最核心的两个功能：

1. **Sorting Functionality**: In the `FindWeapons` Servlet, users can search for weapons by name and sort the results according to various attributes such as name, physical damage, and magical damage. This sorting functionality relies on database interactions and Java Servlet technology, allowing users to view weapon data in multiple ways to find the specific weapons they need.

   （meaningful sort，就是排序要有意义，要么升序要么降序，我们用的是升序，但是这里有一个细节就是，数值排序和名称排序是两个逻辑，这个要区分，详细的细节在下面Implementation of Sorting Functionality里面）

2. **Restriction on Deleting Default Weapons**: In the `WeaponDeleteServlet`, the system manages the deletion of weapons by validating the entered weapon ID. If an attempt is made to delete certain weapons designated as "non-deletable" (such as default or particularly crucial weapons), the system will prevent this operation and provide appropriate feedback through the user interface. This ensures the integrity and security of critical data in the database.

   （在我们的设计方案中，存在一个默认武器，因为要保证mainhand一直是有武器的状态，所以我们之前的逻辑是，如果手上的武器在数据库里被删除，那么就替换为默认武器，所以我们实现了一个功能让系统默认武器不能被删除）

The sorting functionality in the `FindWeapons` Servlet provides a detailed and flexible approach for users to view weapon data according to their preferences. Here's a closer look at how sorting is implemented based on different parameters:

Implementation of Sorting Functionality

1. **Sorting by Name**:
   - **Comparator Implementation**: The `WEAPON_NAME_COMPARATOR` is used for sorting by name. This custom comparator first extracts numerical parts from the weapon names if present, ensuring that names with numbers are sorted in a natural order (e.g., "Weapon 9" before "Weapon 10").
   - **Secondary Sorting**: If the numerical parts are identical or absent, the sorting falls back to a standard alphabetical comparison of the weapon names. This ensures that weapons are listed in a logical and predictable manner.
2. **Sorting by Item Level**:
   - **Simple Integer Sorting**: Weapons are sorted based on their `itemLevel` attribute using `Comparator.comparingInt(Weapon::getItemLevel)`. This comparator provides a straightforward numerical sort, placing weapons with lower item levels before those with higher item levels.
3. **Sorting by Physical Damage**:
   - **Numeric Sorting**: Similar to item level sorting, the physical damage of weapons is used as a sorting criterion. Weapons are ordered by the `physicalDamage` attribute, allowing users to quickly find the most or least powerful weapons in terms of physical attack capabilities.
4. **Sorting by Magic Damage**:

    - **Numeric Sorting**: This follows the same pattern as physical damage, using the `magicDamage` attribute to order the weapons. This is particularly useful for players interested in weapons that enhance magical attacks.

Technical Details

- **Parameter Handling**: When a user submits a search query through the form on the `FindWeapons.jsp` page, they can select a sorting parameter from a dropdown menu. The selected option determines which of the sorting comparators is applied.
- **Database Query**: Initially, the weapons are fetched from the database based on the user's search criteria (name matching). The SQL query involves a JOIN between the `Weapon` and `Item` tables, filtering by weapon names that contain the input string.
- **Application of Sorting in Java**: After the list of weapons is retrieved from the database, the Java Servlet applies the selected comparator to the list. This is done using the `Collections.sort()` method or the `List.sort()` method, which takes the comparator as a parameter.
- **Dynamic Response**: Once sorted, the weapons are returned to the frontend, where they are displayed in the specified order. This allows for a responsive and user-tailored viewing experience.

By leveraging both backend database operations and frontend user choices, the sorting functionality in the `FindWeapons` Servlet enhances data accessibility and usability, enabling users to efficiently navigate and select weapons based on detailed criteria.