**TCC App Workflow Documentation**

**For Microsoft Field Service Integration**

**📌 Table of Contents**

1. **Overview**
2. **User Roles & Responsibilities**
3. **Workflows**
   * **Worker Data Entry & Confirmation**
   * **Equipment Assignment & Usage**
   * **Daily Reporting & Progress Tracking**
   * **Project Cost Allocation & Budget Control**
   * **Microsoft Field Service Integration**
4. **Flowcharts**
5. **Next Steps & Recommendations**

**1- Overview**

The **Total Cost Control (TCC) App** is designed to track **worker hours, equipment usage, and project costs** in construction projects. It provides **real-time reporting, labor tracking, and cost allocation**, ensuring efficiency in field operations.

To enhance functionality, the system is transitioning to **Microsoft Field Service (Dynamics 365)**, integrating **work orders, scheduling, and reporting** into the Microsoft ecosystem.

**2- User Roles & Responsibilities**

| **Role** | **Responsibilities** |
| --- | --- |
| **Worker** | Logs hours worked, selects activity codes, reports equipment usage |
| **Foreman** | Reviews and approves worker hours, assigns equipment |
| **Project Manager** | Monitors labor, material, and equipment usage, approves costs |
| **Admin** | Configures system, manages users, generates reports |

**3️- Workflows**

**3.1 Worker Data Entry & Confirmation**

**🔹 Actors Involved**

* Worker
* Foreman

**🔹 Trigger**

* Worker logs in and selects a project.

**🔹 Step-by-Step Process**

1. Foreman selects the **project and date**.
2. Foreman enters **labor hours** worked.
3. Foreman selects an **activity code** from the dropdown.
4. If using **equipment**, worker selects from the available list.
5. Clicks **“Add”** to submit the entry.
6. Project manager reviews and **confirms the worker’s data**.
7. If approved, the system marks it as **completed**.

**🔹 System Interactions**

* Stores data in the **workers** table.
* Validates activity codes from the project **activity\_codes**.
* Project manager approval updates **confirmation status**.

**🔹 Output & Next Steps**

* Approved hours are **added to project costs**.
* Data syncs with **Microsoft Field Service** for reporting.

**3.2 Equipment Assignment & Usage**

**🔹 Actors Involved**

* Foreman
* Worker

**🔹 Trigger**

* Foreman assigns equipment for daily work.

**🔹 Step-by-Step Process**

1. Foreman selects the **project and worker**.
2. Chooses **equipment** from the dropdown.
3. Assigns a **start time and expected end time**.
4. Clicks **"Confirm Assignment"**.
5. Worker logs actual usage hours.
6. System logs usage and updates **equipment availability**.

**🔹 System Interactions**

* Equipment status updates in **equipment table**.
* Worker-equipment link stored in **equipment\_assignments**.

**🔹 Output & Next Steps**

* Equipment costs **added to the project**.
* Syncs with **Microsoft Field Service Assets**.

**Repeat Same operations for materials and subconstractors**

**3.3 Daily Reporting & Progress Tracking**

**🔹 Actors Involved**

* Worker
* Project Manager

**🔹 Trigger**

* Daily work log submission.

**🔹 Step-by-Step Process**

1. Worker fills out **work completed**.
2. Attaches **pictures or notes** (if needed).
3. Submits the **daily report**.
4. Project Manager **reviews and approves**.
5. Report is **locked for editing** once approved.

**🔹 System Interactions**

* Stores reports in **daily\_report\_statuses**.
* Updates **progress tracking** for the project.

**🔹 Output & Next Steps**

* Generates **Field Service Work Orders**.

**3.4 Project Cost Allocation & Budget Control**

**🔹 Actors Involved**

* Admin
* Project Manager

**🔹 Trigger**

* Monthly budget review.

**🔹 Step-by-Step Process**

1. System aggregates **labor and equipment costs**.
2. Compares actual vs. planned budget.
3. Highlights **over-budget items**.
4. Generates a **financial summary report**.

**🔹 System Interactions**

* Queries **tab\_progress & payment\_items** for cost data.

**🔹 Output & Next Steps**

* Exports data to **Power BI for financial analysis**.

**3.5 Microsoft Field Service Integration**

**🔹 Actors Involved**

* System Admin
* Microsoft Field Service API

**🔹 Trigger**

* Syncing data to **Field Service**.

**🔹 Step-by-Step Process**

1. Worker entries sync with **Microsoft Dataverse**.
2. Field Service creates **work orders** from TCC data.
3. Equipment status updates in **Microsoft Field Service**.
4. Reports are sent to **Power Automate workflows**.
5. Users access reports in **Field Service dashboards**.

**🔹 System Interactions**

* Uses **API calls** to connect **Flask to Field Service**.

**🔹 Output & Next Steps**

* Field Service automates **dispatch and scheduling**.

**4️- Flowcharts**

**Flowchart 1: Worker Data Entry**

pgsql

Copier

+---------------------+

| Worker Logs In |

+---------------------+

|

v

+--------------------------+

| Select Project & Date |

+--------------------------+

|

v

+----------------------------+

| Enter Hours & Activity |

+----------------------------+

|

v

+----------------------------+

| Submit for Approval |

+----------------------------+

|

v

+----------------------------+

| Foreman Approves? |

+----------------------------+

| |

| No | Yes

| v

| +----------------------+

| | Add to Project Cost |

| +----------------------+

| |

| v

| +----------------------+

| | Sync to Field Serv. |

| +----------------------+

| |

v v

+---------------------------+

| Data Saved & Completed |

+---------------------------+

**5️- Next Steps & Recommendations**

1. **Implement Microsoft Dataverse Integration**
   * Map TCC data to **Field Service tables**.
   * Use **Power Automate for workflow automation**.
2. **Develop an API Connection**
   * Flask API should sync **workers, projects, and reports**.
3. **Train Users on Microsoft Field Service**
   * Conduct **workshops** for Project Managers & Workers.
4. **Migrate Historical Data**
   * Load **past project data** into **Dataverse**.