**Route: “/”**

Get Request For Login: {

Username: “Cade”,

Password: “1234”,

Status: “authorizing”

}

Get Respond For Login {

Username: “Cade”

Status: “Authorized”

}

Put Request For register: {

Name: “Cade Powers”,

Password: “1234”,

EmployeeId: 2345,

Address: “2134 Red Wolf Jonesboro, AR, 72142”,

PhoneNumber: “501-515-4086”

}

Put Respond For Register {

Username: “Cade”,

Status: “authorized”

}

**Route: “/accountsRecievable”**

Request Get request

Respond {{

AccountNumer: 001,

CompanyName: “Redstone”,

AmountOwed: 1200.23,

DateDue: “10/05/20”

},{

…

},…}

List of all the accounts with outstanding debt

**Route: “/accountsRecievable /paid”**

Post Request: {

AccountNumber: 001,

AmountPaid: 1000.00,

}

Post Respond {

Status: “posted”

}

**Route: “/inventoryManagement/partsManagement”**

Get Request: {

PartNumberPattern: “\*125A”

}

Get Respond: {{

PartNumber: “45125A”,

Description: “Gear”,

Quantity: 2,

Price: 120.00

},{

PartNumber: “523125A”

…

},…}

List of part numbers containing sent substring

**Route: “/inventoryManagement / partsManagement /detailed”**

Get Request: {

PartNumber: “45125A”

}

Get Respond: {

QuantityOO: 2,

Cost: 50.00,

Brand: “Cobra”,

Source: “Intercontinental”,

ClassID: 001

}

Post Request: {

PartNumber: “45125A”,

Quantity: 4,

QuantityOO: -2

}

Behavior of QuantityOO, The specified value sent is relative to the current value in DB. So -2 would subtract 2 from the current value in DB. Likewise, 2 would add 2 to the current value of DB.

Post Respond: {

Status: “posted”

}

**Route: “/inventoryMangement/ partsManagement /add”**

Put Request: {

PartNumber: “101-45214-000”,

\*Description: “Freightliner engine mount”,

Quantity: 8,

\*QuantityOO: 2,

Price: 50.00,

\*Brand: “Freightliner”,

Cost: 25.00,

\*Source: “Freightliner”

ClassID: 005

}

Put Respond: {

Status: “put”

}

‘\*’ delineate optional parameters should have default values if one is not provided.

**Route: “/inventoryMangement/classManagement”**

Get Request

Get Respond: {{

ClassID: 001,

ClassDescr: “Gears”,

Margin1: 1.2,

Margin2: 1.6,

Margin3: 2.0

}, {

…

},…}

In the class management nothing will be sent. The frontend will expect a list of all classes present in the db.

**Route: “/inventoryMangement/classManagement/update”**

Post Request: {

ClassID: 001,

ClassDescr: “Flywheels”,

Margin1: 1.4

}

Post Respond: {

Status: “posted”

}

In an update for a class the fields in the sent should be updated while the ones not provided should be left alone. If the classID sent is not in the db it should be added with what is included in the sent and what is not included should be defaulted.

**Route: “/customerManagement”**

Get Request: {

Name: “Red\*”

}

Get Respond: { {

ID: 001,

Name: “Redstone”,

Addr: “1234 Redstone St. Ruelle, AR, 72451”,

Phone: “501-405-5124”

}, {

…

},…}

List of all customers whose name contains the substring Request

**Route: “customerManagement/detailed”**

Get Request: {

ID: 001

}

Get Respond: {

BillingAddr: “POBox 203 Ruelle, AR, 72451”,

ShippingAddr: “ATTN: Robby 1234 Redstone St. Ruelle, AR, 72451

CityTax: 0.05,

StateTax: 0.1,

FederalTax: 0.025,

ChargeMax: 20000.00,

CurrentCharge: 10000.00

Post Request: {

ID: 001,

CityTax, 0.06

}

Post Respond: {

Status: “posted”

}

**Route “/customerManagement/add”**

Put Request: {

Name: “Cade Powers”,

Addr: “245 Redwolf Blvd. Jonesboro, AR, 72154”,

Phone: “501-515-4088”,

BillingAddr: “POBox 203 Ruelle, AR, 72451”,

ShippingAddr: “ATTN: Robby 1234 Redstone St.

Ruelle, AR, 72451”,

CityTax: 0.05,

StateTax: 0.1,

FederalTax: 0.025,

ChargeMax: 20000.00,

CurrentCharge: 10000.00

}

Put Respond: {

Status: “put”

}

Values not provided should be defaulted and cash accounts should use local tax requirements which will be global variables. If this account was a charge account tax information would be necessary.

**Route “/employeeManagement”**

**Route “/employeeManagement/detailed”**

**Route “/employeeManagement/add”**

Employee management will follow the same pattern as the customer management with the fields adjusted.

**Route “/partsCounter/invoice”**

Get Request {

CustomerName: “Red\*”

} OR {

ID: 30001

}

If CustomerName is in the sent, then the server should return a list of all invoices with the customer containing the sent pattern and they should be sorted in descending order based on date. If the ID is sent, then the server should return the details of the invoice under that ID.

Get Respond if name is sent: {{

ID: 30001,

Customer: “Redstone”,

Date: “10/07/2020”,

FirstPart: “101-45214-000”,

Total: 1000.23

},{

…

},…}

Get Respond if ID is sent {

A list of all details about the invoice that has the ID sent.

}

**Route “partsCounter/invoice/add”**

Get Request: {

CustomerID: “001”

}

Get Respond: {

Name: “Redstone”,

BillingAddr: “POBox 203 Ruelle, AR, 72451”,

ShippingAddr: “ATTN: Robby 1234 Redstone St.

Ruelle, AR, 72451”,

Phone: “501-405-5124”,

Email: “test@google.com”

}

Get Request: {

PartNumber: “1245AV”

}

Get Respond: {

Description: “Gear”,

PriceM1: 140.00,

PriceM2: 160.00,

PriceM3: 200.00,

Cost: 100.00,

Quantity: 2

}

Request for each part added to invoice

Get Request: {

CustomerID: “001”,

Parts: “{1245AV, 2,160.00},{101-45251-000,4,80.00},…”

}

Get Respond: {

SubTotal: 640.00,

TaxTotal: 112.00,

Total: 752.00

}

Request to get totals after all parts are added

Put Request: {

CustomerID: 001,

EmplyeeId: 004,

Parts: “{1245AV, 2,160.00},{101-45251-000,4,80.00},…”,

SubTotal: 640.00,

TaxTotal: 112.00,

Total: 752.00

}

Put Respond: {

InvoiceID: 3002

}

Once the invoice is ready to be posted the final put request is sent.

**Route: “partsCounter/quote”**

**Route: “partsCounter/quote/add”**

Follows patter of invoice with the id’s first number decrementing to 2. For example an invoice ID would be 3001 and a quote ID would be 1001.

**Route: “partsCounter/workOrder”**

**Route: “partsCounter/workOrder/add”**

**Route: “partsCounter/workOrder/update”**

Follows the pattern of invoice but the Id is 2001 and fields are adjusted properly.

**Route: “partsCounter/history”**

Get Request: {

Type: 1,

\*Customer: “Red\*”,

\*PartNumber: “101-254\*”,

\*Date: “10/05/2020”

}

Get Respond: {

A list where the patterns sent are held in the invoices.

}

Any combination of these parameters should be accepted. Type tells the server where to Get the history from. The number is related to the initial number in the ID’s. So 1 would be quotes, 2 would be work orders, and 3 would be invoices.