MAGNET MAIL READER

Version 1.0

User Manual

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S. No** | **Document Version** | **Updated By** | **Reviewed By** | **Release Date** |
| 1 | 1.0.1 | Vishnu Moorthy K | Hariharan | 21-May-16 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

***Magnet Mail Reader***

**Description:**

* The Magnet Mail Reader is an Executable Jar, Which access the given Email Account’s Inbox and reads the all Incoming Emails.
* In this Jar module, we are accessing the multiple email accounts, each email account executes as a Thread, Which are taken from respective table from database.
* When the New Customer is created, then the New Email Property is created using the Customer’s Account Id with default mail domain configuration (SMTP Host, Port, Email, Mail password and Protocol). These mail domain configuration is getting from respective table
* Once the corresponding email account is created for the corresponding Email configuration, then the email property will be activated.
* In this Jar module, access the each email account and gets an Incoming Emails from its Inbox, which are to be file attached. It ignores the non-attachment Emails.
* It parses the information like sender Email Id, Email Sent date, Email Subject and Email Attachment.

After it extract the attachment to parse the data, it opens the file and Extract the Candidate Information like Name, DOB, Gender, Age, Degree, Salary Information, Experience & etc.., and save these information to Database.

**Email Properties:**

In this mail reader executable jar needs the below information to access the Corresponding Email Account,

1. Email Id
2. Email Password
3. Mail Host
4. Mail Port
5. Mail Protocol (IMAP/POP3)

**Process Mail Attachments:**

|  |  |  |  |
| --- | --- | --- | --- |
| **File Type** | **File Extraction** | **Data Parsing** | **Saved to Database** |
| .pdf | Yes | Yes | Yes |
| .doc | Yes | Yes | Yes |
| .docx | Yes | Yes | Yes |
| .rtf | Yes | Yes | Yes |
| .html | Yes | Yes | Yes |
| .htm | Yes | Yes | Yes |
| .xls | No | No | Yes |
| .xlsx | No | No | Yes |
| .odt | No | No | Yes |
| .jpeg/.jpg | No | No | Yes |
| .png | No | No | Yes |
| .bmb | No | No | Yes |
| .zip/.rar | No | No | Yes |

## Required Files & Procedures:

## Create Base Folder under /home/ec2-user

## The Base Folder must contains the following files/folders

## “properties” folder – which contains what are the property files need for the corresponding jar (Database property, Log 4j property)

## “lib” folder – which contains the what are the dependency jars needed for the core

## “Executable Jar” – which performs the corresponding functionality using crontab

**Process Flow Diagrams:**

Attachment Found

Attachment Not Found

Email Id not exists

Email Id exists

**\*Extract the attachment(s) to parse data**

Check if the sender email id is already exist or not for the corresponding customers

Check if the incoming message has attachment(s) or not

Check incoming message sent date is greater than last incoming email’s sent date from database

Get the list of Incoming Messages from Inbox

Iterate the Incoming Messages

Iterate the Email Property

Access the each Email Account Using Thread

Create the Session to authenticate the corresponding Email account

Get a Store object that implements the specified protocol

Get Connection from Store for accessing the Mail Folders (Inbox)

Get the last incoming email sent date from Incoming email table, which are already saved to Database

Database

Get the Activated Email Property List

Check next incoming message

**Process Flow Diagram - \*Extract the attachment(s) to parse data**

Save Resume Object

\*Create Resume Data Object

Create & Save the Resume Transaction Object

\*Mapping Resume with corresponding customer’s Job Positions

Database

Get Last Resume URN and create the new upcoming Resume URN

Create Incoming Email Object with Attachments

Get the mail attachment(s) & save it to Repository. Save incoming Email Object with attachment Repository location to Database

Create Resume Object

Remove the auto mailer content from candidate email id.

Save Resume Data Object

**Process Flow Diagram - \*Create Resume Data Object**

Save Resume Data Object

Yes

Yes

No

Extract the Resume Attachment and get it’s file content as String[], based on the corresponding file type.

Returns the empty Resume Data Object

Extract the corresponding File based on its extension & Get the file content as String[]

Check if the String[] is null or empty

\*Parsing the Resume Information like name, DOB, age & etc…, from attachments

Get the Resume Data Object

Check the file extension to extract. If the file extension is .doc/.docx/.rtf/.pdf/.htm/.thml only

No

Database

**Process Flow Diagram - \*Mapping Resume with corresponding customer’s Job Positions**

Iterate Matched Job Positions List

Create the Job Position Resume Mapping Object & Map the corresponding Resume with Others Job Position

Check next job positions

Yes

Yes

Check if the corresponding mail subject contains the job position’s job title

Compare the job title & mail subject word by word. For ex. Check if Logistics contains Logistic or not

Get the Others Job Position Object

Not Equals

Iterate Job Positions List

Check if the job position’s job title equals “Others” or not

Get the Job Positions list for the corresponding Customer

Database

Equals

Add the corresponding job positions to Matched Job Position List

No

No

Check if the Matched Job Positions List is empty or not

Empty

Not Empty

Create the Job Position Resume Mapping Object & Map the corresponding Resume with Matched Job Position

Save Job Position Resume Mapping Object to Database

**\*Parsing the Resume Information**

**To Parse Name – Process Flow Diagram:**

Yes

No

Get Name from Sender name via Mail Id (“From” Field)

If Name contains the word like Auto mailer

Get Name from Mail Subject. For ex, “Application for Customer Service Specialist from SHIEK RAHIM, FAZILA”. The name should be “SHIEK RAHIM, FAZILA”

Set Name to Resume Data Object

**To Parse Email ID – Process Flow Diagram:**

Yes

No

Gets Email Id from Resume using Regex Pattern Matcher

Check If Email Id is empty or null

Set Email Id to Resume Data Object

Set Highest Qualification as empty to Resume Data Object

**To Parse Date of Birth - Process Flow Diagram:**

Date

Not Date

Yes

No

Get DOB from Resume using keyword search

Check If DOB is empty or null

Check if the DOB is date or anything else

Set DOB as empty to Resume Data Object

Convert the date format into “DD/MM/YYYY”

Set DOB to Resume Data Object

**To Parse Year of Birth - Process Flow Diagram:**

Yes

No

Check If DOB is empty or null

Get the Year from DOB

Set Year of Birth as empty to Resume Data Object

Set Year of Birth to Resume Data Object

**To Parse Age - Process Flow Diagram:**

Yes

No

Check If DOB is empty or null

Get the Year from DOB

Set Year of Birth as empty to Resume Data Object

Set Year of Birth to Resume Data Object

**To Parse Nationality - Process Flow Diagram:**

No

Yes

Get Nationality from Resume using keyword search

Check If Nationality is empty or null

Set Nationality to Resume Data Object

Set Nationality as empty to Resume Data Object

**To Parse Race - Process Flow Diagram:**

Yes

No

Get Race from Resume using keyword search

Check If Race is empty or null

Set Race to Resume Data Object

Set Race as empty to Resume Data Object

**To Parse Marital Status - Process Flow Diagram:**

No

Yes

Yes

Get Marital Status from Resume using keyword search

Check If Marital Status is empty or null

Set Marital Status to Resume Data Object

Check if the Resume contains “single, married, widow”

Set Marital Status as empty to Resume Data Object

No

**To Parse Gender - Process Flow Diagram:**

Yes

No

Get Gender from Resume using keyword search

Check If Gender is empty or null

Set Gender to Resume Data Object

Check if the Resume contains “female, male”

Set Gender as empty to Resume Data Object

Yes

No

**To Parse Highest Qualification - Process Flow Diagram:**

Yes

No

Get Highest Qualification from Resume using keyword search

Check If Highest Qualification is empty or null

Set Highest Qualification to Resume Data Object

Set Highest Qualification as empty to Resume Data Object

**To Parse Phone/Mobile No - Process Flow Diagram:**

No

Yes

Yes

No

Gets Phone No from Resume using Regex Pattern Matcher

Check If Phone No is empty or null

Check if it is numeric or not

Get Phone No from Resume using keyword search

Object

Set Phone & Mobile No as empty to Resume Data Object

Check if it length is 8 and starts with 6

Set Phone No to Resume Data Object

Set Mobile No to Resume Data Object

No

Yes

**To Parse Expected Salary - Process Flow Diagram:**

Yes

No

No

Yes

Get Expected Salary from Resume using keyword search

Check If Expected Salary is empty or null

Set Expected Salary Min & Max as empty to Resume Data Object

Check if it contains the numeric with price format

Remove the alphabets and split by special characters

Get the split words length

If length is one

If length is two

Set Expected Salary Min & Max, both are same value to Resume Data Object

Set Expected Salary Min & Max to Resume Data Object

**To Parse Current Salary - Process Flow Diagram:**

Yes

No

Get Current Salary from Resume using keyword search

Check If Current Salary is empty or null

Set Current Salary to Resume Data Object

Set Current Salary as empty to Resume Data Object

**To Parse Address - Process Flow Diagram:**

No

Yes

Get Address from Resume using keyword search

Check If Address is empty or null

Set Address to Resume Data Object

Set Address as empty to Resume Data Object

**To Parse Postal Code - Process Flow Diagram:**

Yes

No

Yes

No

Yes

Check If Address is empty or null

Get Postal Code from Resume using keyword search

Check if Address contains the postal code like 6 digit numeric value

Check if is numeric and its length is 6

Set Postal Code to Resume Data Object

Set Postal Code as empty to Resume Data Object

No

**To Parse Availability - Process Flow Diagram:**

Yes

Get Availability from Resume using keyword search

Check If Availability is empty or null

No

Set Availability to Resume Data Object

Set Availability as empty to Resume Data Object