**1. Project Setup and Libraries**

* **Create a New Project**: In Android Studio, start a new project with a basic Activity template.
* **Add Dependencies**: You'll need several libraries, including:
  + **Google Maps API**: For map functionalities.
  + **Firebase**: For authentication, database, and notifications.
  + **Retrofit or Volley**: For network requests (if needed).

**2. User Authentication**

* **Firebase Authentication**: Implement user sign-up and login functionalities.
* **User Roles**: Define different user roles (Volunteer, Clean Up Location Owner, Super User).

**3. Google Maps Integration**

* **Setup Google Maps**: Integrate Google Maps in your app. Display a map view with custom markers for clean-up sites.
* **Custom Markers**: Customize markers for different locations. Use icons that reflect the clean-up site status or type.

**4. Location Management**

* **Creating and Managing Locations**:
  + Allow location owners to create and manage clean-up sites.
  + Implement forms for entering details about the clean-up location.
* **Viewing Locations**:
  + Implement functionality for volunteers to view and select clean-up sites on the map.
  + Show detailed information about the site when a marker is clicked.

**5. User Interactions**

* **Registration for Clean-Up**: Enable users to register for a clean-up site.
* **Friends Addition**: Allow users to add friends to their registration.
* **Data Input Post Clean-Up**: Let location owners input data post clean-up, like the amount of waste collected.

**6. Reporting and Analytics (Super User)**

* Implement functionality for super users to run reports on various metrics (number of volunteers, waste collected, etc.).

**7. Additional Features**

* **Search and Filter**: Implement search and filter functionalities to find clean-up sites based on various criteria.
* **Route Finding**: Integrate a feature to find routes from the user's current location to the selected site.
* **Notifications**: Implement notifications for updates or changes in clean-up sites.

**8. Non-Functional Requirements**

* **Code Quality**: Ensure proper use of try/catch blocks and condition checking.
* **Consistent UI**: Maintain consistency in UI design - fonts, colors, images.
* **Navigation and Usability**: Ensure easy navigation and minimal clicks to perform actions.
* **Database and Notifications**: Use Firebase or a local database for storing data and managing notifications.

**9. Testing and Debugging**

* **Unit Tests**: Write unit tests for your logic.
* **UI Tests**: Use Espresso for UI testing.
* **Debugging**: Regularly test and debug your app to fix issues.

**10. Documentation and Submission**

* **Code Documentation**: Comment your code for clarity.
* **Submission Requirements**: Check your assignment requirements for submission formats and adhere to them.

**Additional Resources**

* **Android Developer Documentation**: Refer to [Android Developers](https://developer.android.com/) for detailed guides.
* **Google Maps API Documentation**: For custom implementations with Maps.
* **Firebase Documentation**: For authentication, database, and notification features.