To develop the "FactoryBot" project, you should follow these essential steps:

1. **Project Planning and Requirements Analysis**:
   - Define the project scope, objectives, and requirements.
   - Identify the specific tools to be managed and their locations.
   - Determine voice recognition needs and obstacle avoidance criteria.
   - Assess noise levels and sources within the factory.

2. **Hardware and Software Selection**:
   - Choose appropriate robotic hardware, including sensors, actuators, and a durable chassis.
   - Select the necessary microcontrollers or computing platforms.
   - Identify suitable noise filtering algorithms and voice recognition software.

3. **Robot Design and Construction**:
   - Design the robot's physical structure to accommodate the selected hardware.
   - Implement tank tracks for optimal mobility across factory terrains.
   - Assemble the robot and integrate hardware components.

4. **Voice Recognition Integration**:
   - Develop or integrate a voice recognition system capable of handling noisy environments.
   - Train the system to recognize specific voice commands for tool retrieval.

5. **Obstacle Avoidance System**:
   - Implement obstacle detection sensors (e.g., ultrasonic or LiDAR) to ensure safe navigation.
   - Develop algorithms for real-time obstacle avoidance and path planning.

6. **Noise Filtering Mechanism**:
   - Implement noise filtering algorithms to enhance voice recognition accuracy.
   - Fine-tune the system to filter out factory-specific noise sources.

7. **Adaptive Obstacle Recognition**:
   - Design algorithms to adaptively identify and respond to new obstacles.

- Implement sensor data processing for continuous obstacle monitoring.

8. **Testing and Iteration**:

- Conduct rigorous testing of the robot's functionality in a controlled environment.

- Collect data on its performance, including voice recognition accuracy and obstacle avoidance capabilities.

- Iteratively refine and improve the robot's software and hardware based on test results.

9. **Factory Integration**:

- Deploy the "FactoryBot" in a real factory setting.

- Collaborate with factory personnel to ensure seamless integration into daily operations.

- Monitor its performance and address any issues that arise during operation.

10. **Documentation and Training**:

- Create comprehensive documentation for robot operation, maintenance, and troubleshooting.

- Provide training to factory personnel on how to interact with and utilize the robot effectively.

11. **Continuous Improvement**:

- Establish a mechanism for ongoing monitoring and maintenance.

- Collect feedback from factory users and incorporate improvements as needed.

By following these steps, you can systematically develop and implement the "FactoryBot" project, ensuring its success in enhancing tool retrieval and assistance within industrial environments.

Hardware and Software Selection:

1. **Robotic Hardware**:

- **Sensors**: using a combination of sensors like ultrasonic sensors or LiDAR for obstacle detection and avoidance. These sensors can provide accurate distance measurements and are commonly used in robotics.

- **Actuators**: Selecting reliable actuators for the robot's movement and tool retrieval mechanism. Depending on the payload and mobility requirements, options could include motors or servos.

- **Chassis**: Choosing a robust and durable chassis that can withstand the factory environment's demands. Materials like aluminum or steel are often suitable choices for industrial robots.

2. **Microcontrollers or Computing Platforms**:

- For a balance of power and flexibility, opting for microcontrollers like Raspberry Pi or Arduino. These platforms have strong community support and a wide range of compatible sensors and modules.

- If advanced processing power is required for complex algorithms or deep learning for voice recognition,(I am not sure for now) considering using more powerful single-board computers like NVIDIA Jetson or specialized robotics platforms like ROS (Robot Operating System).

3. **Noise Filtering Algorithms**:

- Implementing noise filtering algorithms in software. Common techniques include spectral subtraction, adaptive noise cancellation, or noise profiling.

- For voice recognition in noisy environments, considering cloud-based solutions like Google Cloud Speech-to-Text or on-device solutions like PocketSphinx.(This part have not worked on it yet)

4. **Voice Recognition Software**:

- For robust voice recognition in noisy environments, considering cloud-based services such as Google Speech Recognition, Amazon Alexa, or IBM Watson Speech to Text. These services offer high accuracy and can handle various accents and languages.

- Alternatively, if privacy or offline operation is a concern, explore on-device solutions like CMU Sphinx (PocketSphinx), which can work without an internet connection.