Time Keeping API

Payroll Purpose:

The user will start his day by clocking in. This is for payroll purposes only. At lunch, he will stop the clock and then begin the clock again on return. Due to lunch breaks, most users will have multiple payroll time periods per day.

Work Order Purpose:

Users will record their time on individual work orders (like task logging). This is to break down how they spent their day. There are multiple types of time (~/api/timetype) that they can log, such as travel, work, miscellaneous.

Field Types

* Workspace ID: Int
* User ID: int
* Start timestamp: epoch time
* Stop timestamp: epoch time
* Duration: seconds
* Start GPS: https://www.rfc-editor.org/rfc/rfc7946
* Stop GPS: https://www.rfc-editor.org/rfc/rfc7946
* Approved: boolean
* Note: string

Documentation Requirements:

Document in the form of a postman collection

Testing Requirements:

API tests

# Endpoints

POST PATCH GET ~/api/payroll\_time\_period

* Response
  + Workspace id
  + User id
  + Start timestamp
  + Stop timestamp
  + Start gps
  + Stop gps
  + Approved y/n
  + Note
* Query Params
  + Workspace id
  + User id
  + stop\_time=null
  + Start time, end time range
* Notes
  + This will be POSTed with start time and gps, without stop time and stop GPS until the user ends the clock. Response needs to include PK of payroll\_time\_period.

POST PATCH GET ~/api/timetype

* Response
  + Name
  + Workspace id
  + Type [work,travel]
* Query Params
  + Workspace id
* Notes
  + This is to set the types of time that can be logged for a work time period.

POST PATCH GET ~/api/work\_time\_period

* Query Params
  + Workspace ID
  + User ID
  + WO ID
  + Start time, end time range
* Response
  + Workspace id
  + User id
  + WO id
  + Time type id
  + Start timestamp
  + Stop timestamp
  + Mileage (can be blank)
  + GPS point start
  + Gps point stop
  + Approved y/n
  + Note
* Notes
  + This endpoint can return multiple nested time periods

POST PATCH GET ~/api/crew

* Response
  + Crew name
  + active/inactive
  + Workspace id
  + Foreman user id
  + Crew users ids
* Query Params
  + Workspace id
  + active/inactive
  + Foreman user id
* Notes
  + This is for the purpose of logging time for multiple users at once (crew of 2-4 people in a truck). The front end will GET /api/crew to know the users to log time for.

POST ~/api/gps

* Response
  + Workspace id
  + User id
  + GPS Point
  + Timestamp
* Query Params
* Notes
  + This is to report the users coordinates while they are logging time

GET ~/api/gps\_path

* Query Params
  + Workspace ID
  + User id
  + Start timestamp
  + Stop timestamp
  + WO ID
  + Payroll\_time\_period ID
* Response
  + <https://www.rfc-editor.org/rfc/rfc7946#section-1.5>
* Notes
  + The purpose of this is to get a GPS path the user took based on data submitted with ~/api/gps during time logging.
  + Ideally , passing WO ID or payroll\_time\_period ID’s would filter and return results. This would have to be accomplished by getting the time and users on the WO, then filtering coordinate records by that.

GET ~/api/current\_gps

* Query Params
  + Workspace ID
  + User id
* Response
  + User ID
  + GPS Point
  + Timestamp
* Notes
  + The purpose of this is to return users and their location and timestamp

# Reporting Endpoints

GET ~/api/time\_sheet **(WIP)**

* Query Params
  + Workspace id
  + User id
  + Start time, end time
  + Approval status
* Response
  + User ID
    - Payroll Time Period
      * {start time, end time, note, duration}
      * Work\_time period
        + Time type
        + Start time
        + End time
        + duration
        + Note
* Notes
  + Nested JSON of the user > payroll time period > work\_time period.

GET ~/api/payroll\_report **(WIP)**

* Query Param
  + Start time
  + End time
  + workspace
  + User ids
  + Approval status
* Response
  + User
  + Date
  + Payroll time period
    - Time Types
      * duration
      * distance
* Notes
  + See Duration Calculation Notes on next page

| User | Date | Payroll Duration | Time Type 1 Duration | Time Type 2 Duration | Time Type 3 Duration | Time Type 3 Distance |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| George Smith | 10/14/2022 | 6.5 hr | 2.3hr | .25 hr | 3.1 hr | 75 miles |  |
| Randy Smith | 10/14/2022 | 8.5 hr | 1.3hr | 1.25 hr | 3 6 hr | 85 miles |  |

GET ~/api/work\_report **(WIP)**

* Query Param
  + Start time
  + End time
  + workspace id
  + User ids
  + Approval status
* Response
  + User
  + Date
  + Payroll time period
    - WO
      * Time Types
        + duration
        + distance
        + note
* Notes
  + See Duration Calculation Notes on next page

| User | Date | WO | Payroll Duration |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| George Smith | 10/14/20 | 124 |  |  |  |  |  |  |
| George Smith | 10/14/20 | 124 |  |  |  |  |  |  |

Get ~/api/time\_utilization

* Query Param
  + Start time
  + End time
  + workspace id
* Response
  + user Id
    - Time types
    - Respective percentage
* Notes
  + This is to calculate the efficiency of techs. The endpoint will return all time times and the percentage of that time relative to payroll time. If they log 8 hours for payroll time and work 4 hours and travel 1 hour, the response would look like:
    - Travel: .125
    - Work: .5
    - Unlogged: .375

GET ~/apti/active\_time

* Query Param
  + workspace id
  + User id
* Response
  + User ID
  + GPS Point
  + Time Log Time [payroll\_timel vs work\_time]
  + Time Type (if work\_time\_period)
  + Work Order ID
* Notes
  + Return a list of all users actively working. Work\_time is more detailed than payroll\_time so if a user has an active timer for work\_time, return that info. If no active work\_time, return payroll\_time (if active).

# Duration Calculation Notes

There is an edge case when calculating duration of time relative to the queried start and end time. We’ll use the following payroll time reporting period: Sunday 12:00am to Saturday 11:59pm. If a user begins work on Saturday at 10pm and works until 6am on Sunday, the duration will need to be concatenated and calculated for the portion of the work time period within the reporting period.

For work\_time\_periods, the start or end time would be amended to not exceed the reporting time.