to have a higher confidence rate in identifying buttons of any sort with the capability of spotting them from a further distance   1. Incorporating an XY coordinate system   -This will be able to give the user more detailed direction for the user to find a button |
|  | 16 | Thank you and any questions? |