**The Reduce Advocates Co.**

**Project Description:**

A sustainability-driven app promoting conscious consumption and plastic waste reduction: **The Reduce Advocates Co.** is a prototype mobile platform born from firsthand environmental volunteering experiences. After witnessing an overwhelming amount of plastic waste, much of it from the same brands, this app was envisioned to encourage consumers to reduce, reuse, and recycle through an incentive-based system. By partnering with major consumer brands, the app rewards users for properly disposing of recyclable waste, especially plastic packaging from canned drinks and food products.

More than just a rewards system, **The Reduce Advocates Co. is built on the belief that change begins within,** and that collective cooperation is essential. Without active participation, the cycle of wasteful consumption will continue unchecked. The app aims to make sustainability not only accessible but personal, inspiring users to become advocates for responsible consumption and environmental stewardship in their own communities.

To ensure wide impact, the platform is designed to be user-friendly and accessible to nearly anyone with a mobile phone making it easy for people of all backgrounds and tech skill levels to participate in sustainable actions, anytime and anywhere.

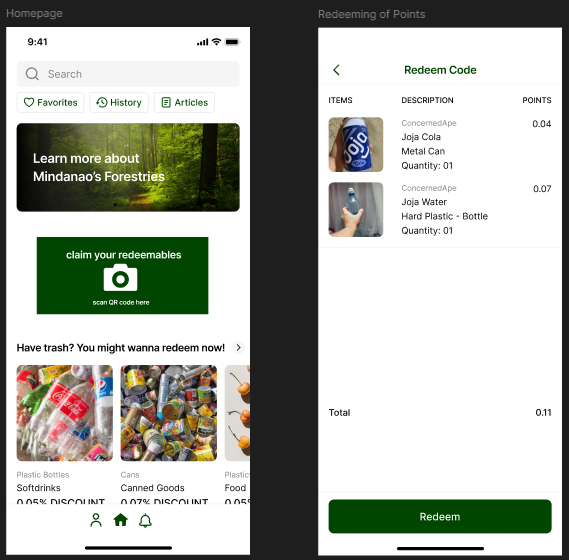
**System Requirements:**

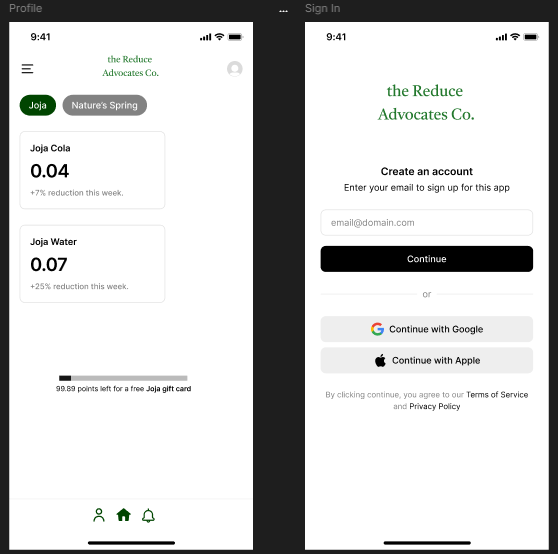
|  |  |
| --- | --- |
| Compatibility: | Android Marshmallow and IOS 15 and above |
| Required storage: | 100 mb and higher |
| Memory (RAM): | 2gb or higher |
| Internet: | Requires internet to access the software |
| Bluetooth: | Does not support bluetooth connection |

**Prototype Description:**   
   
The prototype developed using Java and coded entirely within Visual Studio Code. It was built with a focus on environmental advocacy through digital innovation, transforming everyday recycling habits into a gamified, user-friendly experience. The inclusions of its key interface components are:

* **Homepage (Main Dashboard)**
* **QR Code Redemption**
* Point Redemption Page
* **Brand Profile Page**
* **Sign-In Interface**
* **Navigation Bar (Bottom UI)**
* **User Progress Indicator**
* **Article & Educational Section**  
     
     
  Akiya, a laid-back soda lover loyal to his favorite brand, often found himself tossing bottles aside. But with a growing urge to change his habits and at least start disposing of waste properly, he stumbled upon The Reduce Advocates Co. app, which turned his small effort into a rewarding step toward sustainability.

**The Reduce Advocates Co. Prototype**

****



**Rationale:**   
I decided that prototype was developed using Java due to its platform independence, strong object-oriented design principles, and mature ecosystem of libraries and tools. As a general-purpose language widely adopted in mobile development, especially Android, Java offers a stable foundation for building scalable, maintainable, and performance-efficient applications. Its compatibility with SQLite and GUI libraries also makes it suitable for constructing interactive prototypes within lightweight IDEs like Visual Studio Code.Key Technical Advantages of Java for This Project are:

* **Platform Independence (Write Once, Run Anywhere):**  
  Java applications compiled to bytecode can run on any device with a Java Virtual Machine (JVM), making it a strong candidate for cross-platform compatibility.
* **Strong OOP (Object-Oriented Programming) Support:**  
  Java’s class-based architecture allows for clean, modular code that is easy to maintain, extend, and debug, essential for scalable features like reward systems, user profiles, and brand-specific tracking.
* **Robust Library Ecosystem:**  
  Java offers access to a wide array of libraries for database integration (e.g., JDBC for SQLite), GUI development (e.g., Swing/JavaFX for prototypes), and data management, speeding up development while ensuring code reliability.

**Changes to the Requirements:**

* **Added:**
  + User progress tracker and reward threshold indicator
  + Brand-specific waste tracking and sustainability stats
  + Multi-option login (email, Google, Apple) for accessibility
* **Removed:**
  + Placeholder data screens (replaced with functional point breakdown interface)
  + Generic recycling info pages (replaced with localized, brand-aware articles)
* **Refined:**
  + GUI layout using Java Swing for smoother navigation and visual clarity
  + Reward system logic for accurate point assignment based on item type
  + Sign-in flow for improved user experience across devices

**Initial Evaluation Plan:**   
The evaluation of the application prototype will be conducted on-site under controlled conditions that replicate typical user environments. Test participants will interact with the system by performing common tasks, such as scanning items, navigating through features, and redeeming points while evaluators observe and document the system’s responses in real time. The assessment will be guided by three core components: Usability Specifications, which measure the app’s efficiency and user-friendliness; Heuristic Evaluation, which examines interface design against established usability principles; and Participant Surveys and Feedback, which gather subjective user experiences and improvement suggestions.

**Usability Specifications**   
The system is expected to meet the following criteria during testing:

* **Learnability:** How easily a new user can understand how to use the app on their first try.
* **Efficiency:** How quickly users can perform common tasks once familiar with the app.
* **Memorability:** How easily returning users remember how to use the app after time away.
* **Satisfaction:** How enjoyable and satisfying the overall user experience is.
* **Error Tolerance:** How well the system helps users avoid and recover from errors.
* **Population**   
  Targeting 10–15 individuals (peers and/or instructors) They will complete the following test tasks:
* Set up the system and calibrate sensors
* Monitor environmental data from the dashboard
* Test automatic sprinkler activation through simulated dry soil
* Adjust system thresholds and review alert responses

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| * **Developer / UI Designer Member** | | * **Task(s)** | | |
| * Akiya B. Murai | | * Will be recording and guiding the participant simultaneously, and receive feedback afterward. | | |
| Profile Setup | Within 3 minutes or Below | | Highly Acceptable | Successful |
| Above 3 minutes | | Not Acceptable | Unsuccessful |
| Scanning of Product QR | Within 1 minute or Below | | Highly Acceptable | Successful |
| Above 1 minute | | Not Acceptable | Unsuccessful |
| Article Accessibility and Buffering | Within 1 minute or Below | | Highly Acceptable | Successful |
| Above 1 minute | | Not Acceptable | Unsuccessful |
| Redemption Recognition | Within 2 minutes or Below | | Highly Acceptable | Successful |
| Above 2 minutes | | Not Acceptable | Unsuccessful |

**Heuristic Evaluation**   
The evaluation of my Incentive System will follow the 10 Usability Heuristics for User Interface Design. 

1. **Visibility of System Status**The app should provide immediate feedback after user actions — such as confirming successful item scans, showing point increases, or updating reward progress.
2. **Match Between System and the Real World**Terminology like “scan,” “redeem,” “Joja Cola,” and “gift card” should reflect real-world user language and brand familiarity to make the experience intuitive.
3. **User Control and Freedom**Users should be able to undo mistakes (e.g., remove a wrongly scanned item) and freely navigate between screens without being trapped in a specific flow.
4. **Consistency and Standards**Icons, navigation bars, and menu behaviors should remain consistent throughout the app (e.g., same bottom nav bar icons on all pages).
5. **Error Prevention**The system should guide users to avoid mistakes — for example, by validating QR codes before accepting them, or warning before final redemption.
6. **Recognition Rather Than Recall**Important functions (like scan, redeem, view rewards) should always be visible, with familiar icons and labels so users don’t need to remember how to find them.
7. **Flexibility and Efficiency of Use**Frequent users should benefit from quick actions (e.g., recent scans, saved brands) while new users can follow a slower, guided path if needed.
8. **Aesthetic and Minimalist Design**The interface should remain clean and free of clutter, using calm eco-friendly colors, readable fonts, and only essential information per screen.
9. **Help Users Recognize, Diagnose, and Recover from Errors**When an error occurs (like a failed scan), the app should clearly explain what went wrong and suggest how to fix it (e.g., “This code is unreadable, please try again”).
10. **Help and Documentation**Even in a user-friendly app, optional help sections (like a FAQ or “How it works” guide) should be available for users who need additional support.

**DATA GATHERING METHOD**   
**DESCRIPTION**

To assess the usability and user experience of The Reduce Advocates Co. prototype, two primary data gathering methods will be employed:

**Survey (Quantitative)**   
A structured user survey will be used to collect measurable data regarding the usability, effectiveness, and user satisfaction of the app. The survey will consist of Likert-scale questions (1–5 rating**)** covering specific usability heuristics, such as ease of use, visual clarity, learnability, and perceived motivation. The goal is to obtain statistically analyzable insights on how users interact with and perceive the app’s features.

**Feedback (Qualitative)**   
In addition to the survey, open-ended feedback questions will be included at the end of the testing session. This allows users to express their thoughts, concerns, or suggestions in their own words. This method captures deeper insights into the emotional and contextual aspects of using the app that may not be reflected in numeric ratings.

|  |  |
| --- | --- |
| **Question** | **Method of Answer** |
| **Section 1** | |
| Participant Number | Short Answer |
| On a scale of 1 to 5, how would you rate your experience with the app? | 5-Point Scale |
| On a scale of 1 to 5, how was the UI design of the prototype? |
| **Section 2: Features of the Prototype** | |
| Ease of Use | 5-Point Scale |
| Visual Design |
| QR Scan Functionality |
| Point System Clarity |
| Motivation to Reduce Waste |
| Brand Awareness Feature |
| Login/Signup Process |
| Reward Progress Display |  |
| Overall Satisfaction |  |
| **Section 3: Feedback Section** | |
| Your Feedback | Short Answer |

**Interpretation Scale:**

|  |  |  |
| --- | --- | --- |
| **Score** | **Rating** | **Outcome** |
| 5 | Highly Acceptable | Excellent |
| 4 | Acceptable | Meets expectations |
| 3 | Neutral | Needs improvement |
| 2 | Fairly Acceptable | Needs redesign |
| 1 | Not Acceptable | Failed usability |

**Implementation Challenges and Justification:**   
The development of The Reduce Advocates Co*.* presented several implementation challenges, particularly due to its integration of real-time user interaction, point tracking, and brand-based data management within a Java-based environment. Building the prototype in Java using Visual Studio Code required careful handling of UI components through libraries such as Swing, which, while lightweight, imposed limitations in designing mobile-responsive and modern user interfaces. Additionally, simulating QR scanning functionality and dynamic reward progress tracking posed logic-heavy challenges, especially when aligning user actions with accurate data flow and interface feedback. Despite these constraints, Java was chosen and justified for its object-oriented structure, cross-platform compatibility, and reliability in handling data models, making it suitable for building a scalable foundation. Its strong support for modular design made it easier to manage classes for users, items, brands, and rewards, all essential to the app’s core functionality. Overall, while the project faced complexity in synchronizing interaction, data, and usability, the choice of development tools ensured a flexible and maintainable system architecture that can evolve in future iterations.