Minutes of Meeting

Date and Time	17 September 2025 09:00 PST	Meeting type	Zoom
Organiser	Mr. Rupesh	**Client**	Citywide

Attendees (Internal)
• Rupesh • Kuldeep • Jaspreet • Vishesh • Rahul • Ajay • Gurpreet • Kapil • Amit • Pankaj
Attendees (Client Side)
• Tom, Teresa, Randy, Shirin, Suzanna

Agenda
• **Discussions on the following:** ○ Coastal Payroll Discussion ○ Discussion on Employee Insurance Integration ○ Additional Callers & Call History ○ Call Priority Levels Discussion ○ Label Changes – End of Shift Acknowledgement ○ ESF Requirement – Use of Force Form ○ Linking Reports & Forms – Workflow Design

The following things are discussed:

1. **Coastal Payroll Discussion**
    a. **Introductions & Objectives:**
        i. **Shirin** introduced Tom and CommandHub to Suzanna, outlining the goal: exploring **API-based integration** between **CommandHub (HRIS/timekeeping system)** and **Coastal Payroll (iSolved platform)**.
        ii. Objective: Establish potential **service bureau-level integration** allowing multiple clients of CommandHub to connect with Coastal Payroll for payroll processing.
        iii. **Tom** clarified that CommandHub handles HR, scheduling, and timekeeping but not payroll. The partnership would allow CommandHub to present a fully integrated HR + payroll solution without clients switching between systems.
    b. **Integration Scope & Data Flow**
        i. **Suzanna(Coastal):**
            1. Confirmed integration will be **API-based**.
            2. Teresa will remain the **system of record** for HR data (employee demographics, time data).
            3. Payroll processing will still occur in Coastal Payroll (iSolved). Clients will authenticate on Coastal's site for payroll execution.
        ii. **Data to be integrated:**
            1. **Employee Demographics:** Names, addresses, personal details, certifications, licenses, etc.
            2. **Time & Attendance Data:** Check-in/out records, overtime adjustments, payroll-ready timesheets.
            3. **Pay Data:** Pay rates, schedules, timesheets mapped to payroll grid.
        iii. **Method:**
            1. Event-based triggers preferred (e.g., pressing the "Export" button in CommandHub pushes finalized time data into iSolved).
            2. Employee demographic data may be sent daily/frequency-based to maintain synchronization.
    c. **Technical Discussion**
        i. **Authentication:**
            1. The current approach uses **JWT (JSON Web Token)** for API authorization.
        ii. **API Endpoints:**
            1. Coastal to provide **full access to iSolved API endpoints** rather than piecemeal access to avoid future re-requests.
            2. This ensures scalability when CommandHub adds new features (e.g., employee KPIs, performance reviews).
        iii. **Field Mapping:**
            1. Suzanna emphasized identifying **security industry-specific employee fields** (certifications, licenses) to map correctly

into iSolved (likely under "miscellaneous fields").
    2. Payroll-critical fields will remain standard (demographics, time, pay data).
  iv. **Time Data Handling:**
    1. Adjustments can be made in CommandHub (check-in/out edits, overtime adjustments, additional hours).
    2. Once validated, data will be exported via API into the Coastal payroll grid.
    3. Alternative methods like drag-and-drop exist but are not API-based (less preferred).

d. **Open Questions & Clarifications**
  i. **Future Enhancements:**
    1. Tom asked how future additions (e.g., KPIs, performance reviews) will be handled post-API.
    2. Suzanna confirmed full endpoint access will allow CommandHub to push new fields without renegotiation.
  ii. **Data Duplication Concern:**
    1. Shirin raised whether all employee records (e.g., performance reviews) should flow into iSolved if already in CommandHub.
    2. Suzanna agreed to focus on payroll-relevant data primarily.
  iii. **Payroll-Specific Fields:**
    1. Confirmed that **standard employee profile data + timesheets + pay rates** are the primary fields required for payroll processing.

e. **Integration Workflow Discussion**
  i. **Suzanna** outlined a two-stream data transfer model:
    1. **Employee Demographics (basic info, compensation, etc.)** → Sent from CommandHub to iSolved on a **scheduled/daily basis**.
    2. **Time & Attendance Data (hours, time entries) →** Sent from CommandHub to iSolved on an **event-based trigger** (e.g., finalizing payroll).
  ii. **Rupesh** raised the need to handle **payroll adjustments**, such as promotions, salary rate changes (increases or decreases), requiring resynchronization of employee data when payroll runs.
  iii. **Suzanna** clarified:
    1. **Employee salary/compensation will be managed in CommandHub**, not iSolved.
    2. iSolved will rely on data pushed from CommandHub for pay rates and demographic changes.
  iv. **Tom** explained the process:
    1. Payroll report is generated and adjusted in CommandHub (add/remove hours, adjust rates, correct times).
    2. Once finalized/locked, it is **sent to Coastal Payroll for processing**.
    3. Discrepancies (e.g., single checks) → handled via Coastal, but CommandHub would typically issue **manual checks**, then reconcile in the next payroll cycle to avoid unnecessary

extra fees.

f. **Payroll Calendars & Restrictions**
    i. **Shirin** confirmed that **Coastal Payroll is the system of record for pay dates and calendars.**
        1. Clients and CommandHub **cannot adjust payroll calendars**.
        2. Dedicated Coastal account managers will handle **special or additional payroll runs**.
    ii. **Tom** acknowledged and agreed that Coastal is the authority for payroll schedules.
        1. CommandHub will lock its records once payroll is finalized and pushed, ensuring both systems stay aligned.

g. **Two-Way Data Exchange**
    i. **Suzanna** recommended a **two-way integration model**:
        1. CommandHub → iSolved: Employee demographics, time, pay data.
        2. iSolved → CommandHub: Payroll outputs (checks, adjustments, deductions, memos).
        3. This is needed so CommandHub can stay in sync with Coastal and feed correct financials to QuickBooks if required.
    ii. **Tom** clarified QuickBooks is used only for **invoicing**, not full GL/COA management, but agreed two-way API is necessary.

h. **API Setup & Next Steps**
    i. **Suzanna** to send a **follow-up email** with:
        1. API setup template.
        2. Requirements for designated API user account/email for integration.
    ii. **Rupesh** to provide the **email address for API configuration**.
    iii. **Tom & Teresa** to create a dedicated group titled **"Coastal Payroll API"** (instead of iSolved, to avoid confusion for clients).
        1. Members: Shirin, Suzanna, Teresa, Rupesh, Jaspreet, Koldeev.
        2. Shirin requested to be **CC'd** to monitor communications.

i. **System Capabilities & Client Concerns**
    i. **Randy** asked about iSolved's ability to handle **law enforcement/security-specific requirements**.
    ii. **Shirin** clarified:
        1. iSolved is one of the **largest HCM platforms** in the U.S., supporting millions of employees.
        2. Coastal Payroll is the **largest independently owned payroll provider on the West Coast**.
        3. The system is flexible enough to handle **government and security industry payrolls** (e.g., complex railroad payrolls).
        4. Coastal Payroll itself will **only handle payroll, payroll taxes, W2s, direct deposits, checks** for CommandHub clients.
        5. **HRIS, training, scheduling, and timekeeping remain in CommandHub.**

j. **Expansion into Railroads and Law Enforcement**
    i. **Randy** highlighted CommandHub's experience with training railroad

police and emphasized future transition plans to law enforcement, federal agencies, and specialty teams.
    ii. **Shirin** expressed enthusiasm and willingness to support this transition.

k. **QuickBooks Online API Discussion**
    i. **Shirin** informed the group that an API with QuickBooks Online exists and could be activated if needed.
    ii. **Tom** noted this would be "massive" but acknowledged associated costs and the need to avoid overlap with CommandHub's own solutions.
    iii. **Pricing Reference:** Shirin mentioned regular pricing of **$25 per payroll**.
    iv. **Decision:** For now, the team will proceed with payroll directly, and revisit QuickBooks integration later.

l. **Beta Testing Approach**
    i. **Shirin** suggested testing payroll integration internally (e.g., Commander account) before involving external clients.
    ii. **Tom** agreed and confirmed that **CommandHub and Citywide Planning accounts will be migrated to Coastal Payroll** as a starting point.
    iii. **Action:** Teresa to coordinate with Hassan, KZ, Sophie, and Shirin to roll over accounts.

m. **Competitive Positioning**
    i. The current payroll provider is **ADP**.
    ii. Tom emphasized moving away from ADP ("dump the enemy, go with the ally").
    iii. **Randy** asked about Coastal Payroll's differentiators vs ADP.
    iv. **Shirin** highlighted:
        1. Direct service model (no phone trees, queues, or chats).
        2. Dedicated implementation specialists and account managers.
        3. Strong client referral base (90% of business).
        4. High-touch communication and personalized service.

n. **Account Manager Allocation**
    i. **Randy** asked if CommandHub and its clients would each get account managers.
    ii. **Shirin** clarified that as the client base grows, a **team-based support structure** will be provided, not just one account manager.

o. **Integration & Data Syncing**
    i. **Kuldeep (Organizer)** raised concerns about syncing **Commander and Coastal Payroll** systems.
    ii. **Discussion Points:**
        1. Coastal Payroll has APIs that can sync payroll status data back to CommandHub.
        2. Possible use of **webhooks** to update payroll run status automatically.
        3. **Suzanne** (not present) will provide detailed API/data requirements over email.
    iii. **Tom's View:** Payroll data should also be accessible within CommandHub to save time for clients (5 seconds vs 30 seconds with

logins).

iv. **Shirin's Note:** Need to evaluate if payroll data truly needs to be stored in both systems or just linked.

v. **Decision:** Suzanne to lead the technical mapping and requirements via email chain.

p. Data Responsibility & Separation

i. Shirin emphasized being mindful of **what data resides in CommandHub vs Coastal Payroll/iSolved**, ensuring only necessary data is shared.

ii. Clear communication to clients on where payroll, reports, and performance data will be accessible.

2. Discussion on Employee Insurance Integration

a. **Kuldeep** raised the question of whether CommandHub should manage employee insurance (health insurance, etc.) within the system.

b. **Context:** Some countries (e.g., India) legally require employers to provide health insurance to employees.

c. Tom's Response:

i. Insurance management itself does not need to be built into CommandHub.

ii. Employers will purchase insurance separately, outside the system.

iii. **CommandHub's role**: Manage the **deductions** related to insurance through payroll, synced with Coastal Payroll via API.

iv. Required fields for deductions:

1. Deduction name
2. Reason/description
3. Amount
4. Frequency
5. Total (or continuous until stopped)

v. These will need to be aligned with Coastal Payroll through API mapping.

vi. **Decision:**

1. CommandHub will **not manage insurance platforms directly**, only deductions and payroll alignment.

d. WC (Workers' Compensation) vs Health Insurance

i. **Jaspreet** raised a clarification about WC (Workers' Compensation) codes, asking if they also count as insurance.

ii. **Randy** clarified: WC = WorkComp, which is different from health insurance.

iii. **Tom's Note:**

1. WorkComp is managed via the payroll company.
2. WC codes appear on payroll reports and are used during WorkComp audits.
3. Payroll reports will show employees' class codes for auditing purposes.

iv. Decision:

1. WC codes will be treated as payroll-linked insurance deductions/line items, not managed separately within CommandHub.

e. Broader Benefits & Perks Handling

        i. **Rupesh** added that in the U.S., companies often provide additional perks/benefits (e.g., allowances, insurances, reimbursements) as part of payroll.
        ii. These could be added as payroll deductions/fields in CommandHub.
        iii. **Decision:**
            1. Benefits/perks will be categorized under **deductions/line items** synced with Coastal Payroll.
            2. Not a separate management module, but handled through payroll structure.
    f. **Team Alignment & Next Steps**
        i. Insurance and WC will **not** be managed as standalone systems inside CommandHub.
        ii. Deduction handling (health, WC, perks, other benefits) will be added to **employee profile payroll fields**.
        iii. Ensure **API sync with Coastal Payroll** for deductions and class codes.
3. **Additional Callers & Call History**
    a. **Demo Overview:**
        i. A new implementation to allow adding **additional callers** to call-in type calls was demonstrated.
        ii. Each caller's information is captured via a form.
    b. **Requirements & Feedback:**
        i. **Call History:** Second caller's call history should also be displayed. (Currently not working).
        ii. **Ease of Use:**
            1. Current design requires scrolling to the bottom to add notes, which slows down workflow.
            2. Suggestion: Implement a **radio button/dropdown/collapsible section** (similar to scheduling sidebar drawer) to manage caller details without cluttering the modal.
            3. Option for **collapsible/expandable menus** for "Available Units" and "Additional Callers."
        iii. **Call Details Integration:**
            1. Call details should be visible in the same view.
            2. Example: First caller reports "loud music in Unit 212," second caller reports "fighting in Unit 212" → both details should be visible in one record.
            3. Each additional caller should have a **separate notes/description box**.
        iv. **Address Autofill:**
            1. Option to prefill site address for new callers.
            2. Add a radio button "Same as address above," with the ability to only modify apartment/unit number.
        v. **Editing Caller Info:**
            1. Form should support editing/updating caller details.
            2. If "Edit" is selected by mistake, user should be able to cancel and reset form.
            3. Form should support editing/updating caller details.

        4. If “Edit” is selected by mistake, user should be able to cancel and reset form.
    vi. **UI/UX Recommendation:**
        1. Current modal view is overloaded (Notes, Caller Details, Assigned Units).
        2. **Proposal:** Use a **side drawer** layout to optimize space, reduce scrolling, and improve dispatcher usability.
  c. **Night Mode & Priority Level Colors**
    i. **Issue Identified:**
        1. In **Night Mode**, priority level colors (High/Hot/Medium) are unreadable.
        2. Yellow and red contrasts are particularly problematic.
        3. Currently, only the orange **color is displayed** correctly.
    ii. **Client Feedback & Suggestions:**
        1. Colors should be **distinct and readable** in dark mode.
        2. Instead of font color only, **the entire button background** should reflect priority color (e.g., bright red for “Hot”).
        3. The font should remain **white or black** for contrast.
        4. Buttons should be **clickable and raised** on selection to visually indicate status.
    iii. **Status:**
        1. Priority color redesign is **still pending**.
        2. Client expects an updated demonstration once design changes are implemented.

4. **Call Priority Levels Discussion**
  a. **Current Structure:**
    i. Existing system uses: **Hot, High, Medium, Low, None**.
    ii. **“None”** currently acts as a default when no priority is assigned in the database.
  b. **Concerns Raised:**
    i. **Randy:**
        1. “None” is problematic; dispatchers should be forced to assign a priority.
        2. Prefers clear, enforceable levels such as Routine, Medium, High, Hot.
        3. Referenced FBI standards:
            a. **Code 3** → Lights & Sirens (highest urgency).
            b. **Code 2** → Lights only.
            c. **Code 1** → Routine.
    ii. **Tom:**
        1. Wanted to simplify into **Code 1, 2, 3** mapping to different urgency levels.
        2. Later proposed to keep it straightforward as **Hot (highest), High, Medium, Low (Routine)** and remove “None.”
    iii. **Teresa:**
        1. Some agencies use different interpretations of codes (e.g., Code 3 being high in some, low in others).
        2. Suggested a simpler “Low, Medium, High, Hot” to avoid

confusion.

c. **Resolution on Priority Naming & Behavior**
   i. **Final Agreement:**
      1. Remove **"None"** as a priority option.
      2. Adopt four levels:
         a. **Hot** – Highest urgency.
         b. **High** – Second level.
         c. **Medium** – Standard priority.
         d. **Low (Routine)** – Lowest priority.
      3. Default fallback: If no priority is set, it will be automatically assigned as **Routine (Low)**.
      4. All calls will be **highlighted automatically** (no unhighlighted calls).
   ii. **Activity Code Adjustments:**
      1. Activity code priorities (previously none, low, high) will be re-labeled as:
         a. **No Highlight**
         b. **Highlight Blue**
         c. **Highlight Yellow**
      2. Activity codes will only affect **visual highlighting**, not override call priorities.
      3. Call-level priority selection (Hot/High/Medium/Low) will override activity code priority when assigned.
   iii. **Customization Options**
      1. **Company-Specific Settings:**
         a. Organizations can configure their own **priority labels and colors** under company settings.
         b. Example: A company may want **Code 1, Code 2, Code 3** instead of Low/Medium/High.
         c. Colors can also be customized per company preference (e.g., Code 1 = Red, Code 2 = Orange, Code 3 = Yellow).
      2. **Flexibility:**
         a. Allows each company to define terminology that aligns with their internal policies (security vs. law enforcement).
         b. Reduces confusion when agencies use different definitions for the same code.

5. **Label Changes – End of Shift Acknowledgement**
   a. **Issue Identified:**
      i. In **Company Settings**, the current **Acknowledgement** button is mislabeled.
      ii. Regardless of settings, the system always forces users to acknowledge the first end-of-shift message.
   b. **Clarification:**
      i. The current **Acknowledgement** button should be relabeled as **Alternative Message**.
      ii. This button allows users to provide an alternative message/reason instead of simply acknowledging.

c. **Decision:**
    i. Change all instances of the **Acknowledgement** button to **Alternative Message**.
    ii. Keep the first shift-clear acknowledgement as is (mandatory).
d. **Action Item:** The Dev Team Update label changes in the system.

6. **ESF Requirement – Use of Force Form**
    a. **Request:**
        i. ESF Instance requested a **Use of Force Form** to be added only to their instance.
    b. **Clarification:**
        i. The form will be a **separate form**, not embedded inside the Incident Report.
        ii. It will appear alongside other forms (e.g., FI, Trespass).
        iii. Each form should be linkable to an **Incident Report/Call number**.
    c. **Decision:**
        i. Create a **new Use of Force Form** under the forms module.
        ii. Ensure it can be linked to an Incident Report ID.
    d. **Action Item:** The Dev team Implement the form addition and linking.
7. **Linking Reports & Forms – Workflow Design**
    a. **Problem Raised:**
        i. Currently unclear how forms should link to incident reports.
        ii. Options considered: open in side drawer vs. new tab.
    b. **Tom's Recommendation:**
        i. **Incident Report should be the parent record.**
        ii. Workflow:
            1. User completes and saves an Incident Report.
            2. From the **Forms module**, user selects a form (Parking Citation, Trespass, Field Interview, Use of Force).
            3. User selects an **Incident Report ID** from a dropdown.
            4. System auto-populates related data (individuals, vehicles, etc.).
            5. User selects relevant entity (victim, witness, suspect, vehicle) → system copies data into the new form.
            6. User adds reasons/details and saves.
    c. **Reference:**
        i. Randy confirmed this workflow is similar to Axon's DraftOne app (well-received).
    d. **Decision:**
        i. Adopt Tom's proposed workflow.
        ii. No direct embedding of forms inside the Incident Report.
        iii. All forms should be created separately but linkable to the Incident Report.
    e. **Action Item:** [Ditstek] Implement incident-form linking as per workflow.