Altametrics

AnyConnector API Reference Document

Version 1.3

Prepared by: Altametrics

June 4, 2019

Altametrics, LLC
3191 Red Hill Avenue
Costa Mesa, CA 92626 USA
Tel: (800) 676-1281

Last edited: 20 June 2019

Contents

1. Introduction

The purpose of this document is to provide detailed description of data fields required to fetch and post the data from Hubworks, also contains the type of file format that Hubworks will support to display the data. In Any connector, we are trying to fetch the sales, employees, menu-mix, invoices, and ordering etc. data from POS and try to post the Payroll data.

2. Data Transmission Files

There are basically two types of files:

1. **Configuration File:** It is the configuration file where different parameters are defined with their unique ids and name. This file will be transmitted when there is any modification or addition of any new parameter. This table acts as a look-up table for transaction file
2. **Transaction file:** This file will transmitted on daily basis that contains the sales data, employees data, menu-mix data etc.
3. File naming convention of both the files should be as per the format given below:

<filename>_storeID_<MMDDYYYY>

2.1 Configuration File

2.1.1 Jobs

This look-up contains the details of all the positions that are configured at the POS system. Each position will be uniquely identified using its ID and notation of the same will be done based on the supplied name of the position.

Sample JSON


```
{
 "jobs": [
  {
   "id": "701",
   "name": "Chef",
   "isSalaried": true,
   "wage": 0
  },
  {
   "id": "2",
   "name": "Cashier",
   "isSalaried": true,
   "wage": 0
  },
  {
   "id": "3",
   "name": "Manager",
   "isSalaried": true,
   "wage": 0
  }
]
}
```

Parameter details of Jobs:

Parameter	Description	Data Type	Required	Example
Id	ID of the job.	string	Yes	1001
name	Name of the job.	string	No	Chef
isSalaried	If the position is salaried (non-hourly pay)	Boolean	No	No
wage	Hourly pay rate of job.	decimal(2)	No	8

2.1.2 Categories

This look-up contains an array of unique categories and each category should have the fields mentioned in the parameter details table below.

Sample JSON:


```
{
 "categories": [
  {
   "id": "500",
   "name": "Burgers"
  },
  {
   "id": "501",
   "name": "Breakfast food"
  },
  {
   "id": "502",
   "name": "Lunch food"
  },
  {
   "id": "503",
   "name": "Dinner food"
  },
  {
   "id": "504",
   "name": "Breakfast Beverage"
  },
```

```
{
    "id": "506",
    "name": "Dinner Beverage"
  },
  {
    "id": "511",
    "name": "Beer"
  },
  {
    "id": "512",
    "name": "Wine"
  },
  {
    "id": "513",
    "name": "Starters"
  },
  {
    "id": "515",
    "name": "Dessert"
  }
 ]
}
```

Parameter Details of Categories:

Parameter	Description	Type	Required	Example
Id	ID of the item category	string	Yes	500
Name	Name of the item category	string	No	Burgers

2.1.3 Items

This lookup contains the details of unique menu items that are identified using its id and name.

Sample JSON:


```
{
 "items": [
  {
   "id": "5001",
   "name": "Bacon Avocado Cheeseburger",
   "categoryID": "500",
   "price": 11.59
  },  {
   "id": "5002",
   "name": "Matcha Shot - Soy - Single",
   "categoryID": "500",
   "price": 4.50
  },
```

```
{
    "id": "5003",
    "name": "Matcha Shot - Soy - Single",
    "categoryID": "500",
    "price": 4.50
  },{
    "id": "5004",
    "name": "Tropical Kick Start 28oz",
    "categoryID": "500",
    "price": 3.50
  },{
    "id": "5005",
    "name": "Kale Orange Power 28oz",
    "categoryID": "500",
    "price": 3.50
  },{
    "id": "5006",
    "name": "Cucumber Orange Cooler 16oz",
    "categoryID": "500",
    "price": 6.22
  }
 ]
}
```

Parameters Details of items:

Parameter	Description	Type	Required	Example
id	The ID of an item	string	Yes	5001
name	The name of the item.	string	No	Veg Burger
categoryID	The name of the category	string	No	500
price	Price of an item	Decimal(2)	No	2.5

2.1.4 Dayparts

It is a concept of dividing the working hours of a restaurant in different parts like Lunch, Dinner, and Happy Hour etc. The lookup contains the details of day parts. They are uniquely identified using its id and name.

Sample JSON:

Parameter	Description	Type	Required	Example
id	The ID of an item	string	Yes	1
name	The name of the item.	string	No	Happy Hour
startTime	The start time of the Daypart	time	Yes	1000
endTime	The end time of the Daypart	time	Yes	1250

```
{
"dayParts": [
    {
     "id": "1",
     "name": "Breakfast",
     "startTime": 420,
     "endTime": 600
    },
    {
     "id": "2",
     "name": "Lunch",
     "startTime": 600,
     "endTime": 840
    },
    {
     "id": "3",
     "name": "Evening",
     "startTime": 840,
     "endTime": 1140
    },
    {
     "id": "4",
     "name": "Dinner",
     "startTime": 1140,
     "endTime": 1380
    }
]
}
```

2.1.5 Revcenter

This lookup contains the details of sales by type from where the order is being placed. They are uniquely identified using its id and name.

Sample JSON:


```
{
  "revCenters": [
    {
      "id": "1",
      "name": "Dining Room"
    },
    {
      "id": "2",
      "name": "Dine In Sales"
    },
    {
      "id": "3",
      "name": "Dine in Checks"
    },
```

```
{
    "id": "4",
    "name": "To Go Sales"
   },
   {
    "id": "5",
    "name": "To Go Checks"
   },
   {
    "id": "6",
    "name": "Drive Through Sales"
   },
   {
    "id": "7",
    "name": "Drive Thru Checks"
   },
   {
    "id": "8",
    "name": "Catering Sales"
   },
   {
    "id": "9",
    "name": "Catering Checks"
   }
  ]
}
```

Parameters Details of Revcenter:

Parameter	Description	Type	Required	Example
id	The ID of the revCenter	string	Yes	1
name	The name of the revCenter	string	No	Dining Room

2.1.6 Voids

This lookup contains the details of the voids due to which the items in a check are being canceled. These voids are defined with a unique id and their name or reason.

Sample JSON:


```
{
 "voids": [
  {
   "id": "V011",
   "name": "Unavailable"
  },
  {
   "id": "V012",
   "name": "Retail Voids Amt"
  },
  {
   "id": "V013",
   "name": "Catering Voids Amt"
  },
  {
   "id": "V014",
   "name": "Overringst"
  },
  {
   "id": "V015",
   "name": "Refunds"
  }
 ]
}
```

Parameters Details of Voids:

Parameter	Description	Type	Required	Example
id	The ID of the void	string	Yes	V012
name	The reason for void	string	No	Retail Voids Amt

2.1.7 Discounts

This lookup contains the details of the discount that are being used in the check when an order is placed. It is identified by their unique id and name.

Sample JSON:


```
{
 "discounts": [
  {
   "id": "C011",
   "name": "OTH 100% Off"
  },
  {
   "id": "C012",
   "name": "Landlord/Prop Mgr Disco 20%"
  },
  {
   "id": "C013",
   "name": "Mall/Build Employee Disc 10%"
  },
  {
   "id": "C014",
   "name": "Police Officer Discount 10%"
  },
  {
   "id": "C015",
   "name": "Senior Discount 10%"
  },
```


Parameters Details of Discount:

Parameter	Description	Type	Required	Example
id	The ID of the discount	string	Yes	C015
name	Discount name	string	No	Emp 10%

2.1.8 Comps

This lookup contains the details of all the Comps that are configured at the POS system. Each comps will be uniquely identified by its ID and name.


```
"comps": [
  {
   "id": "1",
   "name": "#1 Comp"
  },
  {
   "id": "2",
   "name": "Catering Comp"
  },
  {
   "id": "3",
   "name": "Employee Comp"
  },
```

```
 {
  "id": "4",
  "name": Manager Comp"
 },
 {
  "id": "5",
  "name": "Supervisor Comp"
 },
 {
  "id": "6",
  "name": "Free BWS"
 }
]
```


Parameters Details of Comp:

Parameter	Description	Type	Required	Example
id	The ID of the comp.	string	Yes	3
name	Comp name	string	No	Employee Comp

2.1.9 Payment types

This lookup contains the details of all the payment type that are configured at the POS system. Each payment type will be uniquely identified using its ID and name.

Sample JSON:


```
{
 "paymentTypes": [
  {
   "id": "1",
   "name": "Cash",
   "paymentGroup": "CASH"
  },
  {
   "id": "2",
   "name": "VISA",
   "paymentGroup": "VISA"
  },
  {
   "id": "3",
   "name": "MASTER",
   "paymentGroup": "MASTER"
  },
  {
   "id": "4",
   "name": "DISCOVER",
   "paymentGroup": "DISCOVER"
  },
```

```
{
    "id": "5",
    "name": "AMERICAN EXPRESS",
    "paymentGroup": "AMERICAN EXPRESS"
  },
  {
    "id": "6",
    "name": "GIFT CARD",
    "paymentGroup": "GIFT CARD"
  }
 ]
}
```


Parameters Details of payment types:

Parameter	Description	Type	Required	Example
id	The ID of the payment type	string	Yes	3
name	Name of payment type	string	No	MASTER
payment group	Name of payment group	string	No	MASTER

2.1.10 Paid In/Out

This Lookup contains the detail of all the Paid In/Out payments that are configured in the POS system. Each Paid In/Out type will be uniquely identified using its Id and Name.

Sample JSON


```
{
 "paidInOut": [
  {
   "id": "1",
   "name": "COS: Beverage"
  },
  {
   "id": "2",
   "name": "COS: Meat"
  },
  {
   "id": "3",
   "name": "COS: Poultry"
  },
  {
   "id": "4",
   "name": "COS: Dairy"
  },
  {
   "id": "5",
   "name": "COS: Pasta Oil and Other"
  },
```

```
{
    "id": "7",
    "name": "Vehicle Toll and Maintenance"
  },
  {
    "id": "8",
    "name": "Operating Supplies"
  },
  {
    "id": "9",
    "name": "Cleaning Supplies"
  },
  {
    "id": "10",
    "name": "Courier Cost"
  }
 ]
}
```

Parameters Details of paid In/Out:

Parameter	Description	Type	Required	Example
id	The Id of the payment (Paid In/Out) reason	string	Yes	2
name	The name Of the payment (Paid In/Out) reason	string	No	COS: Meat

2.1.11 Employees

This lookup contains the details of all the employee that are uniquely identified by their id and first name and jobs are assigned to employee.

Sample JSON:


```
{
 "employees":
 [
  {
    "id": "1001",
    "firstName": "Drew",
    "lastName": "Barrymore",
    "empNum": 99998889,
    "roleID":"1"
  "empJobs":
                {
  "id": "701",
  "wage": 3.45,
  "isPrimary": false
                },
  },
```

```
{
    "id": "1002",
    "firstName": "Tim",
    "lastName": "Cook",
    "empNum": 9090909090,
    "roleID":"1"
  "empJobs":
                {
  "id": "701",
  "wage": 3.45,
  "isPrimary": false
                },
  }
 ]
}
```


Parameters Details of Employees:

Parameter	Description	Data Type	Required	Example
id	The ID of an employee	string	Yes	1003
First name	First name of an employee	string	Yes	John
Last name	Last name of an employee	string	No	Watson

empNum	Employee Number of an employee	integer	No	99998889
roleId	Role Id of an employee	string	No	3
empJobs				
Id	ID of the job.	string	Yes	703
wage	Hourly pay rate of job.	decimal(2)	No	7.50
isprimary	Default job of the Employee	boolean	No	false

2.2 Transaction file:

Transaction file is the file containing daily transactions data of a single store.Transaction data refers to four types of data checks, paidins & paidouts, deposits, and shifts (timekeeping data). One or more types of transaction data can be sent in the same request.

2.2.1 Check Data

A "check" refers to a single customer bill of sale. Within every "check" there are several data types. Check Info, Check Discounts, Check Items (includes voids, discounts, and modifiers), and Check Payments. Following is the Sample JSON for General Check Info:

Sample JSON:


```
{
  "checks": [
    {
      "checkNum": "10001",
      "orderNum": "1007",
      "employeeID": "104",
      "openedAt": "2018-04-10T09:09:00",
      "closedAt": "2018-04-10T10:42:00",
      "revCenterID": "1",
      "dayPartID": "1",
      "lastModifiedAt": "2018-04-10T10:42:00",
      "guestCount": 1,
      "tableNum": "2",
      "inclusiveTax": 0,
      "exclusiveTax": 0.25,
      "total": 3.14,
      "netAmount":3.14,
       "grossAmount":3.39,
       "adjustedGrossAmount":3.39,
      "busiDate": "2018-04-10",
      "busiTime": "09:09:00",
```

Parameters Details of Check Info:

Parameter	Description	Data Type	Required	Example
checkNum	Unique ID for a check for a given POS in a location.	string	Yes	10002
orderNum	Order number which is sometimes associated with a check in some POS systems	string	Yes	10002
busiDate	Date used for reporting the sales	date	Yes	2018-04-10
busiTime	Time used for reporting the sales. This is a local time (not UTC) and is used so that we can easily report on sales from 3-4pm at multiple locations across different time zones.	time	Yes	10:42:00
openedAt	Date & time the check was opened	datetime	No	2018-04-10T10:42:00
closedAt	Date & time the check was closed, if this is NULL, we will report the check as being “open”	datetime	No	2018-04-10T10:46:00

lastModifiedAt	Last time the check was updated	datetime	No	2018-04-10T10:46:00
employeeId	ID for Employee assigned to the check. From the configuration data.	string	No	3
revCenterId	The revenue center ID. From the configuration data.	string	No	3
dayPartID	The Daypart ID. From the configuration data.	string	No	1
guestCount	Total number of guests. If QSR, just default to 1.	integer	No	1
tableNum	Table number	string	No	2
inclusiveTax	Total amount of inclusive tax	decimal(2)	No	0.34
exclusiveTax	Total amount of exclusive tax	decimal(2)	No	0.15
autoGratuity	Total amount of auto gratuity applied to the check	decimal(2)	No	3.39
netAmount	Total net amount	decimal(2)	No	1.84
grossAmount	Total gross amount.	decimal(2)	No	5.72

adjustedGrossAmount	Total adjusted gross amount	decimal(2)	No	5.45
total	The check total	decimal(2)	No	5.45

2.2.2 Check Items

A check can have multiple items applied to it.

Sample JSON:

```
"items": [
{
        "itemID": "100",
        "quantity": 1.0,
        "price": 3.69,
        "inclusiveTax": 0.0,
        "amount": 1.84,
        }, {
        "itemID": "101",
        "quantity": 1.0,
        "price": 2.0,
        "inclusiveTax": 0.0,
        "amount": 2.0
        }, {
        "itemID": "102",
        "quantity":1.0,
        "price": 0.0,
        "inclusiveTax": 0.0,
        "amount": 2.89
        },{
        "itemID": "103",
        "quantity": 1.0,
        "price": 3.69,
        "inclusiveTax": 0.0,
        "amount": 1.84
        },
```

```
{
        "itemID": "104",
        "quantity": 1.0,
        "price": 2.58,
        "inclusiveTax": 0.0,
        "amount": 2.0
        }, {
        "itemID": "105",
        "quantity":1.0,
        "price": 3.25,
        "inclusiveTax": 0.0,
        "amount": 2.89
        },{
        "itemID": "106",
        "quantity": 1.0,
        "price": 3.69,
        "inclusiveTax": 0.0,
        "amount": 1.84
        }, {
        "itemID": "107",
        "quantity": 1.0,
        "price": 2.99,
        "inclusiveTax": 0.0,
        "amount": 2.0
        }, {
        "itemID": "108",
        "quantity":1.0,
        "price": 0.0,
        "inclusiveTax": 0.0,
        "amount": 2.89
        }
]
```

Parameters Details of Check Items:

Parameter	Description	Type	Required	Example
itemid	Sales item ID. From the configuration data.	string	Yes	107
quantity	Total quantity sold	decimal(2)	No	2
amount	Total sales amount for this item (this amount SHOULD INCLUDE discount/comp but SHOULD NOT INCLUDE taxes)	decimal(2)	No	5.98
inclusive_tax	Inclusive tax amount	decimal(2)	No	0.15
exclusive_tax	Exclusive tax amount	decimal(2)	No	0.35
price	The price of item	decimal(2)	No	6.48

2.2.3 Check Payments

A check can have multiple payments applied to it. Here are the fields for a payment:

Sample JSON:


```
"payment": [
{
   "total": 1.85,
   "paymentTypeID": "2",
   "received": 0,
   "change": 0,
   "tip": 0,
   "appliedAt": "2007-03-29T18:10:00"
 },{
   "total": 2.85,
   "paymentTypeID": "2",
   "received": 0,
   "change": 0,
   "tip": 0,
   "appliedAt": "2007-03-29T18:10:00"
 },{
   "total": 3.59,
   "paymentTypeID": "3",
   "received": 0,
   "change": 0,
   "tip": 0,
```

```
    "appliedAt": "2007-03-29T18:10:00"
},
{
    "total": 7.49,
    "paymentTypeID": "4",
    "received": 10.00,
    "change": 2.51,
    "tip": 0,
    "appliedAt": "2007-03-29T18:10:00"
 },{
    "total": 5.00,
    "paymentTypeID": "5",
    "received": 10.00,
    "change": 5.00,
    "tip": 0,
    "appliedAt": "2007-03-29T18:10:00"
}
```


Parameters Details of Check Payments:

Parameter	Description	Type	Required	Example
paymentTypeId	ID for the payment type	string	Yes	5
total	Total amount of the payment applied towards the check total	decimal(2)	No	5.00

received	Total amount received from the customer	decimal(2)	No	10.00
change	Total amount of change given to the customer	decimal(2)	No	5.00
tip	Credit card tip amount	decimal(2)	No	2.00
appliedAt	Date & time the payment was appli	datetime	No	2018-04-10T10:42:00

2.2.4 Check Discount

A check can have multiple discounts applied to it. Here are the fields for a discount:

Sample JSON:


```
"discounts": [
 {
  "id": "C012",
  "name": "Comp",
  "amount": 40.45,
  "itemID": "5011"
 }
```

```
 {
    "id": "C012",
    "name": " Landlord/Prop Mgr Disco 20%"",
    "amount": 7.50,
    "itemID": "5012"
  },   {
    "id": "C013",
    "name": " Mall/Build Employee Disc 10%"",
    "amount": 12.50,
    "itemID": "5013"
  },   {
    "id": "C014",
    "name": " Police Officer Discount 10%",
    "amount": 15.00,
    "itemID": "5014"
  }, {
    "id": "C015",
    "name": " Senior Discount 10%",
    "amount": 4.45,
    "itemID": "5015"
  }
]
```

Parameters Details of Check Discounts:

Parameter	Description	Type	Required	Example
id	The id of item.	string	Yes	C015
quantity	The quantity of the item on which discount is applied.	decimal(2)	No	Senior Discount 10%
amount	The total amount applied as discount.	decimal(2)	No	4.45
itemID	The id of the item on which discount is applied.	string	No	5015

2.2.5 Check Comps

A check can have multiple comps applied to it. Here are the fields for a comps:


```
"comps": [
    {
      "id": "1",
      "qty": 3,
      "amount": 2.07,
      "itemID": "5001"
    },
```

```
    {
      "id": "2",
      "qty": 3,
      "amount": 2.07,
      "itemID": "5001"
    },    {
      "id": "3",
      "qty": 3,
      "amount": 2.45,
      "itemID": "5002"
    },
    {
      "id": "4",
      "qty": 5,
      "amount": 2.07,
      "itemID": "5003"
    },
    {
      "id": "5",
      "qty": 4,
      "amount": 7.90,
      "itemID": "5005"
    },
]
```

Parameters Details of Check Comps:

Parameter	Description	Type	Required	Example
id	The Id of the Comp.	string	Yes	5
quantity	The quantity of the Comped item.	decimal(2)	No	4
amount	Total amount of the comp.	decimal(2)	No	7.90
itemID	The id of the comped item.	string	No	5005

2.2.6 Check Voids

A check can have multiple voids applied to it. Here are the fields for a Void:

Sample JSON:

```
"voids": [
  {
   "id": "V010",
   "qty": 3,
   "amount": 15.15,
   "itemID": "5011"
  },
```

```
{
    "id": "V011",
    "qty": 4,
    "amount": 22.15,
    "itemID": "5011"
  }, {
    "id": "V012",
    "qty": 1,
    "amount": 3.99,
    "itemID": "5010"
  }, {
    "id": "V013",
    "qty": 3,
    "amount": 15.00,
    "itemID": "5014"
  },
{
    "id": "V014",
    "qty": 3,
    "amount": 10.00,
    "itemID": "5015"
  }
]
```

Parameters Details of Check Voids:

Parameter	Description	Type	Required	Example
id	The Id of the Void.	string	Yes	V014
quantity	The quantity of the Voided item.	decimal(2)	No	3
amount	Total amount of the void.	decimal(2)	No	10.00
itemID	The id of the voided item.	string	No	5015

2.2.7 Shift Data

A "shift" refers to a single employee's shift or time slip that was generated from an employee clocking in/out.

Sample JSON:


```
"shifts": [
 {
  "id": "10090",
  "employeeID": "1009",
  "jobID": "3",
  "startedAt": "2007-03-28T00:00:00",
  "busiDate": "2007-03-27",
  "startTime": "00:10:00",
  "endTime": "00:15:00",
  "endedAt": "2007-03-28T00:00:00",
  "shiftUpdatedAt": "2007-03-28T00:00:00",
  "totalTime": 300,
  "totalPay": 47.5,
  "payRate": 9.5,
  "cashTips": 0,
  "ccTips": 0,
  "otPayRate": 0,
  "paidBreakMinutes": 0,
  "unpaidBreakMinutes": 0
 },
```

```
{
  "id": "10092",
  "employeeID": "1002",
  "jobID": "3",
  "startedAt": "2007-03-28T00:00:00",
  "busiDate": "2007-03-27",
  "startTime": "00:15:00",
  "endTime": "00:20:00",
  "endedAt": "2007-03-28T00:00:00",
  "shiftUpdatedAt": "2007-03-28T00:00:00",
  "totalTime": 300.0,
  "totalPay": 47.5,
  "payRate": 9.5,
  "cashTips": 0.0,
  "ccTips": 0.0,
  "otPayRate": 0.0,
  "paidBreakMinutes": 0.0,
  "unpaidBreakMinutes": 0.0
   },
```

```
{
  "id": "10093",
  "employeeID": "1003",
  "jobID": "3",
  "startedAt": "2007-03-28T00:00:00",
  "busiDate": "2007-03-27",
  "startTime": "00:20:00",
  "endTime": "00:25:00",
  "endedAt": "2007-03-28T00:00:00",
  "shiftUpdatedAt": "2007-03-28T00:00:00",
  "totalTime": 300.0,
  "totalPay": 47.5,
  "payRate": 9.5,
  "cashTips": 0.0,
  "ccTips": 0.0,
  "otPayRate": 0.0,
  "paidBreakMinutes": 0.0,
  "unpaidBreakMinutes": 0.0
   },
```

```
{
 "id": "10094",
 "employeeID": "1004",
 "jobID": "3",
 "startedAt": "2007-03-28T00:00:00",
 "busiDate": "2007-03-27",
 "startTime": "00:25:00",
 "endTime": "00:30:00",
 "endedAt": "2007-03-28T00:00:00",
 "shiftUpdatedAt": "2007-03-28T00:00:00",
 "totalTime": 300.0,
 "totalPay": 47.5,
 "payRate": 9.5,
 "cashTips": 0.0,
 "ccTips": 0.0,
 "otPayRate": 0.0,
 "paidBreakMinutes": 20.0,
 "unpaidBreakMinutes": 30.0
  }
 ]
}
```

Parameters Details of Shift Data:

Parameter	Description	Type	Required	Example
id	Unique ID for a shift in a location. If the POS System does not provide a unique ID, you can try combining the given shift ID with the shift start time.	string	Yes	10094
employeeId	employee ID for the shift	string	Yes	1004
jobId	job ID for the shift	string	No	3
busiDate	date used for reporting the labor	date	Yes	2007-03-27"
startTime	time used for reporting the labor	time	No	00:25:00
endTime	business time the shift ends	time	No	00:30:00
startedAt	date & time the shift was started	datetime	No	2007-03-28T00:25:00
endedAt	date & time the shift ended	datetime	No	2007-03-28T00:30:00
shiftUpdatedAt	last time the shift was updated	datetime	No	2007-03-28T00:30:00

totalTime	total time for the shift (including overtime)	decimal(2)	No	300.0
totalPay	total cost for the shift (including overtime)	decimal(2)	No	47.5
payRate	pay rate	decimal(2)	No	9.5
otPayrate	pay rate for overtime	decimal(2)	No	10.50
ccTips	total credit card tips for the employee shift	decimal(2)	No	5.50
cashTips	total cash tips declared by the employee at clock out	decimal(2)	No	4.50
paidBreakMinutes	total number of paid break minutes	decimal(2)	No	20.0
unpaidBreakMinutes	total number of unpaid break minutes	decimal(2)	No	10.0

2.2.8 Paid In/Out PaymentData

A “paid in/out payment” refers to an incoming or outgoing payment using the paid in/out types from the configuration data. Negative amounts indicate paid out

Sample JSON:


```
{
 "paidInOut": [
  {
   "id": "P100",
   "amount": 100,
   "paidAt": "2017-03-28T00:00:00",
   "paymentTypeID": "1",
   "busiDate": "2017-03-28",
   "lastModifiedAt": "2017-03-28T00:00:00",
   "paidInOutID": "1",
   "employeeID": "23",
   "custAccountID": "13",
   "tip": 2
  },{
   "id": "P101",
   "amount": 200.0,
   "paidAt": "2017-03-2T00:00:00",
   "paymentTypeID": "1",
   "busiDate": "2017-03-29",
   "lastModifiedAt": "2017-03-29T00:00:00",
   "paidInOutID": "1",
   "employeeID": "23",
   "custAccountID": "13",
   "tip": 5.0
   },
```

```
{
    "id": "P102",
    "amount": 250.0,
    "paidAt": "2017-03-29T00:00:00",
    "paymentTypeID": "1",
    "busiDate": "2017-03-29",
    "lastModifiedAt": "2017-03-29T00:00:00",
    "paidInOutID": "1",
    "employeeID": "24",
    "custAccountID": "13",
    "tip": 10.00
    },
        {
    "id": "P103",
    "amount": 150.00,
    "paidAt": "2017-03-29T00:00:00",
    "paymentTypeID": "1",
    "busiDate": "2017-03-29",
    "lastModifiedAt": "2017-03-29T00:00:00",
    "paidInOutID": "1",
    "employeeID": "24",
    "custAccountID": "13",
    "tip": 25.00
    },
    ]
}
```

Parameters Details of Paid In/Outs:

Parameter	Description	Type	Required	Example
id	The Id of the payment	string	Yes	P103
amount	The total amount of the payment	decimal(2)	No	150.00
paidAt	The date and time of the payment	datetime	No	2017-03-29T00:00:00
paymentTypeID	The id of the payment type	string	No	1
lastModifiedAt	Last time the cash drawer amount was updated	datetime	No	2007-03-28T00:00:00
paidInOutID	Unique ID of Paid In and Paid Out	string	Yes	
busiDate	The business date on which the payment was made	date	Yes	2017-03-29
employeeID	Last time the payment was updated	datetime	No	24
custAccountID	The Id of the payment (Paid In/Out) reason	string	No	13

tip	The Id of the employee doing the payment	string	No	5.00

2.2.9 Cash Deposit

Cash Deposit refers to the amount of cash present in the cash drawer at the end of the day. Here are the fields for cash deposit.

Sample JSON:

```
{
 "deposits": [
  {
   "id": "73309",
   "amount": 1196,
   "busiDate": "2018-04-10",
   "lastModifiedAt": "2007-03-28T00:00:00",
   "depositedAt": "2007-03-28T00:00:00",
   "employeeID": "4018"
  },
  {
   "id": "73312",
   "amount": 166,
   "busiDate": "2018-04-10",
   "lastModifiedAt": "2007-03-28T00:00:00",
   "depositedAt": "2007-03-28T00:00:00",
   "employeeID": "4018"
  }
 ]
```

Parameters Details of Cash Deposits:

Parameter	Description	Type	Required	Example
id	Unique identifier of the cash deposit	string	Yes	73312
amount	Total amount deposited in the cash deposit/drawer	decimal(2)	No	66
busidate	Date used for submitting deposit	date	Yes	2018-04-10
lastModified At	Last time the cash drawer amount was updated	datetime	No	2007-03-28T00:00:00
depositedAt	Amount submitted into drawer	datetime	No	2007-03-28T00:00:00
employeeID	employee ID for cash deposit	string	No	4018