**Tech Stack**

**Frontend (Client)**

| **Tool/Library** | **Why It Was Used** |
| --- | --- |
| **React.js** | Modern and beginner-friendly frontend library. Supports reusable components and is widely used in real-world jobs. Also integrates well with other tools. |
| **React Router DOM** | Allows easy routing between pages like Home, About, Settings, etc., making the app feel like a native mobile experience. |
| **React Icons** | Adds intuitive icons like hamburger menus, logout, and theme toggles to improve user experience. |
| **React Webcam** | Enables camera scanning directly from the browser (especially useful on mobile). It makes the app interactive and inclusive. |
| **Speech Recognition (react-speech-recognition)** | Adds voice input support — ideal for accessibility and mobile users who may prefer speaking instead of typing. |
| **Tailwind CSS / Custom CSS** | Allows fast, responsive styling with a consistent "eco-friendly" visual identity (green, soft tones, rounded UI). Perfect for mobile-first design. |

**Authentication**

| **Tool** | **Why It Was Used** |
| --- | --- |
| **Firebase Authentication** | Free, easy to integrate, and secure. It allows Email/Password login and handles sessions securely. Good choice for apps without backend-heavy user management. |
| **Forgot Password + Settings Page** | Gives users control over their account. Built-in security re-authentication ensures best practices, as required by Firebase. |
| **Why Firebase?** | Popular in industry. Great resume and interview booster. Also used in real-world mobile/web apps. No server management needed. |

**Backend (Server)**

| **Tool** | **Why It Was Used** |
| --- | --- |
| **Flask (Python)** | Lightweight Python web framework. Easy to connect to AI models and return JSON responses to the frontend. |
| **HuggingFace Transformers + PyTorch** | State-of-the-art pre-trained image classification model (microsoft/resnet-50) enables you to detect objects with high accuracy. HuggingFace is widely used in AI jobs. |
| **Waste Knowledge Base (JSON)** | Stores structured data for common waste items, mapped with category, eco-tip, and reuse ideas. JSON was chosen for simplicity and performance. |

**Additional Tools & Features**

| **Tool** | **Why It Was Used** |
| --- | --- |
| **Axios** | Handles API requests smoothly from frontend to backend. Widely used and beginner-friendly. |
| **Responsive Design** | Designed and tested for Android/iPhone. Uses CSS Flexbox/Grid and mobile-first layouts for real usability. |
| **PWA Support (Coming Soon)** | Makes the app work offline and behave like a real mobile app. Helps showcase cloud-readiness and performance optimization. |

**Tech Fields Covered**

| **Field** | **How SmartBin Applies It** |
| --- | --- |
| **Web Development** | Built using React (frontend) and Flask (backend) – covers full-stack development. |
| **AI/ML** | Uses HuggingFace + PyTorch to classify images in real time. Demonstrates AI integration in real-world apps. |
| **Cloud & Security** | Firebase Authentication covers account security, login flows, and password/email management. |
| **API Development** | RESTful Flask API connects the frontend with AI and data logic. |
| **UI/UX Design** | Focused on intuitive, mobile-friendly interfaces with theme toggle, accessibility, and error handling. |
| **Data Handling** | Uses structured JSON to simulate real-life data management without needing a full database initially. |