

**Smart India Hackathon 2017 Project Documentation**  
**Team: TechWhiz**  
**Project Title: Barcode based Hardware Inventory Management**  
**Ministry: Department of Posts**

***Contents:***

<b>1. PROBLEM STATEMENT</b>	<b>3</b>
1.1 Introduction to DOP and existing scenario of Hardware Inventory Management	3
1.2. Problem Description	4
<b>2. GOALS OF THE SYSTEM</b>	<b>5</b>
<b>3. SOFTWARE REQUIREMENT SPECIFICATION</b>	<b>6</b>
User Requirements:	6
Functional Requirements:	6
Non-Functional requirements:	7
External Interface Requirements	8
<b>4. KEY DESIGN DECISIONS</b>	<b>9</b>
<b>5. SYSTEM ARCHITECTURE</b>	<b>12</b>
<b>6. FEATURES OF THE SYSTEM</b>	<b>13</b>
<b>7. DATABASE SCHEMA</b>	<b>14</b>
<b>8. USE CASE DIAGRAMS</b>	<b>16</b>
<b>9. SOFTWARE TECHNOLOGIES USED</b>	<b>21</b>
Front End Technologies	21
Server Technology	21
Database Technology	22
<b>10. BACKEND SERVER DOCUMENTATION</b>	<b>23</b>
10.1 A note on JSON Web Tokens (JWT):	23
10.2 Endpoint Documentation	23
10.2.1 Login Endpoint:	23
10.2.2 Raise Request Endpoint:	24
10.2.3 Pending Requests Endpoint:	24
10.2.4 Pending Approvals Endpoint:	25
10.2.5 Approve Request Endpoint:	26
10.2.6 Approved Requests Endpoint:	27
10.2.7 Device Search Endpoint:	27
10.2.8 Install Devices Endpoint	28
10.2.9 View Inventory Endpoint:	29

<b>11. PROGRESS TIMELINE</b>	<b>32</b>
<b>12. END USER MANUAL</b>	<b>33</b>
About	33
Register	33
Login	35
Raise Request	36
Pending Requests	38
Pending Approvals	38
Install	40
View Inventory	42
Logout	43

# 1. PROBLEM STATEMENT

## 1.1 Introduction to DOP and existing scenario of Hardware Inventory Management

The **Department of Posts (DoP)**, trading as **India Post**, is a government-operated postal system in India. Generally referred to within India as "the post office", it is the most widely distributed postal system in the world. The postal service is under the Department of Posts, which is part of the Ministry of Communications of the Government of India.

A post office is a customer service facility forming part of a national postal system. Post offices offer mail-related services such as acceptance of letters and parcels; provision of post office boxes; and sale of postage stamps, packaging, and stationery. In addition, many post offices offer additional services: providing and accepting government forms (such as passport applications), processing government services and fees (such as road tax), and banking services (such as savings accounts and money orders).

The country has been divided into 23 postal circles, each circle headed by a Chief Postmaster General. Each circle is divided into regions, headed by a Postmaster General and comprising field units known as Divisions. These divisions are further divided into subdivisions. In addition to the 23 circles, there is a base circle to provide postal services. With 155,015 Post Offices, the DoP has the most widely distributed postal network in the world. Each of the post offices have a large collection of hardware devices like printer, scanner, barcode reader, computer, monitor, etc. So each post office may contain at least 20 hardware devices accounting to more than 3,100,300 ( $155,015 \times 20$ ) devices in total. Hence it is necessary to have a centralized database, that holds all the device informations and the history of that devices.

In the existing scenario for hardware management in post offices across India, most records of inventories are still maintained manually, through written logs and printed copies of information about the devices installed in the respective post offices. Some post offices are gradually modernizing, and maintain electronic records of devices installed in them with essential information such as cost, manufacturer, duration of warranty, device specifications, date of installation, number of times serviced, etc. These electronic records are usually maintained in excel spreadsheets or local databases, and there is little synchronization between records maintained by different post offices, and often there is little similarity in the structure of the data records maintained. Thus there is a need for a central management of device inventories across all post offices.

Buying a device in a post office is a tedious task. If suppose a local post office needs a scanner, it cannot directly buy the device. The local post office official needs to raise a request to buy a device. The division officer is the first major deciding authority who needs to allocate funds to the post offices under his control. Suppose if a division officer gets a request from any of his undertaking post offices, he can either approve the device or can forward the request to a higher official. Approving of the devices has another major concern. Consider two types of post offices, one office have more work load and other office have very less workload. When a printer in the latter post office becomes defective and they need

a new printer, it will be efficient to divert the printer from post office 1 to post office 2 and then buy a new printer in post office 1 where the workload is heavy. This scenario is called as diversion of devices from one post office to another. This decision should be taken by the authorizational person.

Once the device gets dispatched, it needs to be installed in the respective post offices. So there comes the requirement of a centralized system accessed by all the post offices, that captures all processes and scenarios involved in procuring and maintaining hardware inventory.

## **1.2. Problem Description**

With about 3 crore devices in post offices across India, the handling and processing of all the devices becomes a tedious task. Buying of the devices goes through a complex process and their current status needs to be properly updated in a centralized database. History of devices should be properly stated and also the warranty of devices should be maintained. A proper tracking mechanism should be maintained from the requesting of the device to the installation of device in the respective place. A formal procedure should be incorporated for all the processes in procuring and maintaining the devices in the hardware inventory.

The above problem is modelled using a Barcode Based Hardware Inventory Management System (BHIM) that uses barcode to track the devices and handles various process involved in installation and maintaining of the devices. The system is built by using Vuejs for frontend, flask for RESTful API and the backend database is using PostgreSQL.

## **2. GOALS OF THE SYSTEM**

- The main goal of the system is to provide an efficient and user-friendly inventory management system with almost zero modifications to the existing infrastructure and zero learning curve.
- The system should have a proper asset tracking system that keeps track of all the devices.
- The system should capture all the use cases in procuring of the devices. That is all the request and approvals should be captured in the system.
- The system should have a proper tracking mechanism from the requesting of the device to the installation of device in the respective place.
- There should be an efficient search model to search for devices under a specific user.
- The system should reduce the manual labour to the possible extent.
- The system should properly maintain the history of devices, their previous post offices, warranty etc.
- The system should have a highly secure authentication and the data should be consistent.
- There should be secure transmission of information and the data should be checked for authorization.
- The system should be easy to handle by all members in the post office.
- The system should be easy to learn.
- The system should be able to capture the existing hierarchy in the post offices.

### 3. SOFTWARE REQUIREMENT SPECIFICATION

#### User Requirements:

The main users of the system are System Admin and authoritative users of post office. By using this system, users can access the details of devices under their governance and also manage them. System admin should be able to login to the system ,raise request for devices and install the devices once it has been approved.The authoritative users should be able to login to the system and approve/disapprove/escalate the requests sent to him (based on decision) .

#### Functional Requirements:

S.No	Requirement Name	Requirement Description
1	Login	This function will enable the System Admin/Authoritative users to login to the system
2	Register	This function will enable the System Admin/Authoritative users to register into the system
3	Change password	This function will enable the System Admin/Authoritative users to change password
4	View Inventory	This function will enable the System Admin/Authoritative users to view the inventory details of the post offices under his governance
5	Raise Request	This function will enable the System Admin to place request for devices needed in his post office
6	Pending Request	This function will enable the System Admin to view the status of the requests placed during the raise request operation
7	Pending Approval	This function will enable the Authoritative user to either approve/disapprove/escalate the requests(to higher authority) placed by the system admin
8	Install Device	This function will enable the System Admin to install the approved devices in the system
9	Logout	This function will enable the System Admin/Authoritative users to logout of the system

## **Non-Functional requirements:**

### **Security**

- ☐ The system must have protection from unauthorized users .
- ☐ The system should show error if the username or password is incorrect
- ☐ To use the system, every user has to login by keying in their username and password

### **Usability**

- ☐ The system must be easy and accessible to use by all types of users (SysAdmin and Authoritative) such that they do not need to read an extensive amount of manuals.
- ☐ The system must be intuitive and simple in the way it displays all relevant data and relationships.
- ☐ The menus of the system must be easily navigable by the users with buttons that are easy to understand.

### **Reliability**

- ☐ The System must give accurate inventory status to the user continuously. Any inaccuracies need to be taken care instantly.
- ☐ The system must provide a password enabled login to the user to avoid any foreign entity changing the data in the system.
- ☐ The system should provide the user updates on completion of requested processes and if the requested processes fail, it should provide the user the reason for the failure.
- ☐ The system should not update the data in any database for any failed processes.

### **Performance**

- ☐ The system must not lag, and must provide quick responses to users' requests and queries.
- ☐ The system must complete updating the databases, installing a device, raising a request and approving or disapproving a request successfully every time the user requests such a process.
- ☐ All the functions of the system must be available to the user every time the system is turned on.
- ☐ The calculations performed by the system must comply according to the norms set by the user and should not vary unless explicitly changed by the user.

### **Supportability**

- ☐ The software is designed such that it works even on systems having the minimum configuration.
- ☐ The system is adaptable even if additional plugins or modules are added at a later point.

### **Packaging**

- ☐ The packaging must come with a manual that details the use of the system, and also the instructions on how to use the program.

- ❑ This manual may be included either in a booklet that comes with the software, or on the disc that the software itself is on.

### **Interfacing**

- ❑ The system must offer an easy and simple way of viewing the current inventory.

### **External Interface Requirements**

#### **User Interface**

- ❑ The System User Interface is built using HTML, CSS and Vue js framework.

#### **Hardware Interface**

- ❑ A browser which supports HTML and Javascript
- ❑ The system should be connected to the Internet

#### **Software Interface**

- ❑ The backend server application is written in Python using the Flask microframework.
- ❑ The database is implemented using PostgreSQL, consisting of RDBMS tables.
- ❑ PyGreSQL, an open-source Python module that interfaces to a PostgreSQL database is used for connecting the database and the System.



## 4. KEY DESIGN DECISIONS

The purpose of this section is to outline some of the important design decisions that were made while architecting the system and its use cases, and explain the reasoning behind why the decisions were made. The section also explores the potential trade-offs associated with these decisions and assesses the significance of these trade-offs.

### 1. Decision: **A barcode-based system**

#### Reason:

A barcode reader is an efficient input device that is used for various purposes. Scanning a barcode takes far less time than having to manually enter some item ID for a given device. It was believed that having a system where devices are identified by barcodes will allow for ease of use and higher user-friendliness. Moreover, post offices around India already possess barcode scanners which could be easily interfaced with a computer system for this purpose.

#### Trade-offs:

It is possible that certain devices may not contain barcodes, or that the barcode is attached to the device in an angle/orientation that is difficult to pick up by a scanner. For this purpose, device serial numbers are also maintained in the system, and can be used as alternative references for a device.

### 2. Decision: **Relational Database Management System (RDBMS) in the backend**

#### Reason:

No-SQL and other unstructured database systems are becoming increasingly popular and offer features like indexing and dynamic querying. However, for this particular application, we found that an RDBMS would better suit the architecture since the inventory management has conventionally been implemented with SQL-like database systems, and with resounding success.

#### Trade-offs:

The database was normalized to the maximum possible extent, although not fully. The reason for this was to avoid data retrieval queries from becoming increasingly complex (multiple degrees of nesting or natural joins), but at the cost of allowing certain levels of redundancies within the database. We believe that an optimum balance has been struck between query optimization, minimizing the number of DB queries and full-blown normalization of the schema.

### 3. Decision: **Token-based client server communication**

#### Reason:

To reinforce security and authentication, each user-client is assigned a token upon successful login. The information and identity of the user is embedded within the token and must be relayed to the server for all subsequent client-server interactions. This establishes identity and also prevents arbitrary third parties from gaining access to critical application data, without a token.

#### Trade-offs:

Encrypting the user information to generate the token and decryption of the token to verify its validity is an additional overhead to the server program; these functions are

triggered with reasonable frequency. However this overhead is considered a trade-off to ensure secure access to the system.

4. **Decision: System consists of only two types of users: system admins and authoritative users**

**Reason:**

Domain research indicated that system admins are the employees with technical know-how to make requests for devices and install said devices once they are procured by a post office. Authoritative users are the decision makers of what requests to approve and procuring the devices requested from the vendors. Though the actual employee roles in the Department of Posts is far more extensive than just these two roles, it was decided that only these two types of users would be involved in the management of hardware inventory and hence it would be sufficient to model these two roles alone in the system.

**Trade-offs:**

There are several types of system administrators, as well as authoritative users. They take different titles depending on the level of the post office they operate in, and have minor differences in their budgetary privileges; these variations were foregone while designing the system in order to keep things simple for the user and avoid confusion.

5. **Decision: To not incorporate functionality for authoritative users of the same post office, to authorize requests submitted by system admins**

**Reason:**

Initially, it was decided that the authoritative user (such as postmaster general, etc.) should be kept informed of the happenings of the post office, and thus would be required to authorize device requests submitted by the respective post office system admins, before the request escalates to the appropriate approving authority. In other words, the system was designed to route every request through an appropriate higher official for authorization before dispatching it to the approver.

However, upon further discussion, it was found that it was impractical for someone in the ranks of a postmaster to focus attention on authorizing requests. Moreover, intimation about requests raised happens offline through pen-and-paper documentation anyway. Therefore, this user flow was skipped while implementing.

6. **Decision: Implementation technology (Vue.js + Dockerized Python Flask server):**

**Reason:**

The main reason for deciding upon the above-mentioned technology for implementation was to allow for easy, incremental development, while at the same time making the software system open to extensions and modifications if required. The database operations and server logic are completely abstracted from the Front End Application code. Moreover, the choice to Dockerize the server was to ensure that it could be readily deployed in house without the hassle of installing dependencies and requirements.

NOTE: More details about these technologies can be found in Chapter 9 - Software Technologies Used.

7. **Decision : To not allow user to specify device cost while raising request**

**Reason:**

There is no option given to specify cost for the device by the sysadmin while raising a request to allow the decision making authority to decide the approval budget based on offline research and elicitation. Also the decision making authority can decide to either procure a new device or divert an existing device from another office. This also reduces the work of the system admin as he only needs to worry about raising the request and not calculate the budget for the request.

8. **Decision : To allow partial approvals and partial escalations**

**Reason:**

As mentioned earlier, the system was designed with the goal of providing the utmost flexibility for the end user, and accurately modelling the processes and procedures involved. During our domain knowledge acquisition, we discovered that authoritative users might often not approve a request entirely in one go. He may wish to approve a certain device like a printer first, and then approve some other device, like a scanner, after due consideration at a later point in time. Moreover, since the budget of authoritative is not uniform, the system also offers the option of escalating a request to higher ranked authoritative user, if it falls out of their mandate.

9. **Decision : To not allow system admin to specify urgency/priority during request raise**

**Reason:**

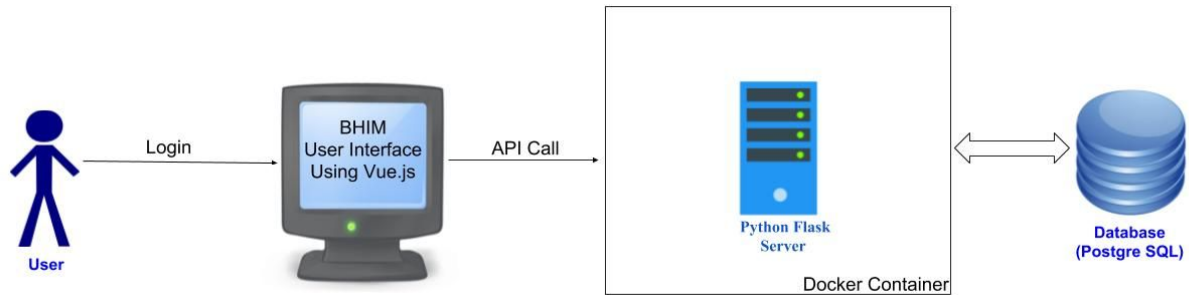
It was observed that as a general trend, system admins who make requests tend to mark all device requests as urgent/having high priority. Thus allowing an option to set a priority for requests would not contribute to the betterment of the system. It was decided to internally sort requests by their date of request to prioritize earlier requests for approval, instead of including an explicit priority tag.

10. **Decision : For allowing all lower level requests directly to Division Level**

**Reason:**

Based on our initial assumptions we routed all device requests to the parent post office. But later we came to know that all decision making power related to procurements and diversions of devices was decided by the Division Level Post offices. Hence there was no need to route the requests of lower level offices to their parent P.O, instead it was decided to directly route it to its corresponding Division Post Office.

## 5. SYSTEM ARCHITECTURE



The users of the application login into the application via BHIM user interface. The BHIM user interface is implemented using Vue.js. It is an open-source JavaScript framework for building user interfaces. Integration into projects that use other JavaScript libraries is made easy with Vue.js because it is designed to be incrementally adoptable. Vue can also function as a web application framework capable of powering advanced single-page applications. User can view the devices in the post offices, can request for additional devices, they can approve/reject or escalate the device request.

Once any of the above operation is performed, an API call is sent to backend flask server deployed via a docker container on a linux server. Main Server is written in Python Flask and contains the code for all endpoint activities. The Server interacts with PostgreSQL Database. Server is deployed and runs within a Docker container. Thus a JSON response is sent from the server to the front end, where it is processed and displayed back in the BHIM user interface.

## 6. FEATURES OF THE SYSTEM

The main objective of the system is to provide an efficient and user-friendly inventory management system with almost zero modifications to the existing infrastructure and zero learning curve. These are offered by the following features of the system:

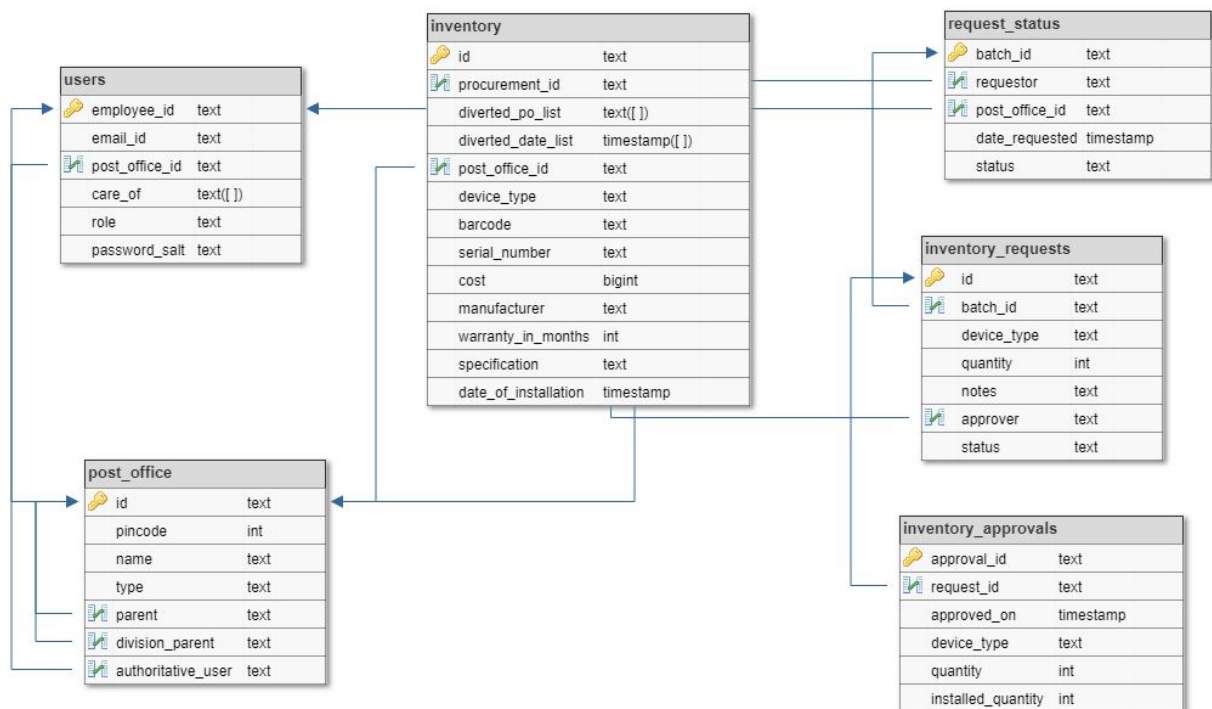
1. The most important feature of any inventory management system is asset tracking. This is done efficiently in our system with the help of barcodes and/or serial numbers.
2. Security is ensured in the system with the use of JWT (JSON Web Tokens). It is useful both for authentication and information exchange.
  - a. **Authentication:** This is the most common scenario for using JWT. Once the user is logged in, each subsequent request will include the JWT, allowing the user to access only the resources that are permitted with that token.
  - b. **Information Exchange:** JWTs are a good way of securely transmitting information between parties. Additionally, as the signature is calculated using the header and the payload, one can also verify if the content has been tampered or not.
3. Barcode based system. Saves a lot of time compared to manual data entry.
4. An efficient end-to-end system that captures all processes and scenarios involved in processing and maintaining hardware inventory.
5. Flexibility for users. System places no constraints on users in performing tasks (e.g. partial approval, escalation, etc.).
6. The *filter* feature provides the user with the capability to view inventory details of various post offices under their governance.
7. A one-stop repository to track various devices' history such as tracking inventory as it is transported/diverted between different post offices.
8. The system captures the existing hierarchy in the post offices (Level 1 - Directorate, Level 2 - Circle Office, Level 3 - Regional, Level 4 - Division, Level 5 - Head, Level 6 - Sub, Level 7 - Branch) enabling escalation of requests and approval of requests to be handled appropriately at the specified level of the hierarchy.
9. The system categorizes the various types of devices into well-defined categories thereby enabling easy and efficient search operations.
10. The system provides with up-to date details of the inventory, tracking the active hardware devices. These up-to date, accurate and complete information can be used to make various analytical and business decisions.
11. Every stage of the status of a request is captured, thereby keeping the user informed at all the times.

## 7. DATABASE SCHEMA

While designing the database, due consideration was taken to normalize the tables as much as possible, in order to avoid redundancies and potential scope for inconsistencies within the DB fields.

The decided schema consists of a total of 6 tables, required to model the necessary specifics of the system including the post office, users, inventory and requests for procurement.

The following diagram depicts the tables in the system database and illustrates the relationships (such as foreign key dependencies) between them.



The following is a brief description of the tables and their significance in the system:-

### 1. **post\_office**

- This table maintains the information of all the post offices, such as the pincode, the type (directorate, division, head, branch, etc.).
- The hierarchy is captured using the parent and division\_parent fields, which are foreign keys pointing to the corresponding parent PO so as to build the hierarchy of PO's as a tree.

### 2. **users**

- This table maintains user information such as *employee\_id*, *email*, *password\_hash*, etc.
- The *role* field indicates if the user is *authoritative* or *sysadmin* and takes only these two values

- c. Only a hash of the password is stored, and not the actual password, to enhance security.
- d. The *care\_of* field maps the user to a list of PO's which he is incharge of, and can make requests for.

### 3. ***request\_status***

- a. This table keeps track of the status of the various batches of requests that enter the system.
- b. Each batch request has a requestor, and a *post\_office\_id* for which the request is being made.
- c. The *status* field can take values like *open*, *partially\_approved*, *approved*, etc.

### 4. ***inventory\_requests***

- a. A batch request consists of a list of *devices* requested along with the *quantities* required for each device. This information is stored in the table, along with the *employee\_id* of the *approver*.
- b. Request escalations are handled by updating the *approver* field.

### 5. ***inventory\_approvals***

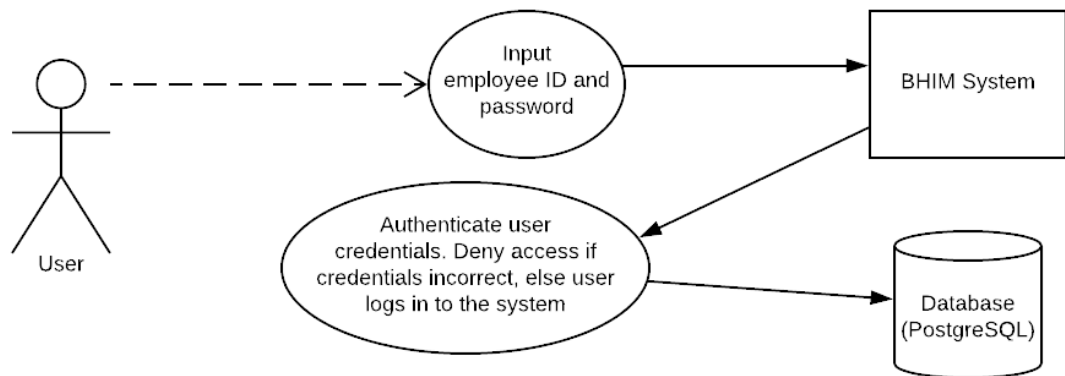
- a. This table corresponds to the *inventory\_requests* table, to which it holds a foreign key relationship.
- b. The *quantity* of devices approved for the request is captured in this table.
- c. The field *installed\_quantity* is used to count if all devices that were approved. have been installed in the requesting PO.

### 6. ***inventory***

- a. Tuples in this table are populated when an approved device is finally installed in the PO, or updated when a diverted device is installed.
- b. Information about the device is stored in 6 self-explanatory fields - *barcode*, *serial\_number*, *cost*, *manufacturer*, *warranty\_in\_months*, *specification*.
- c. In addition, foreign key relationships are used to link the devices with their corresponding *request\_id* and the *post\_office*.
- d. The *diverted\_po\_list* and *diverted\_date\_list* is the list of post offices through which the device was diverted and the dates on which they were installed in those post offices.

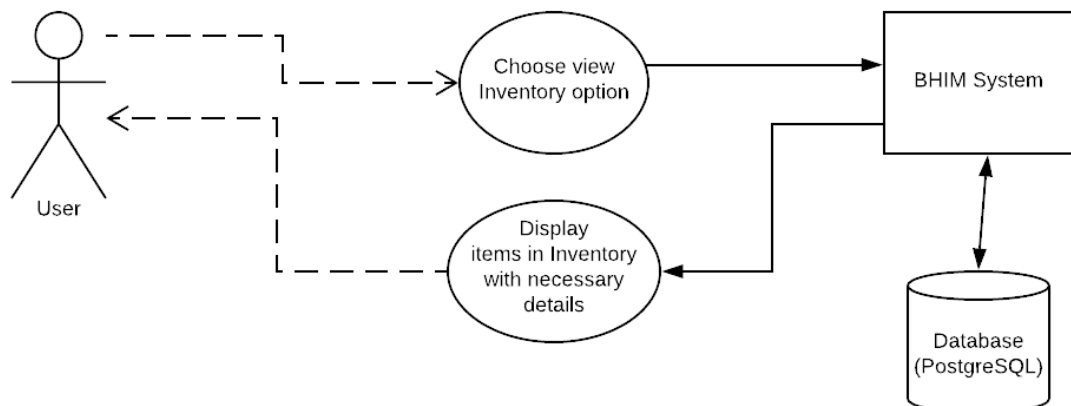
## 8. USE CASE DIAGRAMS

### 1. Login use case



The user enters credentials such as employee ID and password in order to log in to the BHIM system. The credentials are authenticated before a user can successfully log in. In case of incorrect credentials, an error message is displayed.

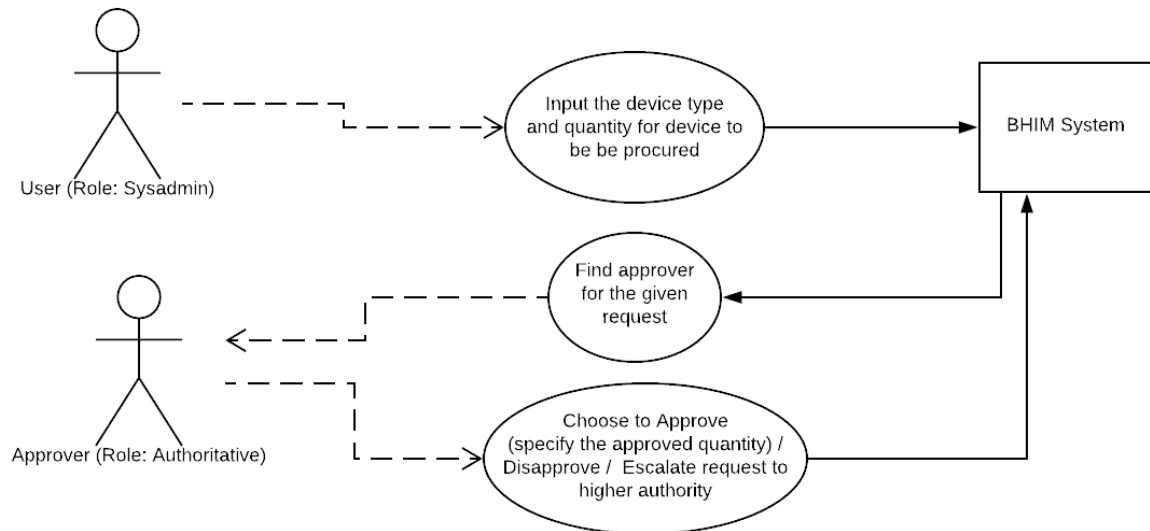
### 2. View Inventory use case



The user chooses the 'View Inventory' option to view the list of all devices installed at their post office (P.O). This list of devices is retrieved from the database and displayed to the user.

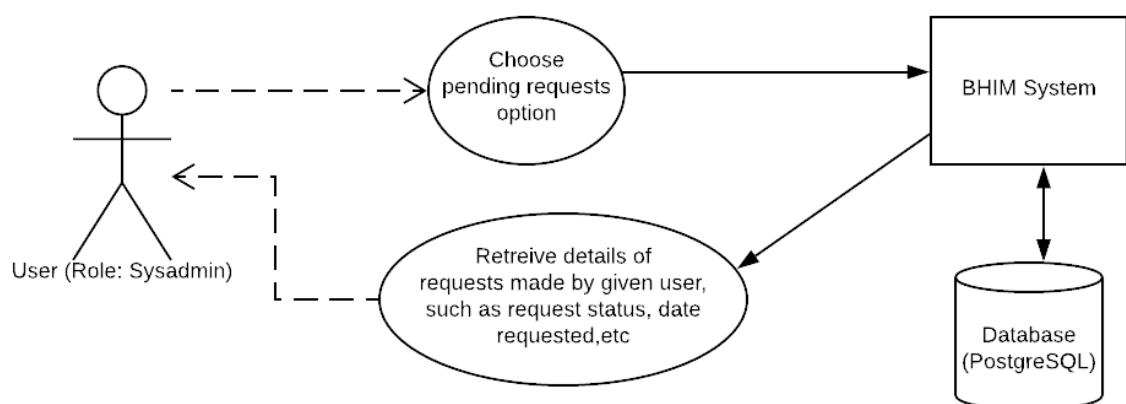


### 3. Raise Request use case



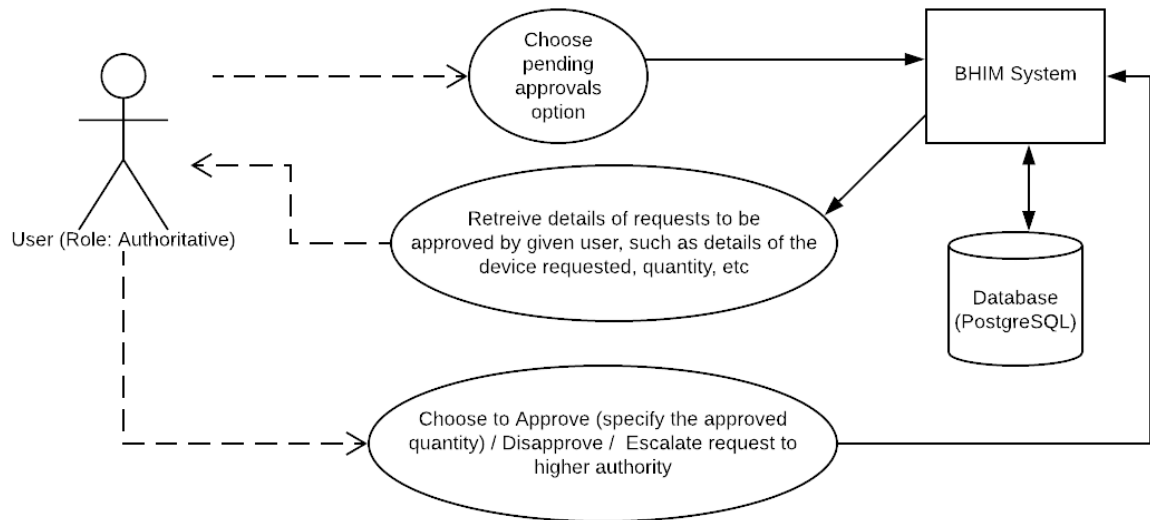
A System Admin can access this function. To raise a request, the user inputs the type and quantity of the new device to be procured. The appropriate user to approve and sanction this request is notified of this request by the system. The 'approver' who is an Authoritative user, then chooses to either approve (by specifying the approved quantity for the requested device type) / disapprove/ escalate the request to a higher authority. After an action is taken by the approver, the status of the request is changed accordingly. In case of escalation, the approver is changed to the higher authority.

### 4. Pending Request use case



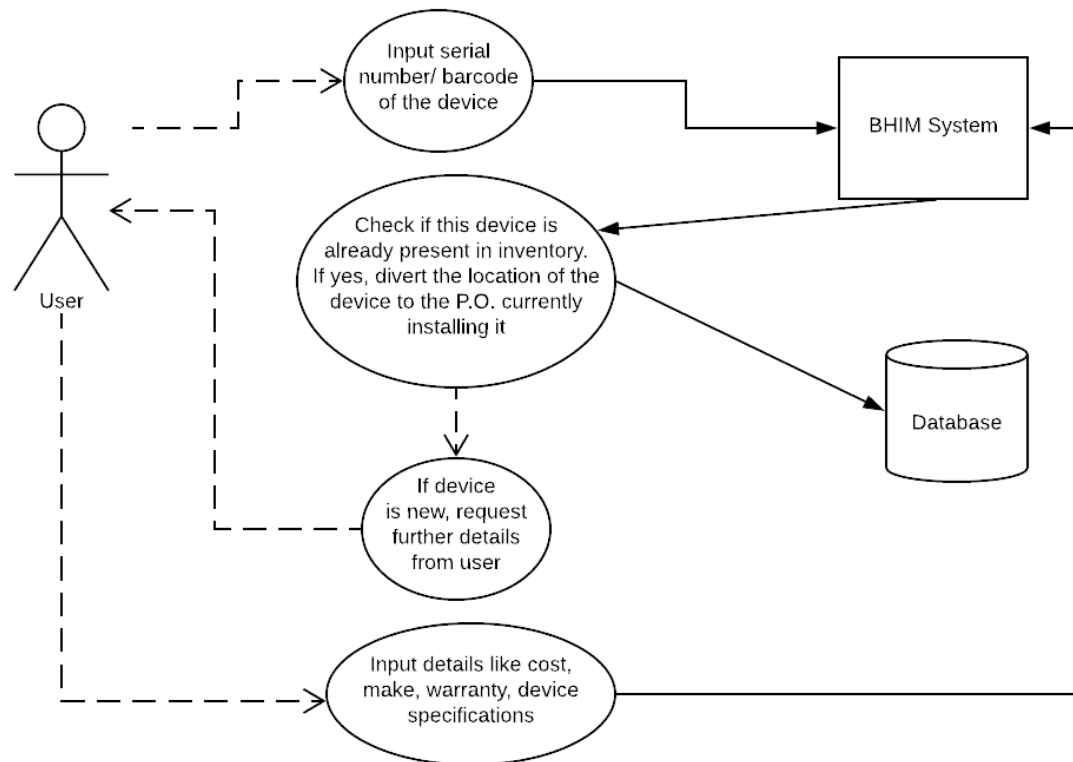
This function is used to view the status of the requests made by a system admin. The details of the device requests made by the given user, such as request status, type and quantity of device requested, date of request, etc are retrieved from the database and displayed to the user.

## 5. Pending Approval use case



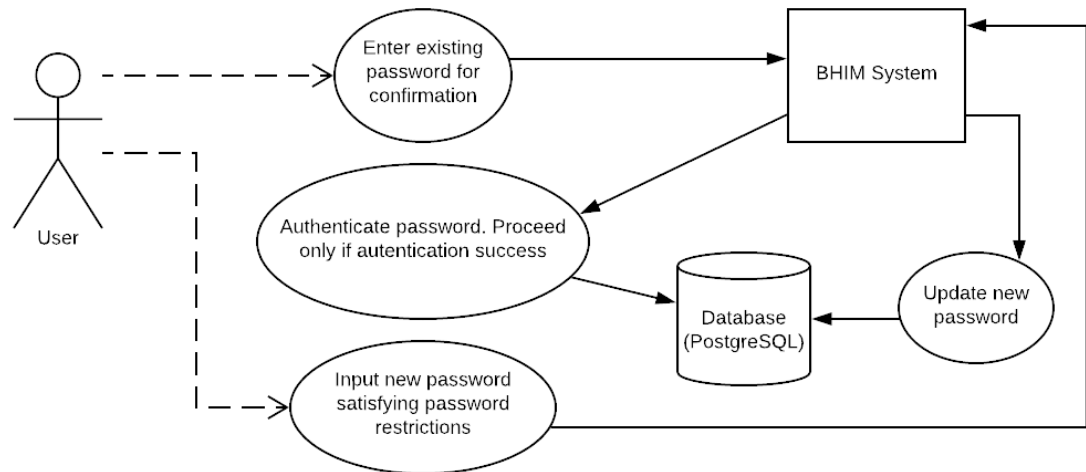
This function can be accessed by an authoritative user. It is used to view the requests for which the given user has been designated as the approver. The details of requests to be approved, such as type and quantity of device requested, date of request are listed. The approver can then choose to either approve (by specifying the approved quantity for the requested device type) / disapprove/ escalate the request to a higher authority. After an action is taken by the approver, the status of the request is changed accordingly. In case of escalation, the approver is changed to the higher authority.

## 6. Install device use case



When a new device that was requested for by the P.O has arrived, this function is used to register it into the inventory. The serial number/ barcode of the device is entered by the user. The system then checks if the device id already present in the inventory in which case it means that the device in question has been transferred/ diverted to the given P.O. In this case, the location of the device is updated in the database. Else, the system prompts the user to enter more details of the new device, such as cost, duration of warranty, manufacturer, etc which is then stored in the inventory.

## 7. Change password use case



A user can change the password for their account through this function. The system asks for the existing password for the account. Upon verifying it, the user is prompted to enter the new password, which must satisfy length and complexity restrictions. Then, the new password is updated in the database.

## 9. SOFTWARE TECHNOLOGIES USED

Based on the design of the system, use cases and feasibility of extension/modification, it was decided to implement the software using the following technologies:-

### 1. Front End Technologies

Front end consists of a collection of web pages interacting with each other and the server program.

#### a. HTML and CSS

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web.

#### b. Vue.js - JavaScript Framework

Vue (pronounced /vjuː/, like view) is a progressive framework for building user interfaces. Unlike other monolithic frameworks, Vue is designed from the ground up to be incrementally adoptable. The core library is focused on the view layer only, and is easy to pick up and integrate with other libraries or existing projects. On the other hand, Vue is also perfectly capable of powering sophisticated Single-Page Applications when used in combination with modern tooling and supporting libraries.

For documentation and guide, see: <https://vuejs.org/>

### 2. Server Technology

The backend server application was written in Python using the Flask microframework. The server implements the backend logic and interacts with the database.

#### a. Flask

Flask is a simple yet efficient server microframework in Python that allows for modular development of RESTful APIs for web applications. It is open sourced and has a BSD licence. The reason for choosing Flask for the server was the readability and the opportunity to leverage lot of the useful functionalities of other Python libraries.

For documentation and information, see: <http://flask.pocoo.org/>.

#### b. PyGreSQL

PyGreSQL is an open-source Python module that interfaces to a PostgreSQL database. It embeds the PostgreSQL query library to allow easy use of the powerful PostgreSQL features from a Python script or application. Though there are several other Python libraries available to connect to PostgreSQL

database servers, PyGreSQL was chosen owing to its powerful functionality and simplistic usage.

Website for information and documentation: <http://www.pygresql.org/>

Some example code: <http://www.pygresql.org/contents/postgres/basic.html>

### **3. Database Technology**

The database consists of a Relational Database Management System, consisting of tables. PostgreSQL was the chosen technology for the database system.

#### **a. PostgreSQL**

PostgreSQL is a powerful, open source object-relational database system with over 30 years of active development that has earned it a strong reputation for reliability, feature robustness, and performance. Moreover, it implements the SQL standard well and supports lot of advanced data-types such as (multidimensional) arrays, JSON data types and user defined datatypes.

Official website of PostgreSQL: <https://www.postgresql.org/>

## 10. BACKEND SERVER DOCUMENTATION

This is the developer documentation intended to provide extensive explanation of the API endpoints of the server program. The server program is written as a modular, RESTful APIs that use a JSON-based request-response formats for client-server interaction. The endpoints in the server are consumed by the front end web application. The logic and the functions for database retrieval involved in the various user flows of the application are contained within the server program.

### 10.1 A note on JSON Web Tokens (JWT):

JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed. JWTs can be signed using a secret (with the HMAC algorithm) or a public/private key pair using RSA.

The main advantages of using JSON Web Tokens are:-

1. **Compact:** Because of their smaller size, JWTs can be sent through a URL, POST parameter, or inside an HTTP header. Additionally, the smaller size means transmission is fast.
2. **Self-contained:** The payload contains all the required information about the user, avoiding the need to query the database more than once.

The application makes extensive use of JWT tokens for authentication purposes as well as transmitting user information. The tokens are generated upon successful login of a user and contain the user information embedded within them, enabling any client that submits a request to the server to be identified by decoding the token to retrieve user information.

### 10.2 Endpoint Documentation

#### 10.2.1 Login Endpoint:

Simple endpoint that allows the user to login to the system. Generates and returns a JSON web token(JWT) along with user role if the login was successful.

Request Type: `POST`

Endpoint URL: `BASE_URL + /login`

Request JSON Format:

```
{  
  "employee_id": "e123po",  
  "password": "e1pwd"  
}
```

Login Success Response JSON (status code 200):

```
{  
  "jwt_token": "some-jwt-token",  
}
```

```

    "role": "sysadmin" or "authoritative"
    "success":true
}
Login Failure Response JSON (status code 200):
{
    "success":false,
    "message":"Invalid email or password"
}

```

### **10.2.2 Raise Request Endpoint:**

Allows a user to raise a batch request. A batch is a collection of devices along with the quantity required for each device, something similar to a conventional shopping cart. This endpoint creates a batch and its status is set as `open`; along with which a request is created for each device type in the request, and its status is marked as `open` as well. These are populated in the databases.

Request Type: POST

Endpoint: `BASE_URL + "/request/raise"`

HTTP Headers: `Authorization: Bearer some-jwt-token`

Request JSON Format:

```

{
    "post_office_id":"po5"
    "batch_data":[
        {
            "device_type":"printer",
            "quantity":2,
            "notes":"d1"
        },
        {
            "device_type":"scanner",
            "quantity":3,
            "notes":"d2"
        }
    ]
}

```

Success Response JSON (status code 200):

```

{
    "success":true
}

```

### **10.2.3 Pending Requests Endpoint:**

Allows a system admin to view the current status of the requests raised for the post offices in his care, or for devices to be obtained in the sysadmin's post office.

Request Type: GET



Endpoint: `BASE_URL + "/requests/pending"`

HTTP Headers: `Authorization: Bearer some-jwt-token`

Request JSON: -NIL-

Success Response JSON (status code 200):

```
{
  "pending_batches":[
    {
      "batch_id":"56d236c9-17e9-4e57-a2cb-6bc51bbef486",
      "batch_data":[
        {
          "device_type":"printer",
          "id":"1852c75f-d2ce-4e53-aa5e-81be39de5ff8",
          "notes":"d1",
          "quantity":2,
          "status":"open" or "approved" or "disapproved"
        },
        {
          "device_type":"scanner",
          "id":"54dea9b3-00c4-4ca6-92db-4e2a9c014b14",
          "notes":"d2",
          "quantity":3,
          "status":"open" or "approved" or "disapproved"
        }
      ],
      "date_requested":"Sat, 19 May 2018 06:57:58 GMT",
      "post_office_id":"po5"
      "status":"open" or "part_approved" or "approved"
    }
  ]
}
```

#### **10.2.4 Pending Approvals Endpoint:**

For an authoritative user, displays the list of approvals which are awaiting approval from the user, along with devices and their quantities requested in the batch.

Request Type: `GET`

Endpoint: `BASE_URL + "/approvals/pending"`

HTTP Headers: `Authorization: Bearer some-jwt-token`

Request JSON: -NIL-

Success Response JSON(status code 200)

```
{
  "pending_approvals":[
    {
      "batch_data":[
        {
          "approved_on":null, (populated if status is approved)
        }
      ]
    }
  ]
}
```

```

        "approved_quantity":null, (populated if status is approved)
        "device_type":"printer",
        "id":"cca79a95-b99b-4800-bd38-7e73333e11f6",
        "notes":"d",
        "quantity":2,
        "status":"open" or
    }
],
"batch_id":"78f11923-43fd-4ee6-8a1f-2252d371b6dd",
"date_requested":"Sat, 19 May 2018 14:09:31 GMT",
"po_name":"head po",
"po_pincode":5,
"po_type":"head"
}
]
}

```

#### **10.2.5 Approve Request Endpoint:**

Allows an authoritative user to perform an approval task for a batch from the list of batches which are pending his approval. An approval task involves approving or disapproving or escalating select devices from the batch. Unless all devices have been either approved or disapproved, the batch remains partially approved.

If a request for a particular device is escalated, the approver is updated to reference the parent post office's authoritative user.

Request Type: POST

Endpoint: BASE\_URL + "/request/approve"

HTTP Headers: Authorization: Bearer some-jwt-token

Request JSON:

```

{
  "batch_id":"46ee1966-bf8a-4e6f-b7bd-02c9669fc0d4",
  "batch_data":[
    {
      "id":"69b17455-bd9e-43b3-8a0d-b837ab1d76c1",
      "status":"approved" or "disapproved" or "escalated",
      "device_type":"printer",
      "quantity_approved":2 (matters only when status is approved.
otherwise, it is ignored in the backend)
    }
  ]
}

```

Success Response JSON(status code 200)

```

{
  "success":true
}

```

### **10.2.6 Approved Requests Endpoint:**

Returns the set of device requests which were approved, and are awaiting installation in the current post office. This endpoint serves as an entry point for installing a device into a post office.

Request Type: GET

Endpoint: `BASE_URL + "/requests/approved"`

HTTP Headers: `Authorization: Bearer some-jwt-token`

Request JSON: -NIL-

Success Response JSON(status code 200)

```
{
  "requests_approved": [
    {
      "approved_on": "2018-05-23 06:55:56.845247",
      "batch_id": "46ee1966-bf8a-4e6f-b7bd-02c9669fc0d4",
      "date_requested": "Sat, 19 May 2018 14:11:58 GMT",
      "device_type": "printer",
      "number_approved": 2,
      "number_installed": 1,
      "number_requested": 3,
      "request_id": "69b17455-bd9e-43b3-8a0d-b837ab1d76c1",
      "requestor": "e1"
    },
    {
      "approved_on": "2018-05-23 15:14:55.642527",
      "batch_id": "ea5f4a5e-b85d-4aad-8840-a6ab78695f2d",
      "date_requested": "Sat, 19 May 2018 16:37:07 GMT",
      "device_type": "printer",
      "number_approved": 1,
      "number_installed": 0,
      "number_requested": 1,
      "request_id": "86adc7fa-f648-4cd5-bf95-542e0724318b",
      "requestor": "e1"
    }
  ]
}
```

### **10.2.7 Device Search Endpoint:**

This searches the inventory database for existing devices of the given device type, and having the serial number or barcode specified. It is used in order to decide if a particular device has been newly procured, or diverted during the install flow.

Input : barcode or serial\_number and device\_type which is being installed.

Output: Device details if barcode exists.

Request Type: POST

Endpoint: `BASE_URL + "/device/search"`

HTTP Headers: Authorization: Bearer some-jwt-token

Request JSON:

```
{
  "barcode_serial":"device1",
  "device_type":"printer"
}
```

Success Response JSON if barcode found (status code 200):

```
{
  "diverted":true,
  "device_details":{
    "cost":12000,
    "device_id":"ee6ecaf6-722d-48b1-ada1-1d923673a282",
    "manufacturer":"HP",
    "serial_number":"E123RsF",
    "barcode":"debkajsd-asd-a",
    "specification":"E4521 color printer",
    "warranty_in_months":12
  }
}
```

Success Response JSON if barcode not found (status code 200):

```
{
  "diverted":false
}
```

### **10.2.8 Install Devices Endpoint**

Installs the devices based on barcode search endpoint, and user inputs for non-diverted devices. When devices are installed, the `installed_quantity` field is updated in the database. If a new device is being installed, its details are sent in the request. If the device already exists and is being diverted, simply its `device_id` is sent in the request (NOTE: The `device_id` is obtained from the result of the device search endpoint, which is consumed as a precursor to this endpoint).

Request Type: POST

Endpoint: `BASE_URL + "/devices/install"`

HTTP Headers: Authorization: Bearer some-jwt-token

Request JSON:

```
{
  "request_id":"69b17455-bd9e-43b3-8a0d-b837ab1d76c1",
  "device_type":"printer",
  "device_list":[
```

```

    {
      "diverted":false,
      "barcode":"device1",
      "serial_number":"ASF145HSD",
      "specification":"E4521",
      "manufacturer":"HP",
      "cost":12000,
      "warranty_in_months":12
    },
    {
      "diverted":true,
      "device_id":"ee6ecaf6-722d-48b1-ada1-1d923673a282" (returned from
device search)
    }
  ]
}
Success Response JSON (`status code 200`):
{
  "success": true
}

```

### **10.2.9 View Inventory Endpoint:**

Returns inventory information of the current post office or the post offices under the care of the sysadmin, grouped by the device type. See response for clarity.

Request Type: GET

Endpoint: `BASE_URL + "/inventory/view"`

HTTP Headers: `Authorization: Bearer some-jwt-token`

Request JSON: -NIL-

Success Response JSON (`status code 200`):

```

{
  "inventory_info":[
    {
      "device_type":"printer",
      "count":2,
      "device_list":[
        {
          "barcode":"device1",
          "serial_number":"E124GSDF",
          "cost":12000,
          "date_of_installation":"2018-05-25 15:27:10.480095",
          "device_id":"ee6ecaf6-722d-48b1-ada1-1d923673a282",
          "manufacturer":"HP",
          "specification":"E4521 color printer",
          "warranty_in_months":12,
          "post_office_id": "po4",

```

```

        "diverted_po_list":["po1", "po2"],
        "diverted_date_list":["12/5/18", "12/3/18"]
    },
    {
        "barcode":"device3",
        "serial_number":"E4124GHDF",
        "cost":11000,
        "date_of_installation":"2018-05-25 19:24:55.559002",
        "device_id":"9216ffa6-478e-4c2c-9400-2d3bf7d881cd",
        "manufacturer":"HP",
        "specification":"E$523",
        "warranty_in_months":12,
        "post_office_id": "po4",
        "diverted_po_list":["po1","po2"],
        "diverted_date_list":["12/5/18", "12/3/18"]
    }
]
},
{
    "device_type":"scanner",
    "count":1,
    "device_list":[
        {
            "barcode":"device1",
            "serial_number":"12312BZAG",
            "cost":12000,
            "date_of_installation":"2018-05-26 13:29:33.348791",
            "device_id":"bdb3ede6-aa38-40ca-b505-2c961e921c04",
            "manufacturer":"HP",
            "specification":"A1234",
            "warranty_in_months":24,
            "post_office_id": "po4",
            "diverted_po_list":["po1", "po2"],
            "diverted_date_list":["12/5/18", "12/3/18"]
        }
    ]
}
]
}

```

#### IMPORTANT NOTE:

All the above endpoints, except for the login and register endpoints return the following failure responses:-

Failure Response JSON - Server error/JWT token malformed (status code 500):

```

{
    "message": "some error descriptions for debugging"
}

```

```
}  
Failure Response JSON - JWT unauthorized/invalid/not present (status code 401):  
{  
  "message": "some error descriptions for debugging"  
}
```

Also, in the documentation, the BASE\_URL refers to the home (or root) URL of the deployed server.

Eg. BASE\_URL = `http://<domain/ip-address>/bhim/api/v1`

Domain or IP address is the address of the machine or system on which the server is running.

## 11. PROGRESS TIMELINE

- June 2017 : Proposal was submitted to AICTE for approval
- July 2017 : Proposal was approved by AICTE
- August 2017 to September 2017 : Discussions on architecture and design of system based on inputs taken during hackathon and basic assumptions along with discussions with our mentors to guide us in architecting the basic groundwork for further development.
- October 2017 to December 2017 : Started partial development based on previous discussions. Technologies to be used were decided and necessary learning for the same was done. Majority of the basic backend server endpoints were coded.
- January 2018 to March 2018 : We were occupied with final semester undergraduate project and exams and hence very little work was done during this phase. By March 2018 based on RIFD bureau dated 01.03.2018 government sanctioned 3 lakh Rupees for post development activities.
- April 2018 to May 2018 : We had multiple rounds of discussions with point of contact from CEPT Chennai Chapter, Mr, Sakthivelu. Discussions included gathering of detailed domain knowledge and refining the software requirements. Development proceeded in parallel with these discussions, modifying and altering the features to suit the changes in requirements. A fully functional prototype was demoed during the end of May.
- June 2018 - Finalizing and testing features before code freeze; documentation of the software and handover to CEPT.



## 12. END USER MANUAL

### Purpose

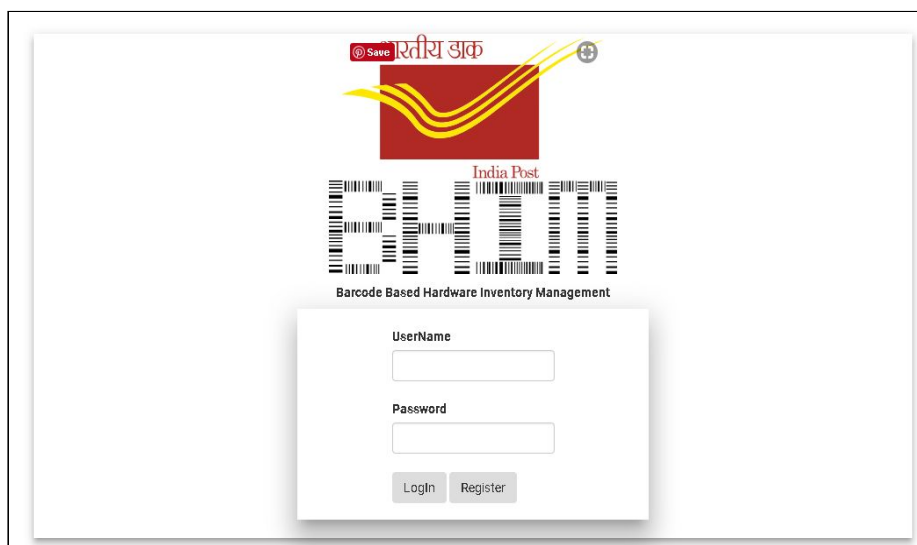
This guide provides comprehensive guidelines and step-by-step instructions for working with the **BHIM Hardware Inventory Management Application**

### About

BHIM is a hardware inventory management application for tracking and managing the inventory in post offices in India. The flow of the system is as follows

- A new user registers in the system and logs in to the system.
- The functionalities of the user varies based on his/her role in the post office.
- The roles of the user are : SysAdmin and Authoritative User.
- A SysAdmin raises request for new devices.
- The Authoritative user can approve, disapprove or escalate the request to higher authority.
- The SysAdmin can view the status of the pending requests.
- Once the request has been approved the sysAdmin can install the device into the system.
- The installed devices in his/her post office can be viewed .

### Register



1. Click on the register button  . A pop up opens.

2. Enter your employee id and email id.
3. Enter a password that contains at least one digit , symbol , lowercase and uppercase alphabet and is at least 8 characters long.
4. Rewrite the password in the confirm password input field. Make sure the passwords match.
5. Enter the post office id.
6. Specify your role from the dropdown displayed in the role field.

Register

7. Click on register button.
8. A pop up saying 'Registered successfully' appears if the submission is correct.

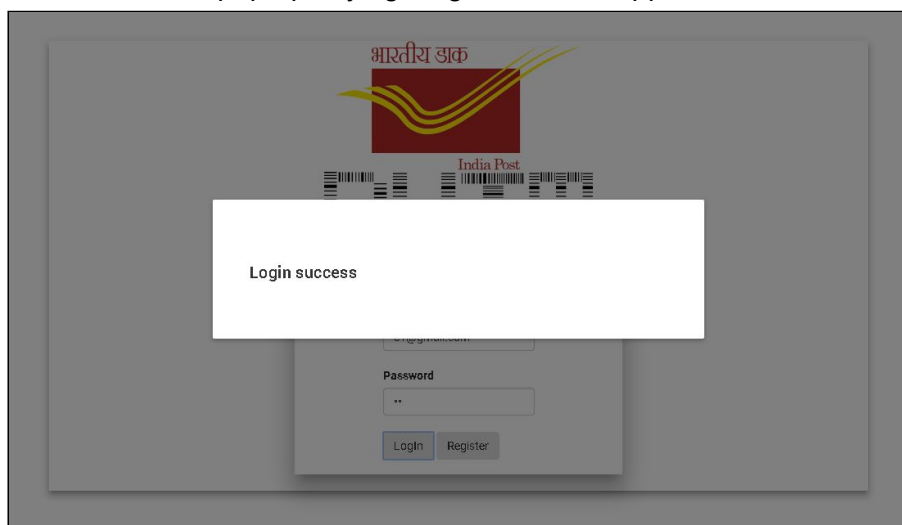
## Login

1. Enter your email id and password set during registration.

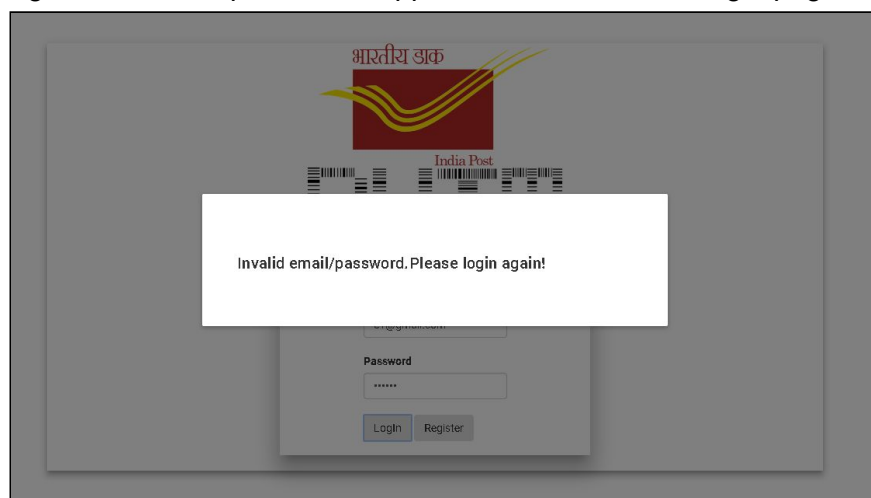


The screenshot shows the login interface for the India Post Barcode Based Hardware Inventory Management system. At the top, there is a logo with the text 'भारतीय डाक' (India Post) and 'India Post' below it, followed by a barcode. Below the barcode, the text 'Barcode Based Hardware Inventory Management' is displayed. The login form consists of two input fields: 'UserName' with the placeholder 'yourname@indiapost.gov.in' and 'Password' with a masked password '\*\*\*\*\*'. Below the password field are two buttons: 'Login' and 'Register'.

2. Click on login button.
3. If the details are valid a pop up saying 'Login success' appears

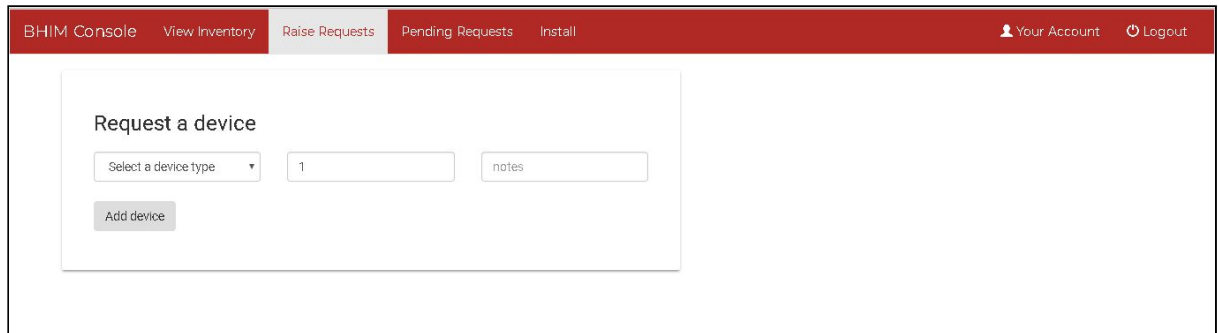


else pop up saying 'Invalid email/password ' appears and redirects to login page.



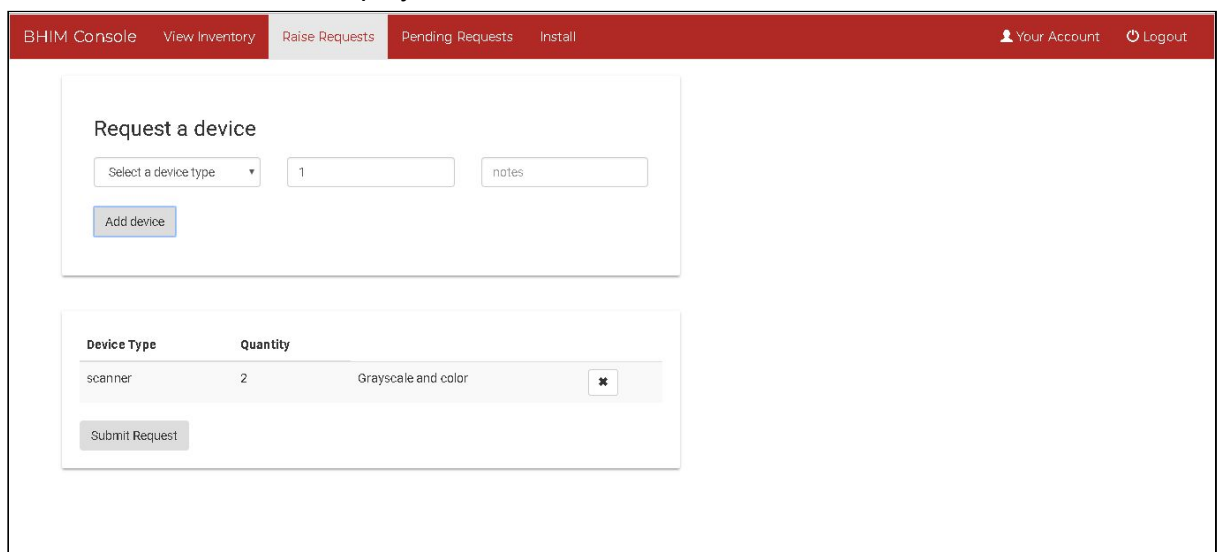
## Raise Request

1. Click on the raise requests tab in the header.




The screenshot shows the 'Request a device' form in the BHIM Console. The header bar is red with navigation tabs: 'BHIM Console', 'View Inventory', 'Raise Requests' (active), 'Pending Requests', and 'Install'. On the right of the header are links for 'Your Account' and 'Logout'. The form itself is titled 'Request a device' and contains a dropdown menu labeled 'Select a device type', a text input field with the number '1', and a text area labeled 'notes'. Below these fields is a button labeled 'Add device'.

2. Select the device type from the dropdown.
3. Select the quantity needed.
4. Enter any specifications(optional).
5. Click on add device.
6. The details selected are displayed in the card below.



This screenshot shows the same 'Request a device' form as the previous one, but with the 'Add device' button highlighted with a blue border. Below the form, a new card is visible, displaying the details of the added device. The card has a table with two columns: 'Device Type' and 'Quantity'. The table contains one row with 'scanner' and '2'. To the right of the table, there is a text input field with 'Grayscale and color' and a button with a plus sign. Below the table is a button labeled 'Submit Request'.

7. To delete the selected item click on the 'into'  symbol.
8. To add more items to the request go to step 1.

BHIM ConsoleView InventoryRaise RequestsPending RequestsInstall

Your AccountLogout

Request a device

Select a device type


1

notes

Add device

Device Type	Quantity		
scanner	2	Grayscale and color	✖
printer	1	Color	✖

Submit Request

9. Click on Submit button  to place the request.
10. A pop up saying 'Request Submitted successfully' appears and redirects to the same page.

BHIM ConsoleView InventoryRaise RequestsPending RequestsInstall

Your AccountLogout

Request a device

Select a device type

1

notes

Add device

Device Type	Quantity		
scanner	2	Grayscale and color	✖
printer	1	Color	✖

Submit Request

Request submitted successfully

## Pending Requests

1. Click on the pending requests tab in the header.

The screenshot shows the 'Pending Requests' tab selected in the header. The main content area displays a batch of requests. At the top, it shows 'Batch Id : 1', 'Batch Status' with an 'open' button, and 'Requested Date : Sun, 03 Jun 2018 09:43:28 GMT'. Below this is a table with columns: Request Id, Item Type, Quantity, Notes, and Request Status. The table contains two rows: one for a scanner (Request Id 1, Quantity 2, Notes 'Grayscale and color') and one for a printer (Request Id 2, Quantity 1, Notes 'Color'). Each row has an 'open' button in the Request Status column.

Request Id	Item Type	Quantity	Notes	Request Status
1	scanner	2	Grayscale and color	open
2	printer	1	Color	open

2. A list of pending requests are displayed.
3. Each pending request contains information about

- Batch Id
- Batch Status
- Date of request

Each batch contains information about

- Request Id
- Device Type
- Quantity
- Notes
- Request Status

4. The status can be of the following types:

- Open - The request placed has not been initiated
- Partial - Only few of the requests in a particular batch has been approved by the higher authority
- Approved - The batch is approved by the higher authority
- Disapproved - The batch is disapproved by the higher authority

## Pending Approvals

1. Click on the pending approvals tab in the header.

The screenshot shows the 'Pending Approvals' tab selected in the header. The main content area displays a table with columns: Batch Id, Date Requested, Post office Name, Post office pincode, and Post office type. The table contains one row: Batch Id 1, Date Requested 'Sun, 03 Jun 2018 09:43:28 GMT', Post office Name 'head po', Post office pincode '5', and Post office type 'head'. There is a 'View Details' button next to the row.

Batch Id	Date Requested	Post office Name	Post office pincode	Post office type
1	Sun, 03 Jun 2018 09:43:28 GMT	head po	5	head

2. A list of pending approvals to be made by the current user is displayed.
  - Batch Id
  - Date Requested
  - Post office Name
  - Post office Pincode
  - Post office Type
3. Each pending approval contains a View Details button
4. Click on View Details button
5. A pop up appears showing the following details of the pending approval
  - Request Id
  - Device Type
  - Notes
  - Status
  - Last approved on
  - Last approved Quantity
  - Number requested
  - Number approved

View Details

Batch ID : 1

Request Id	DeviceType	Notes	Status	Last approved on	Last approved quantity	Number Requested	Number approved	Mark to approve	Mark to disapprove	Mark to escalate
1	printer	Color	open			1	0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	scanner	Grayscale and color	open			2	0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Submit

6. Click on either one of the three checkboxes
  - Mark to approve - To approve the particular device
  - Mark to disapprove - To disapprove the particular device
  - Mark to escalate - To escalate the requests to the higher authority

BHIM Console View Inventory Raise Requests Pending Approvals Install Your Account Logout

### Pending Approvals

Batch ID : 1

Request Id	DeviceType	Notes	Status	Last approved on	Last approved quantity	Number Requested	Number approved	Mark to approve	Mark to disapprove	Mark to escalate
1	printer	Color	open			1	0	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
2	scanner	Grayscale and color	open			2	2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Submit

- Click on the submit button to place the request

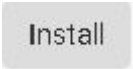
## Install


- Click on the Install tab in the header

BHIM Console View Inventory Raise Requests Pending Requests Install Your Account Logout

Request Id	Requested Date	Device type	Number requested	Number approved	Number disapproved	Number installed	Requestor	
1	Mon, 28 May 2018 12:58:35 GMT	printer	2	1	1	0	e1	Install
2	Mon, 28 May 2018 13:03:57 GMT	printer	5	2	3	0	e1	Install

- The following details are displayed in the card
  - Request Id
  - Requested Date - The date the request was placed
  - Device Type - The type of the device
  - Number requested - The quantity entered when request was submitted
  - Number approved - The quantity approved by the higher authority
  - Number disapproved - The quantity disapproved by the higher authority
  - Number installed - The quantity installed in the system
  - Requestor - The employee id of the person who placed the request

- Click the button Install  to install a device in the system

- Enter the serial number of the device and click on search button 



BHIM Console View Inventory Raise Requests Pending Requests **Install** Your Account Logout

Request Id	Requested Date	Device Type
1	Mon, 28 May 2018 12:58:35 GMT	printer
2	Mon, 28 May 2018 13:03:57 GMT	printer

RequestID : a71df9bb-f8a1-498c-a1cb-2c8b4081c550  
Device Type : printer

device1

Manufacturer: HP Specification: E4123 Cost: 1200 Warranty: 12

Add device

5. If the device has been diverted from another post office the following details of the device are displayed
  - Manufacturer
  - Specification
  - Cost
  - Warranty
6. If the device is to be installed for the first time, enter the above details in the input fields

BHIM Console View Inventory Raise Requests Pending Requests **Install** Your Account Logout

Request Id	Requested Date	Device Type
1	Mon, 28 May 2018 12:58:35 GMT	printer
2	Mon, 28 May 2018 13:03:57 GMT	printer

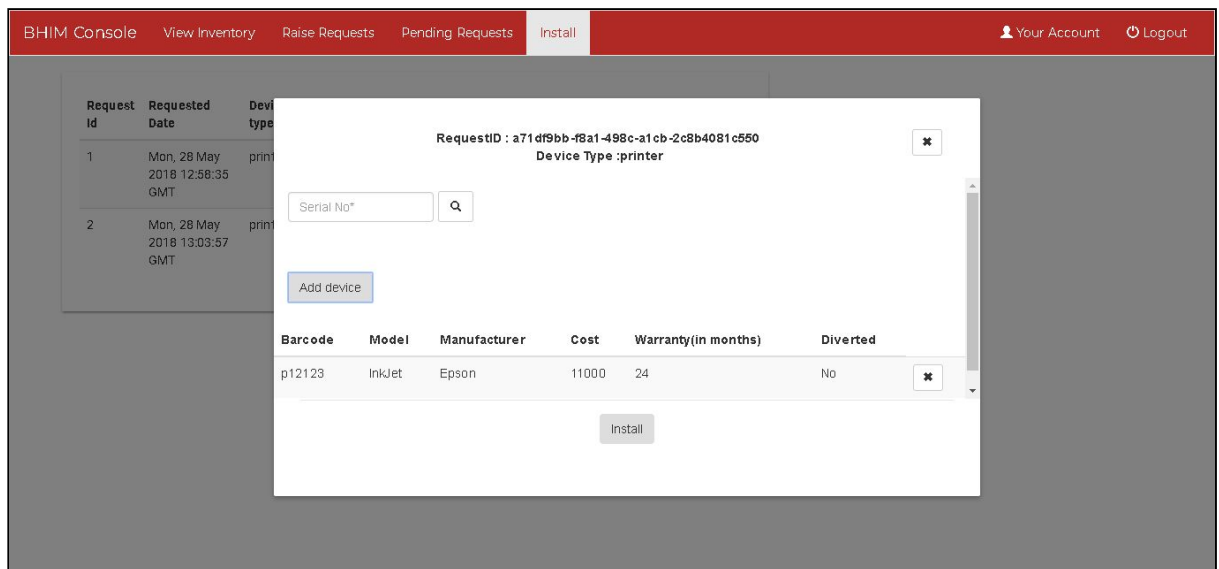
RequestID : a71df9bb-f8a1-498c-a1cb-2c8b4081c550  
Device Type : printer

p12123

Manufacturer: Epson Specification: InkJet Cost: 11000 Warranty: 24

Add device

7. Click on add device button
8. The added device details are displayed in the table below



9. Click on 'into' symbol if you want to delete the added device

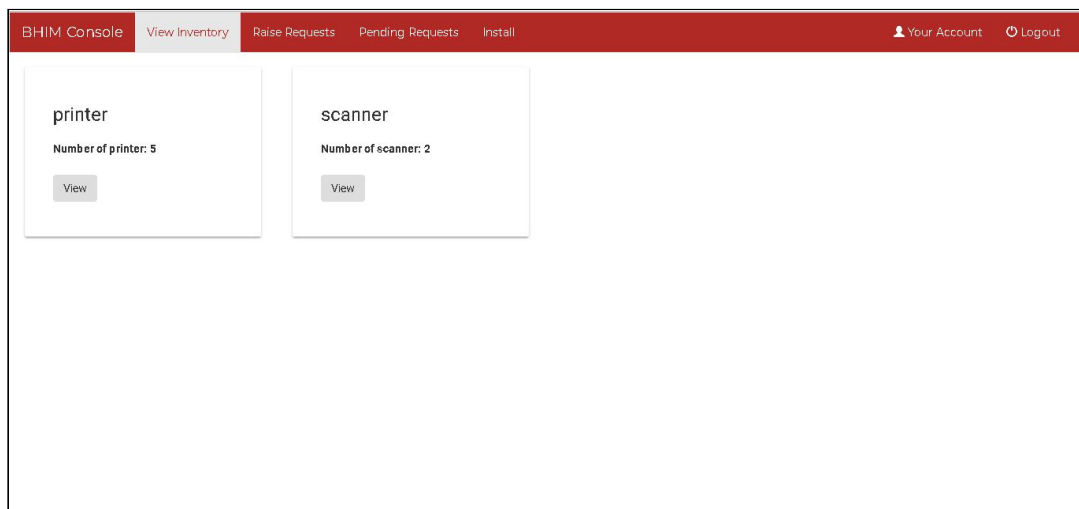


10. Click on Install button to place the request.

Install

## View Inventory

1. Click on the view inventory tab in the header



2. The list of devices and its count is displayed

3. Click on View button

View

to view in detail

4. The following details are displayed

- Id
- Barcode
- Cost
- Manufacturer
- Specifications
- Warranty
- Date of Installation

- Device History

BHIM Console

View Inventory

Raise Requests

Pending Requests

Install

Your Account

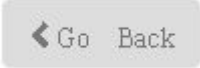
Logout

Go

Back

printer

Id	Barcode	Cost	Manufacturer	Specifications	Warranty	Date of Installation	Device History
1fee9de4-7a35-4cea-a220-2580afea232d	device1	1200	HP	2018-05-28 13:34:33.230681	E4123	12	
bd777591-c082-44bd-ba2c-9d0c643b8d2a	p12123	11000	Epson	2018-06-03 09:54:39.285978	InkJet	24	
9994bc1b-b93e-4bed-bc71-986938b1ae20	p121234	14000	Hp	2018-06-03 09:58:22.883559	Inkjet	24	
c3210dd0-95d6-4b6a-854b-28619de76d64	ps1213	15000	Hp	2018-06-03 09:58:22.886334	Printer & Scanner	24	

5. Click on Go back  to go to the previous page.
6. Repeat from step 2 to view details of all device types.

BHIM Console

View Inventory

Raise Requests

Pending Requests

Install

Your Account

Logout

< Go

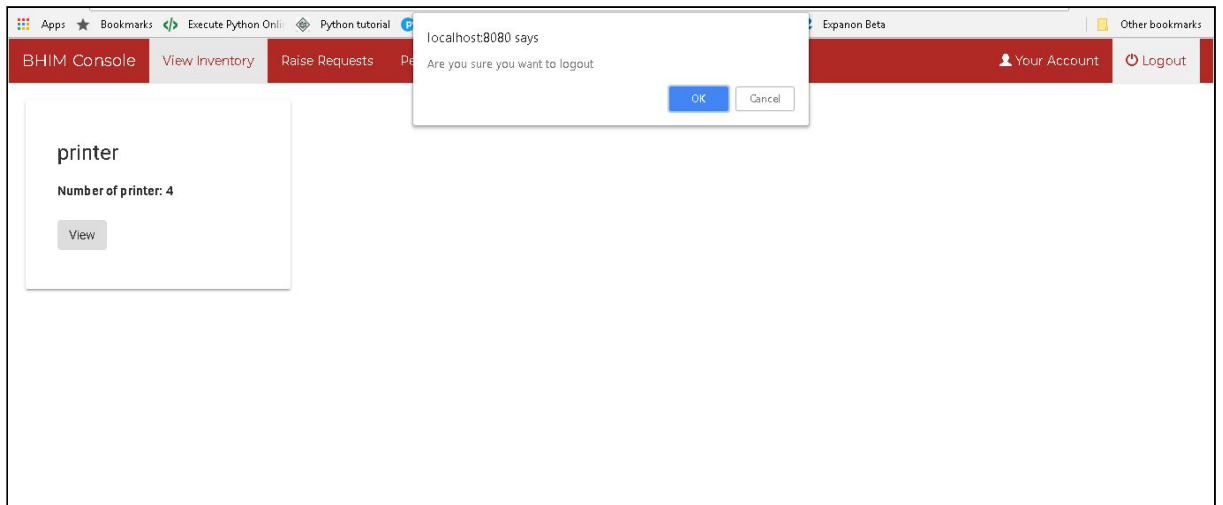
Back

scanner

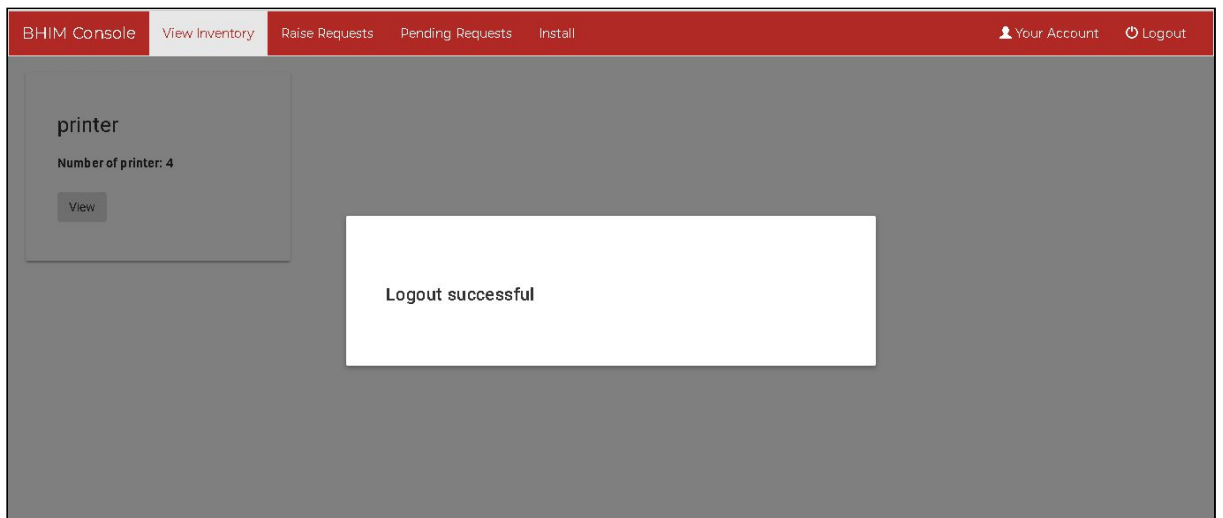
Id	Barcode	Cost	Manufacturer	Specifications	Warranty	Date of Installation	Device History
d677c701-ca05-46d5-bfa5-6958258ffd4a	s1001	11000	Hp	2018-06-03 18:05:05.363081	Grayscale	24	
bd6a7c24-2979-4c28-9b14-d2b408189b7a	s1002	11000	Hp	2018-06-03 18:05:05.365648	Color	24	

## Logout

1. Click on the logout button.
2. An alert appears .



3. Click on Ok to logout successfully.



4. Click on cancel to abort the action