Based on the scenario you've described, here are the attributes and operations for the classes related to the building company "Build U Like." These attributes and operations are designed to model the relationships and processes within the company's system, such as projects, tasks, materials, staff allocation, costing, and specialized tasks.

### 1. \*\*Project\*\*

\*\*Attributes:\*\*

- `projectID`: Unique identifier for each project.

- `customerID`: The customer associated with the project.

- `supervisor`: The supervisor assigned to the project.

- `startDate`: The scheduled start date for the project.

- `endDate`: The scheduled completion date for the project.

- `status`: Current status of the project (e.g., "Scheduled", "In Progress", "Completed").

- `tasks`: List of tasks associated with the project (tasks may include both in-house and specialist tasks).

- `estimatedCost`: Estimated cost of the project, based on materials and labor.

- `totalCost`: Actual total cost of the project.

\*\*Operations:\*\*

- `addTask(task: Task)`: Adds a task to the project.

- `removeTask(task: Task)`: Removes a task from the project.

- `calculateEstimatedCost()`: Calculates the estimated cost of the project based on the materials and labor.

- `calculateTotalCost()`: Calculates the total cost of the project, including actual labor and material usage.

- `assignSupervisor(supervisor: Supervisor)`: Assigns a supervisor to the project.

- `allocateStaff(staffMember: Staff, task: Task)`: Allocates a staff member to a specific task.

- `getTasks()`: Returns all tasks for the project.

- `getStaffForTask(task: Task)`: Returns a list of staff assigned to a given task.

---

### 2. \*\*Task\*\*

\*\*Attributes:\*\*

- `taskID`: Unique identifier for the task.

- `taskName`: Name/description of the task (e.g., "Laying Drive", "Excavating").

- `scheduledDate`: Date when the task is scheduled to start.

- `duration`: Duration of the task (in hours).

- `materialsUsed`: Dictionary of materials used (e.g., materialID → quantity).

- `labourHours`: Dictionary of staff and their hours worked (e.g., staffID → hours worked).

- `cost`: The cost of performing the task (calculated).

- `project`: The project to which the task belongs.

\*\*Operations:\*\*

- `assignMaterial(material: Material, quantity: float)`: Assigns material and quantity to the task.

- `assignStaff(staffMember: Staff, hoursWorked: float)`: Assigns a staff member to the task along with the hours they worked.

- `calculateCost()`: Calculates the cost for the task based on material cost and staff labor rate.

- `getMaterials()`: Returns a list of materials required for the task.

- `getStaff()`: Returns the list of staff assigned to the task.

---

### 3. \*\*Customer\*\*

\*\*Attributes:\*\*

- `customerID`: Unique identifier for the customer.

- `name`: Name of the customer.

- `address`: Address of the customer.

- `email`: Contact email.

- `phone`: Contact phone number.

- `projects`: List of projects associated with the customer.

\*\*Operations:\*\*

- `addProject(project: Project)`: Adds a new project to the customer's list of projects.

- `getProjects()`: Returns a list of all projects for the customer.

- `getContactDetails()`: Returns the contact details of the customer.

---

### 4. \*\*Supplier\*\*

\*\*Attributes:\*\*

- `supplierID`: Unique identifier for the supplier.

- `name`: Name of the supplier.

- `contactDetails`: Contact information (e.g., phone, email, address).

- `materialsSupplied`: List of materials supplied by the supplier.

- `materialPrices`: Dictionary of material prices (e.g., materialID → price per unit).

\*\*Operations:\*\*

- `supplyMaterial(material: Material, quantity: float)`: Supplies the given quantity of material.

- `getMaterialsSupplied()`: Returns a list of materials supplied by the supplier.

- `getMaterialPrice(material: Material)`: Returns the price of a specific material.

- `updateContactDetails(contactDetails: ContactDetails)`: Updates the supplier's contact details.

---

### 5. \*\*Staff\*\*

\*\*Attributes:\*\*

- `staffID`: Unique identifier for the staff member.

- `name`: Name of the staff member.

- `specialization`: Area(s) of expertise (e.g., "Laying Patios", "Excavating").

- `labourRate`: Hourly labor rate of the staff member.

- `tasksAssigned`: List of tasks the staff member is assigned to.

- `projectsWorkedOn`: List of projects the staff member has worked on.

\*\*Operations:\*\*

- `assignToTask(task: Task, hours: float)`: Assigns the staff member to a task for a specified number of hours.

- `getTotalHoursWorked()`: Returns the total number of hours worked across all tasks.

- `getProjects()`: Returns a list of projects the staff member has worked on.

- `getTasks()`: Returns a list of tasks the staff member is assigned to.

- `getLabourCost()`: Returns the total labor cost for the staff member based on hours worked and their labor rate.

---

### 6. \*\*Supervisor\*\* (Inherits from Staff)

\*\*Attributes:\*\*

- `supervisorID`: Unique identifier for the supervisor (inherited from `Staff`).

- `name`: Name of the supervisor (inherited from `Staff`).

- `specialization`: Specialization of the supervisor (inherited from `Staff`).

- `assignedProjects`: List of projects the supervisor is managing.

\*\*Operations:\*\*

- `assignProject(project: Project)`: Assigns the supervisor to a project.

- `getAssignedProjects()`: Returns the list of projects the supervisor is managing.

- `getStaffUnderSupervision()`: Returns a list of staff members working under the supervisor's direction.

---

### 7. \*\*Material\*\*

\*\*Attributes:\*\*

- `materialID`: Unique identifier for the material.

- `name`: Name of the material (e.g., "Hardcore", "Patio Slabs").

- `unitCost`: Cost per unit of the material.

- `stockQuantity`: The quantity of the material in stock.

- `supplier`: Supplier who provides the material.

\*\*Operations:\*\*

- `updateStock(quantity: float)`: Updates the stock quantity of the material.

- `getStockLevel()`: Returns the current stock level of the material.

- `getCost()`: Returns the cost per unit of the material.

- `getSupplier()`: Returns the supplier of the material.

---

### 8. \*\*MaterialTask\*\*

\*\*Attributes:\*\*

- `material`: The material used in the task.

- `quantityUsed`: The quantity of the material used in the task.

- `task`: The task where the material is used.

\*\*Operations:\*\*

- `recordMaterialUsage(material: Material, quantity: float)`: Records the material usage in the task.

- `getMaterialUsage()`: Returns the material and quantity used in the task.

---

### 9. \*\*Specialist\*\*

\*\*Attributes:\*\*

- `specialistID`: Unique identifier for the specialist.

- `name`: Name of the specialist.

- `expertise`: Areas of specialization (e.g., "Water Features", "Landscaping Design").

- `hourlyRate`: Hourly rate charged by the specialist.

- `projects`: List of projects that require specialist tasks.

\*\*Operations:\*\*

- `assignToTask(task: Task)`: Assigns the specialist to a specific task.

- `getExpertise()`: Returns the specialist's area(s) of expertise.

- `getHourlyRate()`: Returns the hourly rate of the specialist.

- `getTasks()`: Returns a list of tasks assigned to the specialist.

---

### 10. \*\*SpecialistTask\*\* (A type of task)

\*\*Attributes:\*\*

- `taskID`: Unique identifier for the specialist task.

- `taskName`: Name/description of the specialist task (e.g., "Building Fountain").

- `scheduledDate`: Date when the specialist task is scheduled to start.

- `duration`: Duration of the specialist task (in hours).

- `specialist`: The specialist assigned to the task.

- `materials`: Materials provided by the specialist (e.g., custom materials).

- `cost`: The total cost for the specialist task (calculated based on hours and rate).

\*\*Operations:\*\*

- `assignSpecialist(specialist: Specialist)`: Assigns the specialist to the task.

- `calculateTaskCost()`: Calculates the total cost for the task, including labor and materials provided by the specialist.

- `getMaterials()`: Returns the materials used by the specialist for the task.

- `getSpecialist()`: Returns the specialist assigned to the task.

---

### Summary

These classes and operations allow for efficient management of projects, tasks, staff, materials, suppliers, and specialists within the building company. The relationships between these entities (e.g., Projects assigned to Customers, Staff allocated to Tasks, Materials supplied by Suppliers, and Specialist Tasks) ensure that the company