**Table of Contents**

Table of Contents 1

1. Project Introduction 2

1.1 Purpose 2

1.2 Document Conventions 2

1.3 Intended Audience and Reading Suggestions 2

1.4 Product Scope 2

1.5 References 2

2. Introduction to Tools 4

2.1 Existing System 4

2.2 Proposed System 4

2.3 Tools and Technology overview 4

3. Overall Description 4

3.1 Product Perspective 4

3.2 Product Functions 4

3.3 User Classes and Characteristics 4

3.4 Operating Environment 4

3.5 Assumptions and Dependencies 4

4. External Interface Requirements 5

4.1 User Interfaces 5

4.2 Hardware Interfaces 6

4.3 Software Interfaces 6

4.4 Communications Interfaces 6

5. System Features 7

5.1 System Feature 1 to 8 7

6. Other Nonfunctional Requirements 8

6.1 Performance Requirements 8

6.2 Safety Requirements 8

6.3 Security Requirements 8

6.4 Software Quality Attributes 8

7. Future Enhancement 8

Appendix A:Glossary 9

Appendix B: Design Documentation 9

**Industry Details**





* Promact is an excellence driven company with a passion for technology where people love what they do.
* It focuses on delivering values, creating impact & producing exciting experiences.
* Fun workplace, technology leader, target excellence - these three pillars of our vision are the key foundation of Promact.
* Promact strive and encourage the culture of start-ups. And not only provide development services but also share the personal start-up experience which can help people streamline and grow with your start-up.
* At Promact we provide all technology solutions including Web Application, Mobile Applications & Devops.
* We constantly invest in new and upcoming technologies to provide our clients as well as to employees, an edge in the ever so fast-moving market.
* Uses latest technologies such as Android, iOS, Ionic, Phonegap fort mobile application development, Angular, NodeJS, ASP.NET, PHP, Python for delivering intuitive web applications.
* Also uses Docker, Amazon AWS, Microsoft Azure, Apache for Devops.

**CHAPTER:1**

**Introduction to Tools**

* 1. **Existing System**
* It is possible to have multiple users or task tracker in particular job. To maintain flow of entities, inspection and evidence, everyone will have to manage it manually by maintaining physical documentation of each and every process.
* Most of the time it would be very difficult to do paper work if tasks or documents are too large with respect to code, time or complexity.
* One will require the single platform which helps the users to perform and manage complex task or evidence without doing any manual work.
* To achieve all these functionalities, this system is being implemented with the help of different tools and technologies.
  1. **Implementation Strategy**
* After successful registration and verification, user will be able to login into system and can use functionalities provided by the system.
* First *UX* designer will create basic mockups for the user interface using different designing tools such as photoshop, Adobe Illustrator etc.
* With the help of design, front end will design using *HTML*, *CSS*, Bootstrap, JavaScript, Angular etc.
* Asp.net Core will be used to create Web services and *APIs* and other backend functionalities. PostgreSQL and *AWS* services is also need to manage resources, data and storage.
* The initial phase of implementation is to learn and go through basic training of the technologies needed for implementation of assigned work.
* Second phase is requirements gathering with questions and assumptions and basic mockups of the task.
* Third phase would be implementation of interactive user interface and binding data with backend functionality and other modules.
  1. **Tools and Technology Overview**
     1. **Angular**
* Angular is a platform that makes it easy to build applications with the web. It combines declarative templates, dependency injection, inbuilt backend integration, end to end tooling, and integrated best practices to solve development challenges
* Angular apps load quickly with the new Component Router, which delivers automatic code-splitting so users only load code required to render the view they request.
* Command line tools: start building fast, add components and tests, then instantly deploy.
* Quickly create UI views with simple and powerful template syntax.
* With Karma for unit tests, you can know if you've broken things every time you save. And Protractor makes your scenario tests run faster and in a stable manner.
  + 1. **ASP.net Core**
* ASP.NET Core is a cross-platform, high-performance, [open-source](https://github.com/aspnet/home) framework for building modern, cloud-based, Internet-connected applications.

With ASP.NET Core, you can:

* Build web apps and services, [IoT](https://www.microsoft.com/internet-of-things/) apps, and mobile backends.
* Use your favorite development tools on Windows, macOS, and Linux.
* Deploy to the cloud or on-premises.
* Run on [.NET Core or .NET Framework](https://docs.microsoft.com/en-us/dotnet/articles/standard/choosing-core-framework-server).
* It allows you to easily differentiate parts of your code for their behavior in development, staging, production, etc.
  + 1. **PostgreSQL**
* PostgreSQL is a powerful, open source relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads.
* PostgreSQL requires very minimum maintained efforts because of its stability.  Therefore, if you develop applications based on PostgreSQL, the total cost of ownership is low in comparison with other database management systems.
* PostgreSQL was then also designed to be portable so that it could run on various platforms such as Mac OS X, Solaris, and Windows.

**CHAPTER:1**

**System Introduction**

* This Software Requirements Specification (*SRS*) provides the overall general and detailed software requirements for the Inspection and evidence control system.
* This section provides an introduction to the INSPECTION AND EVIDENCE CONTROL SYSTEM project and the purpose, scope, and other details require fulfilling customers need.
  1. **Purpose**
* Inspection and evidence system specializing in management and controlling the proper flow of documents and task from point of origin to point of distribution
* Main purpose of this system is to eliminate the paper work to handle all documents. Management of proper service flow with less time by automating manual processes with the help of different technologies.
* The proposed system will feature different modules that ensures the proper flow of documents and related task to all those who are concerned with a job.
* A system that would be able to manage users with authentication, their documents, verification of team members, forms for task, flow of entities from source to destination, assets that are required to perform different task etc.
  1. **Document Conventions**

The following document conventions are followed in preparing this *SRS*.

* All key-words related to the academics are formatted in italics.
* SRS-Software Requirement Specification
* API-Application Programming Interface
* DB-Database
* HTML-Hypertext Markup Language
* CSS-Cascading Style Sheet
* API-Application Programming Interface
* AWS- Amazon Web Services
  1. **Intended Audience and Reading Suggestions**
* This documents is prepared for the software development team for their use in analyzing the requirements
* The product is intended towards business corporations & individuals.
* The Product will be useful for business corporations in order to ensure proper flow of each job via enabling inspection and evidence assurance.
  1. **Product Scope**
* This system will allow users to use all the modules provided in it after successful login followed by registration.
* After proper verification of details such as username/email and password, functionality would be accessible to the user.
* As per the requirement user can create forms for job and assets, view and update profile, upload and download data sheets, create Jobs with required fields. view job with different format, add team members who are responsible to get the task done, generate report, add attachments (images, files, audio, video) etc.
  1. **References**
* <https://www.mindtools.com/pages/article/newPPM_77.htm>
* <https://www.glowtouch.com/business-analysis-process-flow>
* [https://material.angular.io /](https://material.angular.io/)
* <https://www.tutorialspoint.com/angular_material/>

**CHAPTER:3**

**Overall Description**

**3.1 Product Perspective**

* This is the web based open source system implementing in different technologies.
* The web pages are present to provide the UI to the customers.
* Communication between user and server is provided thorough HHTP protocols.
* Database is used to store data and serveices provided by AWS are also used in this product.
* APIs are created for different module’s functionality.

Example: Job module requires different calls given below.

Accept Return Job request

Add Job

Close Job

Get Job by Id

Get Jobs

Get only updated

To Reopen Closed Job

Update Job

**3.2 Product Functions**

First user will have to do sign up to use the system. After proper verification of user’s details such as email verification, contact number verification, one will be able to use the proposed system.

Different Module wise functionality is given below:

* **DASHBOARD**
  + It is provided to the users to view summary like structure of activities.
  + This sub module will display following details.

Number of jobs created by user.

Total added location

Number of reports generated by user

Space/Storage occupy by the documents stored by the user.

Information about jobs such as name, author, status (Pending, in progress, Closed).

* **PROFILE**
  + User information will be display in this sub module.
  + It includes following details.

User Name

Email

Phone

Profile photo

Delete profile, Edit profile functionality

Change password functionality.

* **USER DOCUMENTS**
* If any documents are uploaded by user, then the details such as job name in which documents are uploaded, description, total size, last modified by, and updated time and date would be display in this sub module.
* **JOB**
  + Whenever this type of system will be introduced to user, it is necessary to have one most important feature that is able to manage or prepare single Job/Task.
  + In Job tab, front end part will display List of Jobs with below necessary details.

Different working functionalities are given below.

**Create job, Edit job, Delete job, Share job, View job**

* After implementation of the solution, user will be able to create Job with job details (Title, Description, location, overview, and conclusion).
* In overview and conclusion tab, user will be able to use existing forms or can create form as per the requirements.
* System will provide dynamic form building functionality to satisfy their needs. Different attachment options would be there to attach images or documents in different format.
* Create job functionality will also allow to add task that has to be done, agents who will manage task, assets require to complete job etc.
* User will be able to share job with others who are responsible to actively participate in job.

**Report Generation.**

* After job is created successfully, user can generate report as per his format as there would be different formatting options available mentioned below.
* Basic options:

Print quality (Grayscale, Colorful)

Image options (All images, Select Images)

Layout options (Full, Compact)

Thumbnail size (Normal, 2x)

Additional details (Full size media, print attachments, Download links)

* Advanced options:

Choose sections (Check In, Check List, Images, Audio, Video, Job, Attachment, Email details, Conclusion history)

Grid line options (Horizontal, Vertical)

Table header options (Left-to-right, Top-to-bottom)

* Mail will be sent to user by which he can easily download report.
* **DATA IMPORT**
* This sub module will provide functionality to import or export data sheets (CSV, XLS, XLSX) having important data require to add or download for the task that has to be done.
* **INSTACOUNT**
* As the name suggest, this is the part of project in which instant count would be done.
* During the task, it is not feasible to count entity/objects manually. So, by uploading an image system would be able to give total count of the object such as pipes of different shapes.
* **FORMS**
* Different type of forms would be created here.
* Forms created in this sub module could be use into other Job Module also.
* It will allow creating three basic types of forms given below.

General Form

Overview Form

Conclusion Form.

* **ASSETS**
  + - List of assets require to complete the task would be added from here.
    - It will allow creating forms to add assets by dynamic form builder functionality.
    - These assets could be import to the Job module while creating job.
* **UNIVERSAL SEARCH**
* This system enables functionality to go through all the details in the system in a very easy to use manner.
* As this project contains too many modules and functionality, it would be very crucial to display search result in single tab.
* This may create confusion to user to identify which result is pointing to which module.
* So, the task would be to create the search functionality that is able to display result with different tabs (all, job, data entered, mail conversations, assets, images, audio, video) and the link will redirect to the module in which search result is found.

**3.3 User Classes and Characteristics**

* The users are assumed to have basic knowledge to operate the system and must be familiar to work over the internet via web browser.
* Users shall be familiar with document & multimedia files to interact with the system efficiently.
* Users are supposed to know English language.
* User should be able to do the following functions:

Create Account, Form, New job, Assets,

Access Job, Profile information, Documents

Update Job, Profile

Add attachments (image, audio, video,), Team members

Generate report

Close job, Reopen job

**3.4 Operating Environment**

1. Hardware Requirement:

* PC/Laptop/Smartphone with internet access.
* Memory:2GB
* Processor: Intel x86/ AMD/ ARM based processor @ 1GHz
* RAM: minimum 2.0 GB
* System type: 32/64-bit operating system.

1. Software Requirement:
   * Web browser for PC/ Laptop.
   * Android/ iOS for native mobile application support.

**3.5 Assumptions and Dependencies**

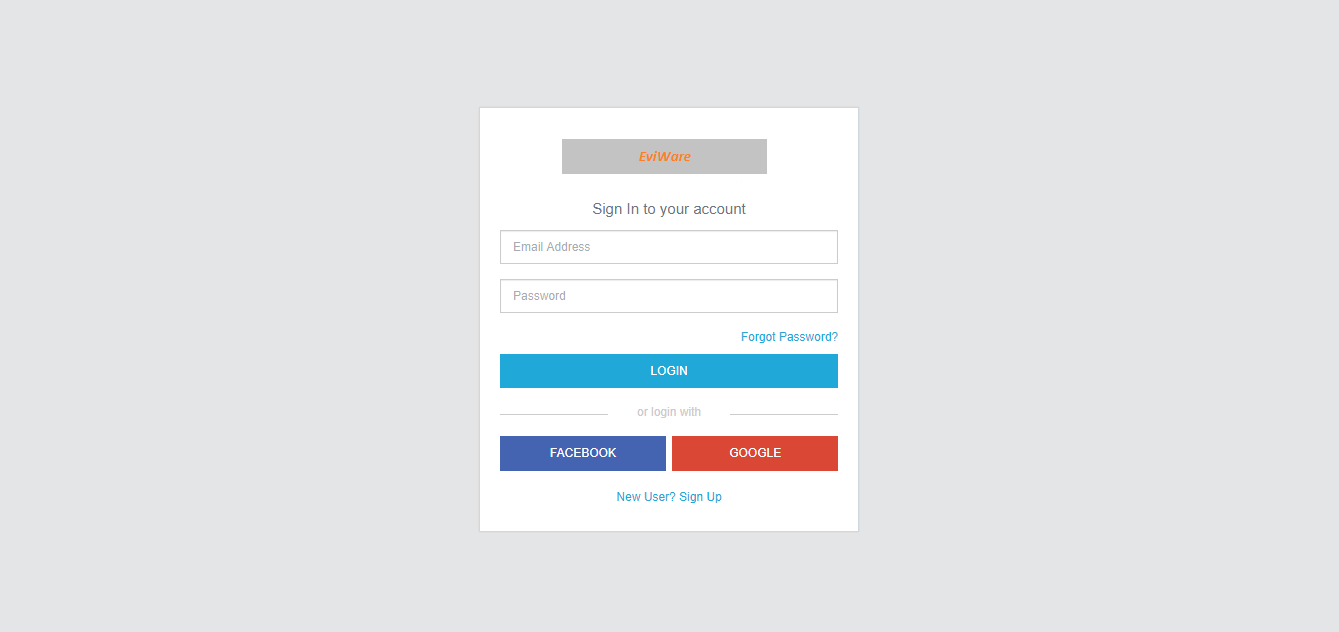
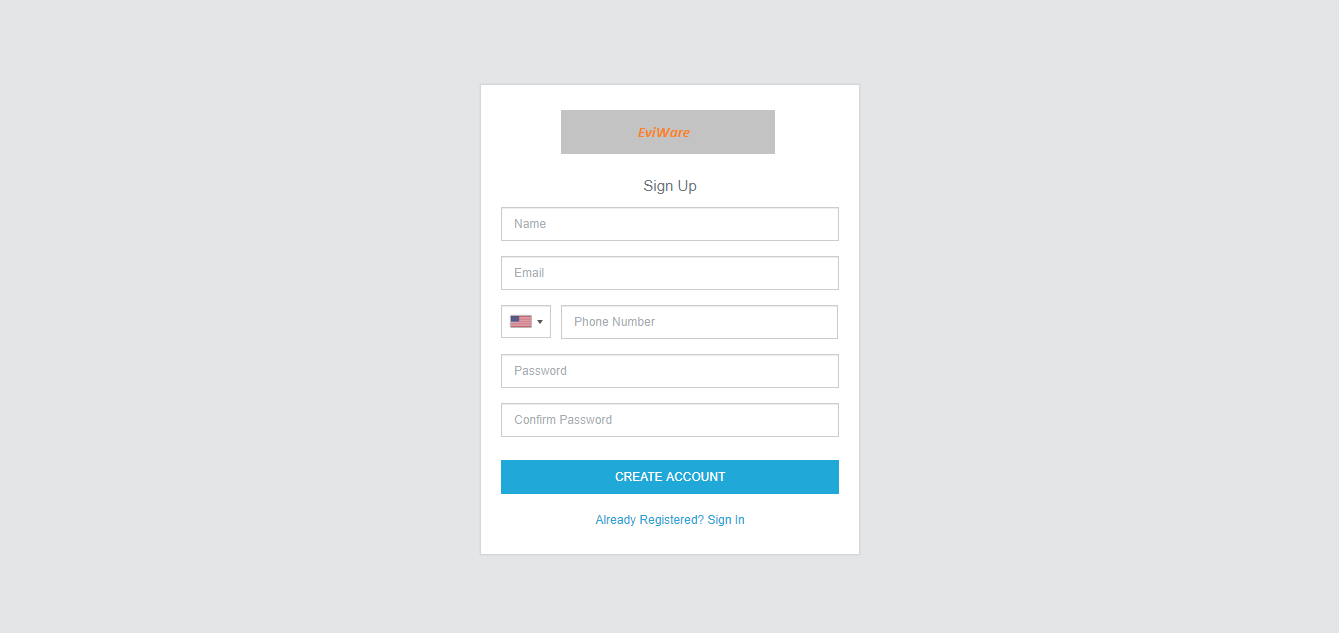
* Dashboard interface is available to users to get an overview.
* Login & registration modules are already created.
* Instacount, Forms & Assets modules are available to use in Job module.
* Administrator interface & functionalities are already created.

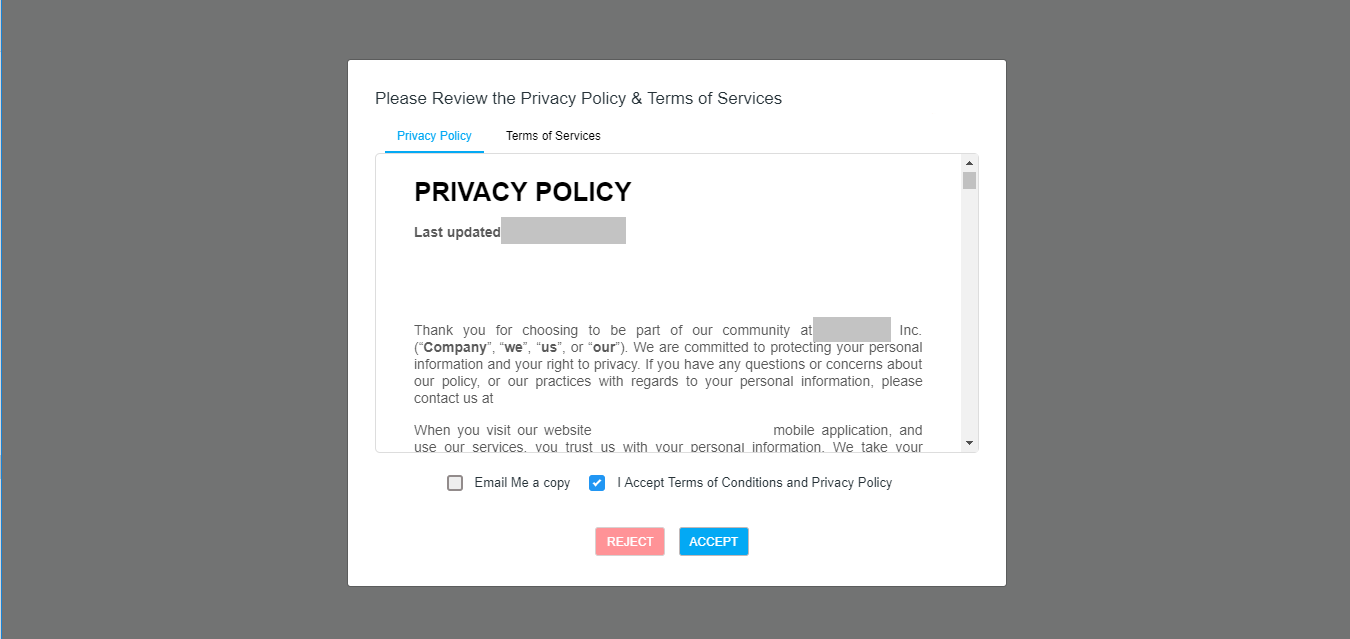
**CHAPTER:4**

External Interface Requirement

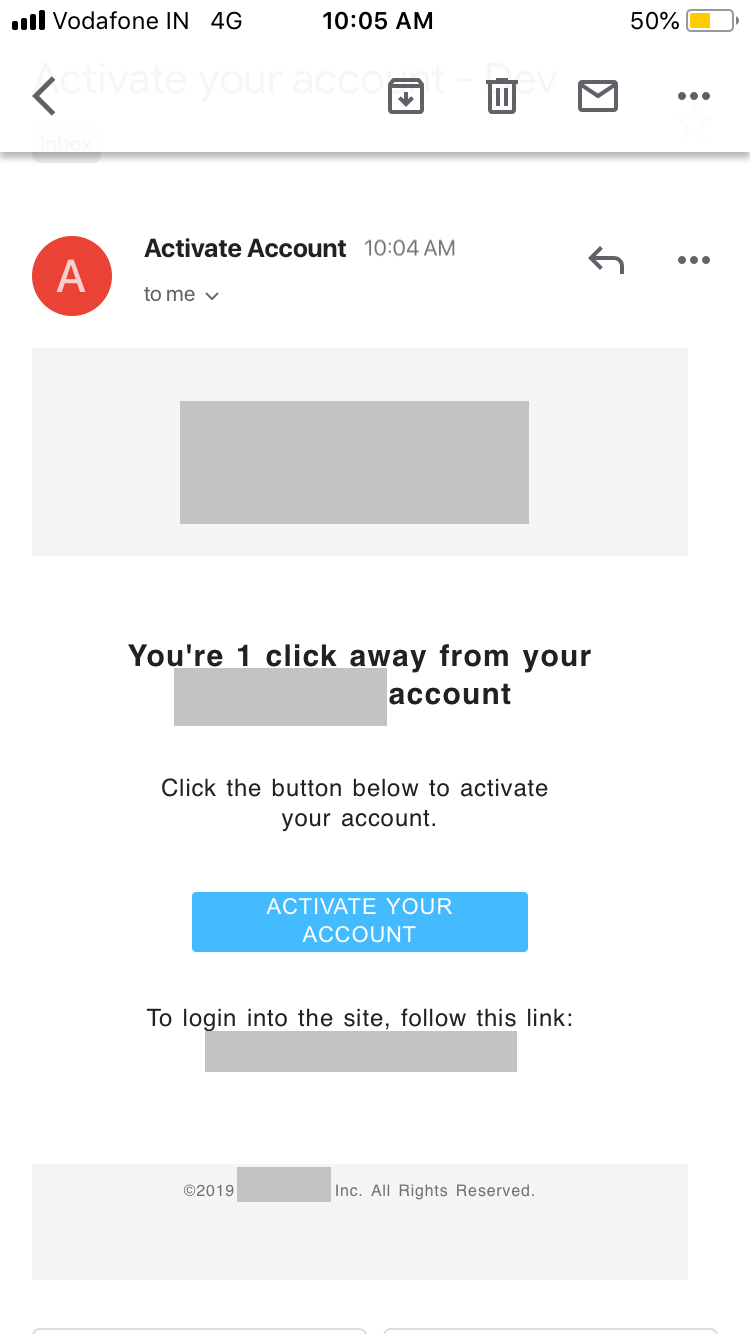
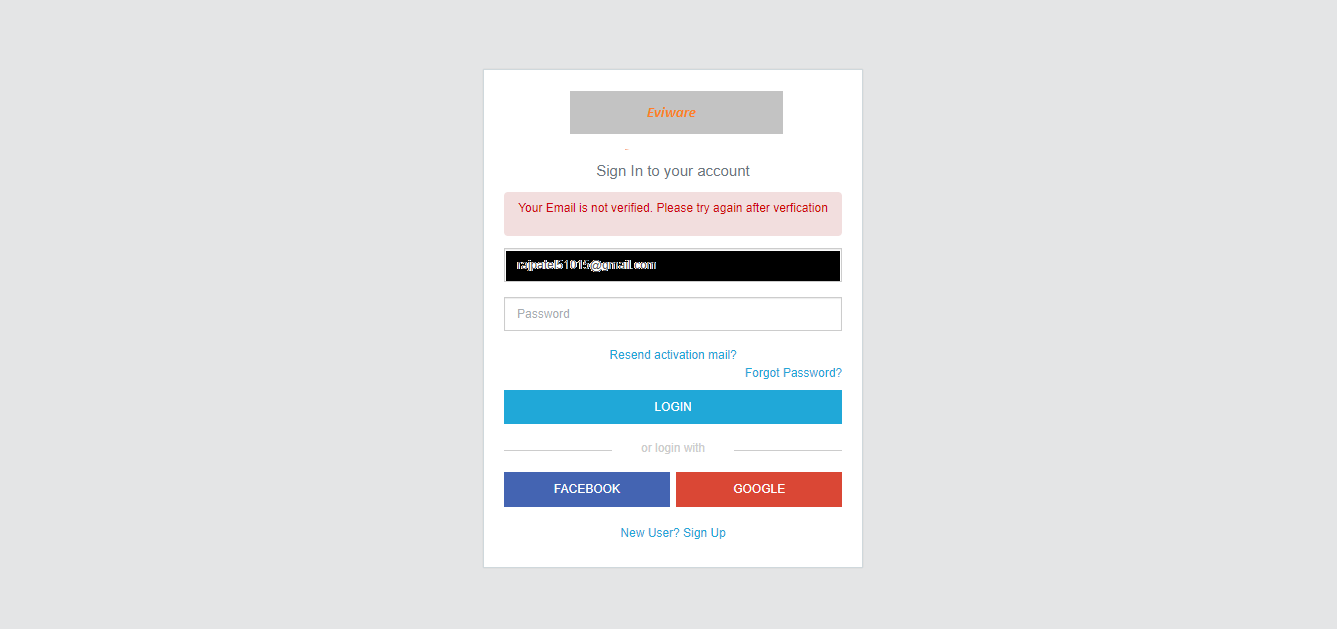
**4.1 User Interfaces**

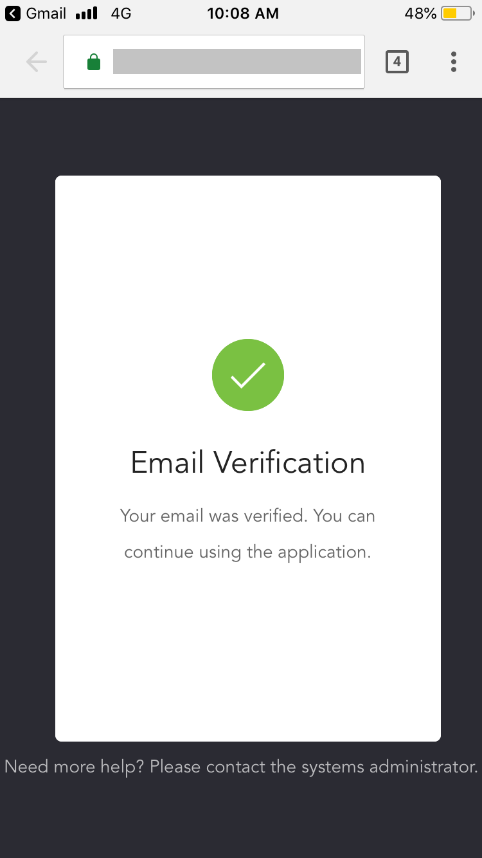
**4.1.1 Registration: 4.1.2 Login:**



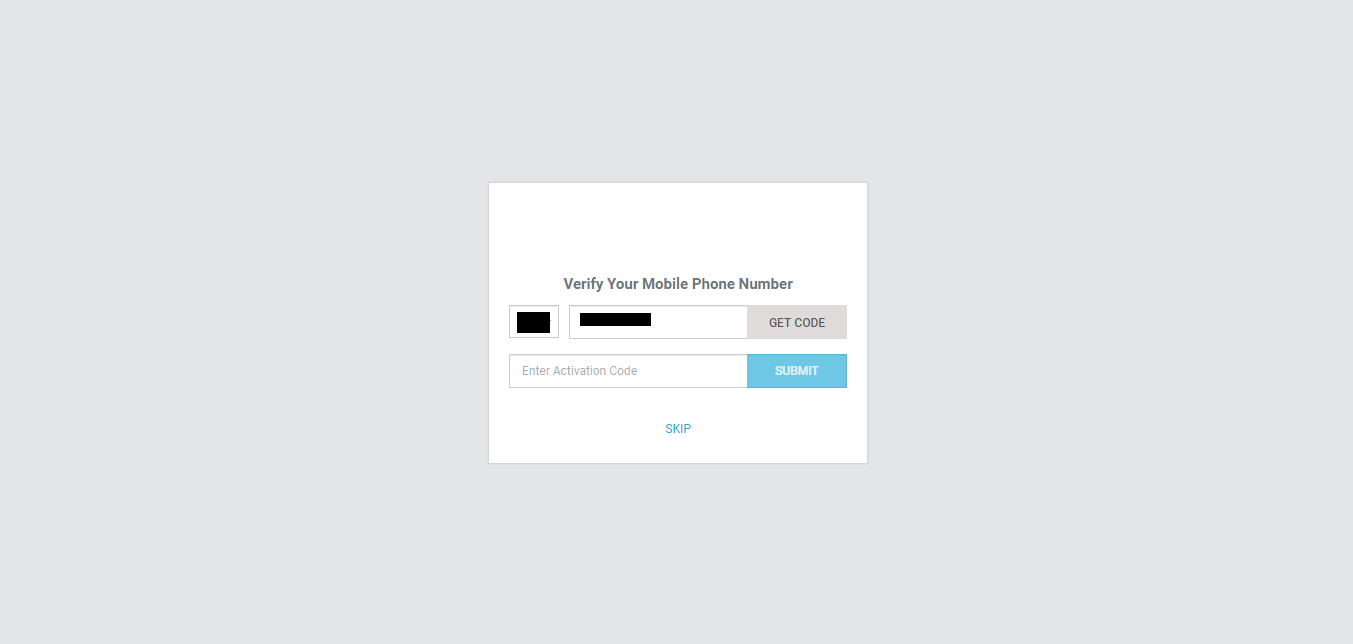


**4.1.3 Email verification.**

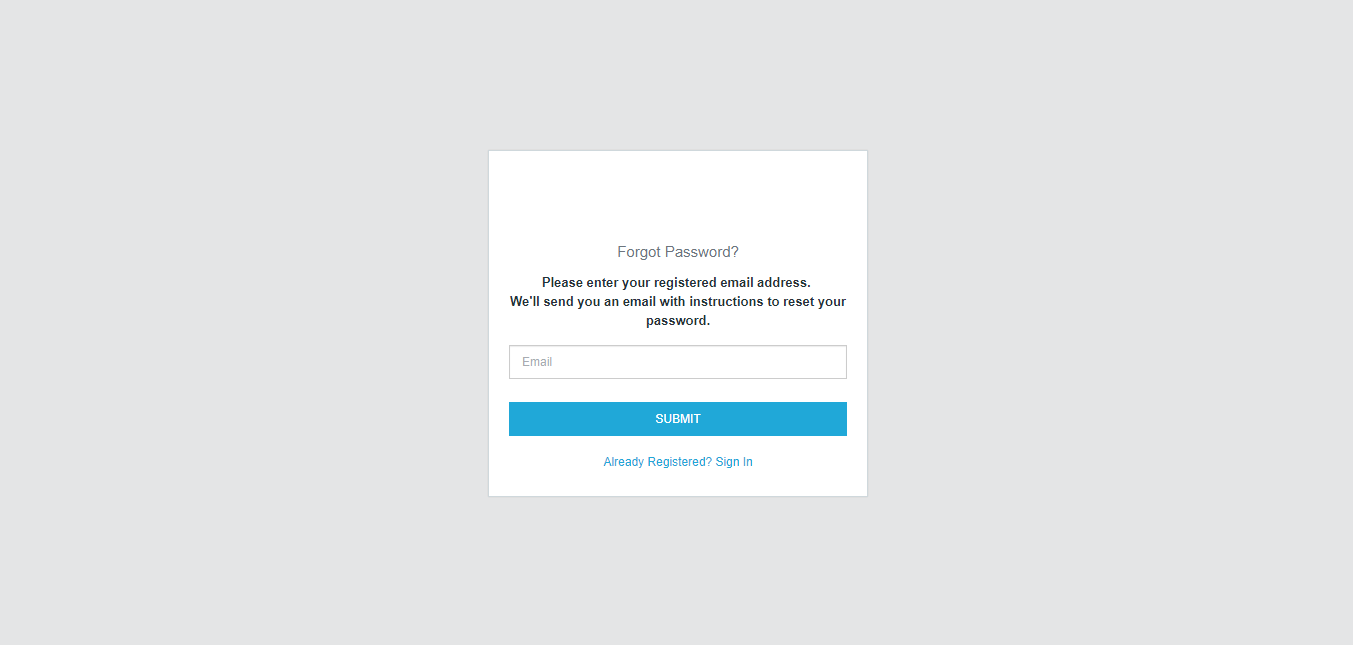




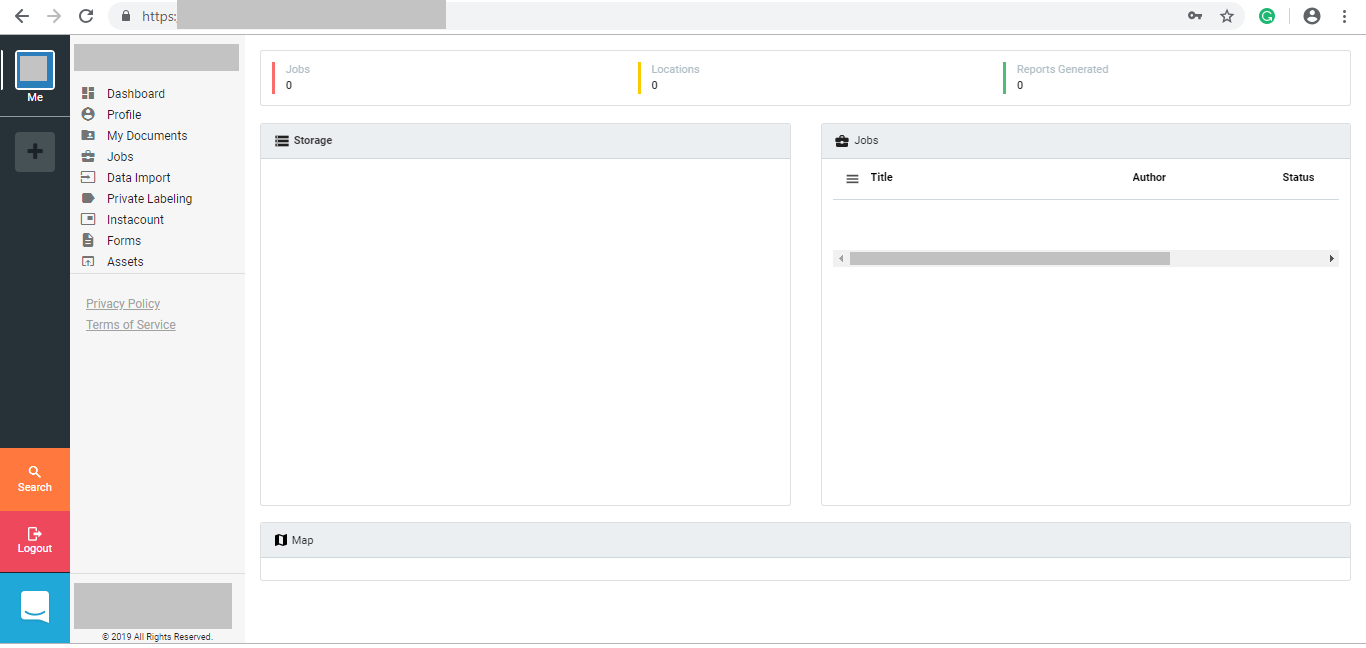
**4.1.4 Phone number verification:**



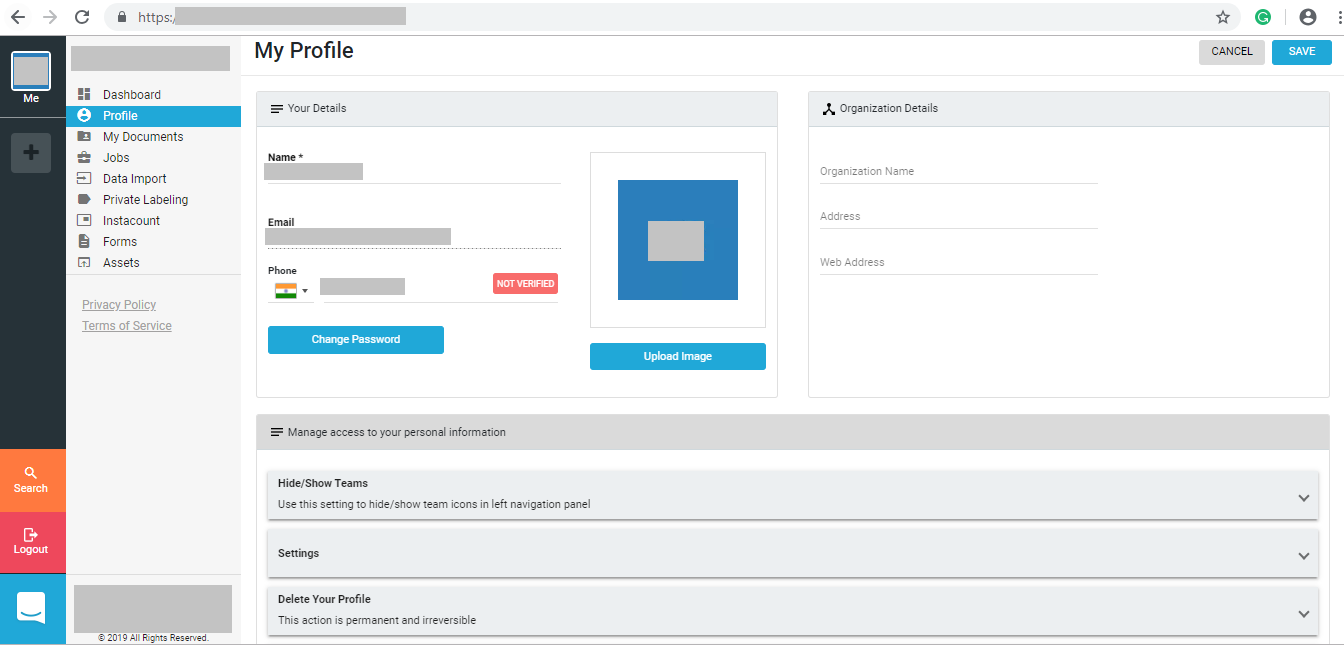
**4.1.5 Forgot Password:**



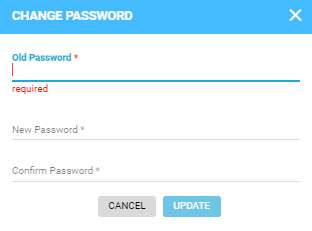
**4.1.6 Main Page and Dashboard:**



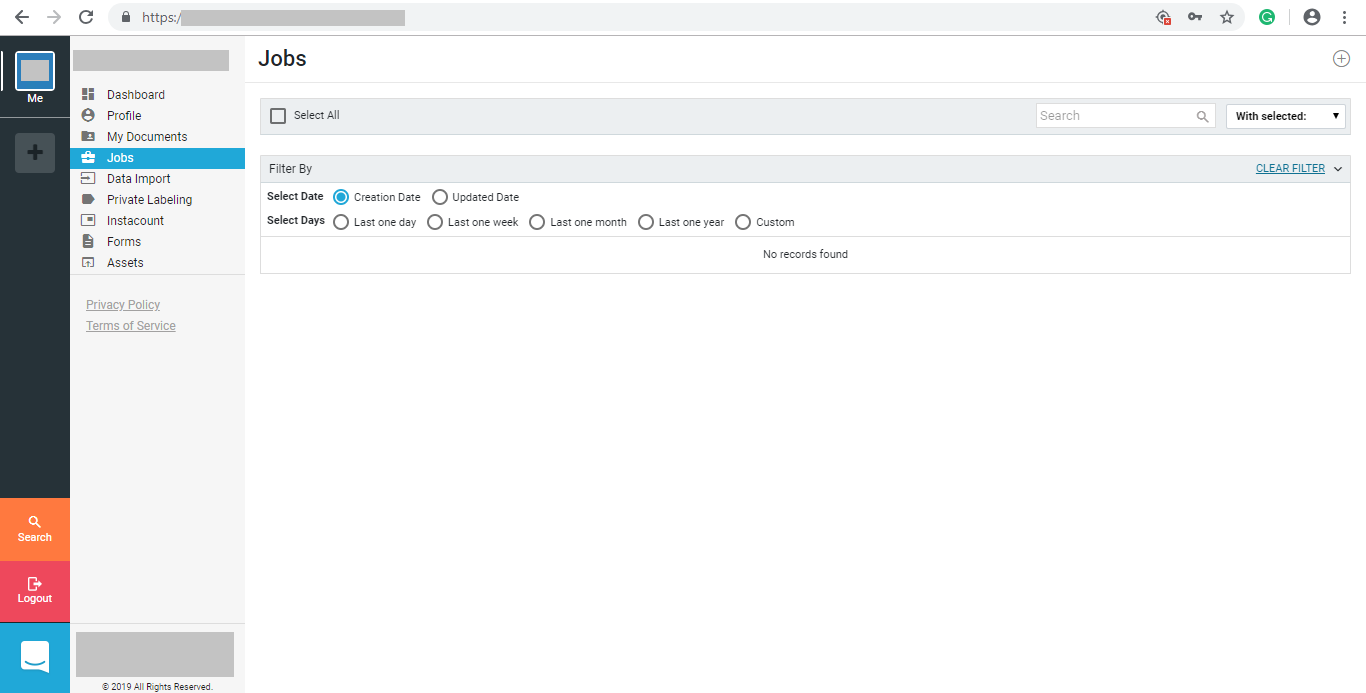
**4.1.7 Profile:**



**4.1.8 Change Password:**

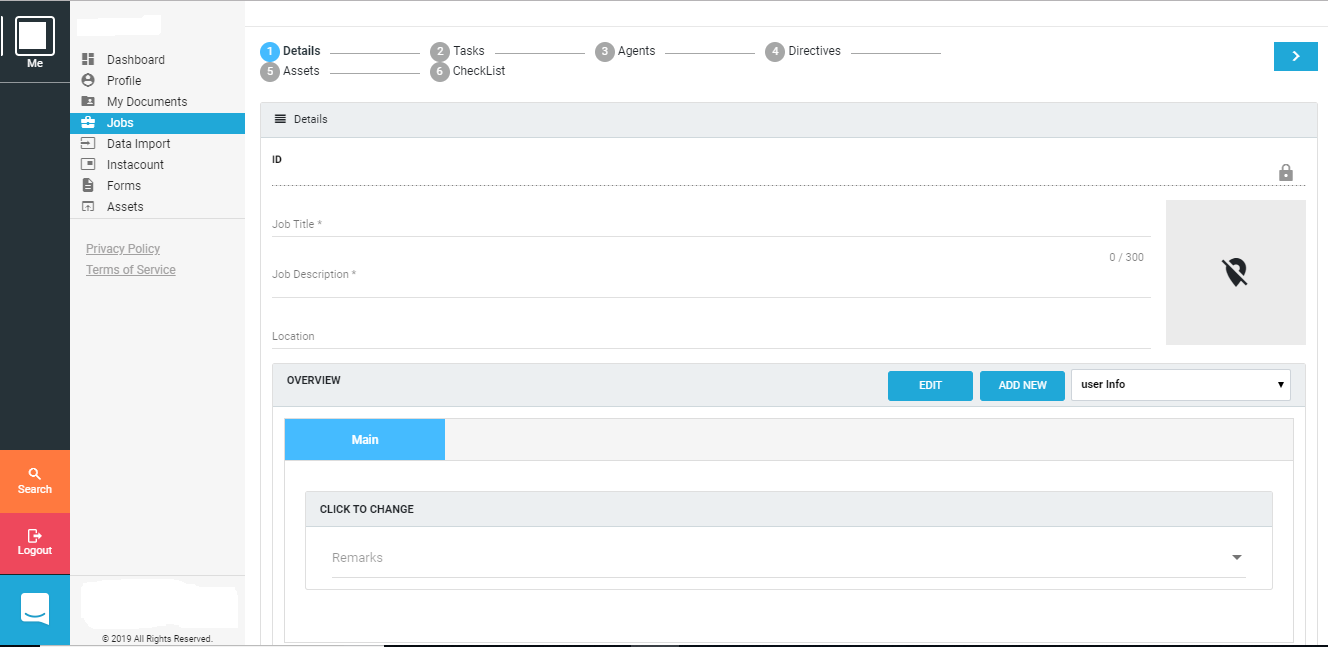


**4.1.7 Job Module main Page**

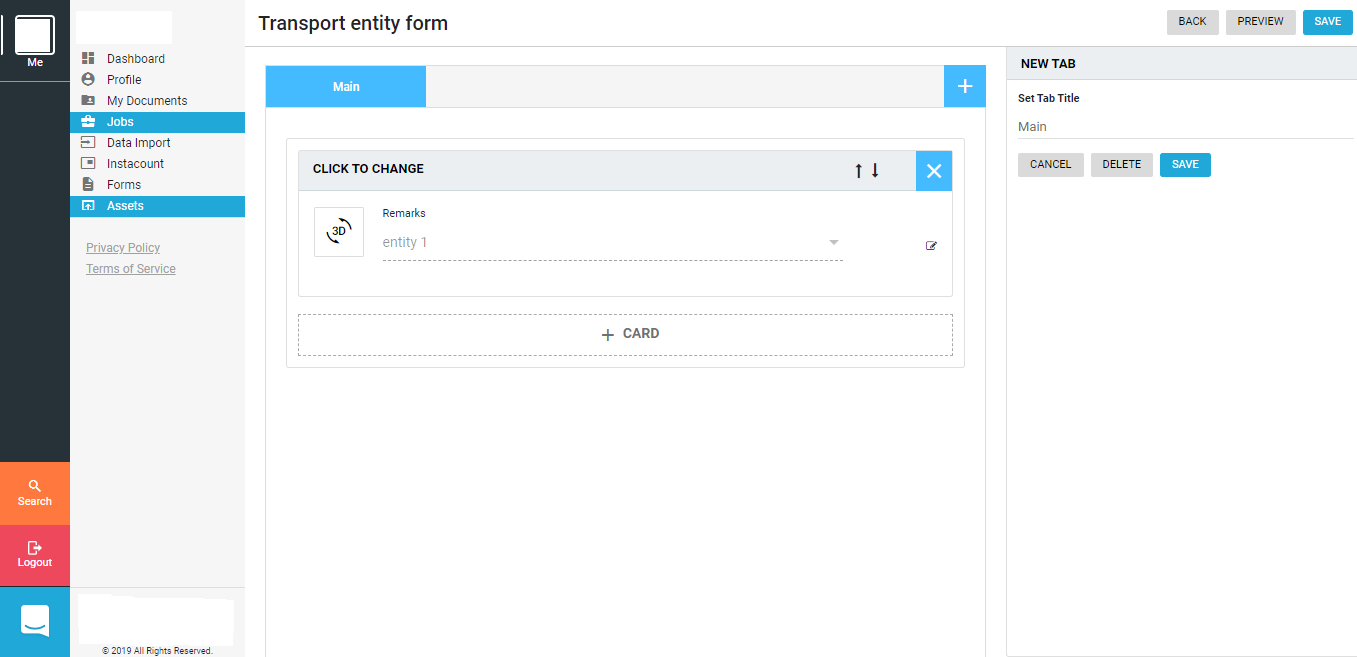


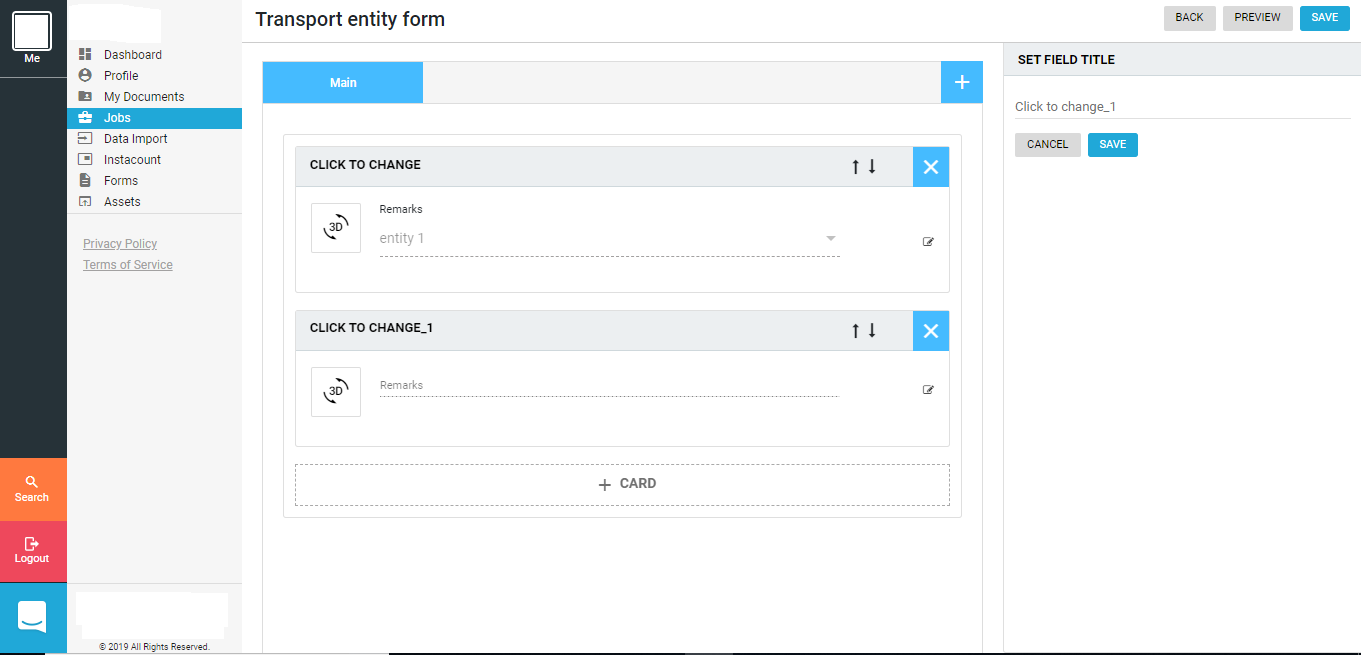
**4.1.8 Create Job**

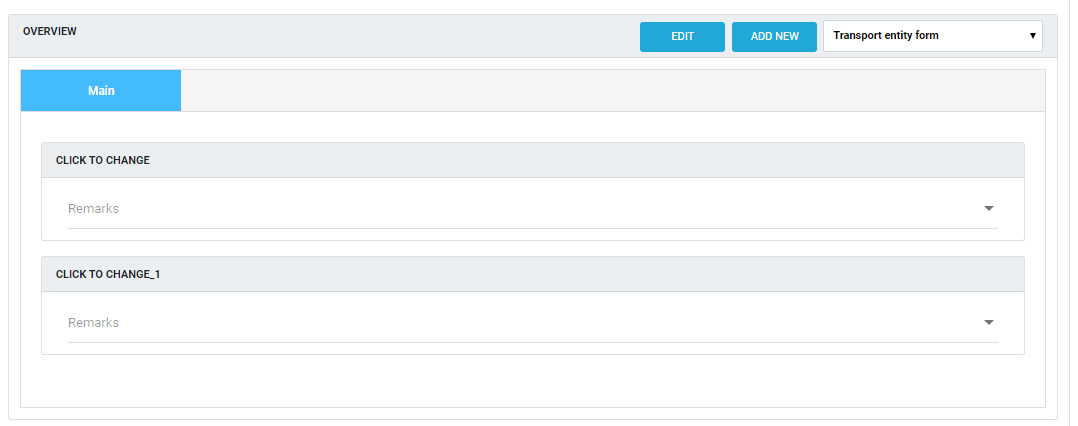
**Details:**

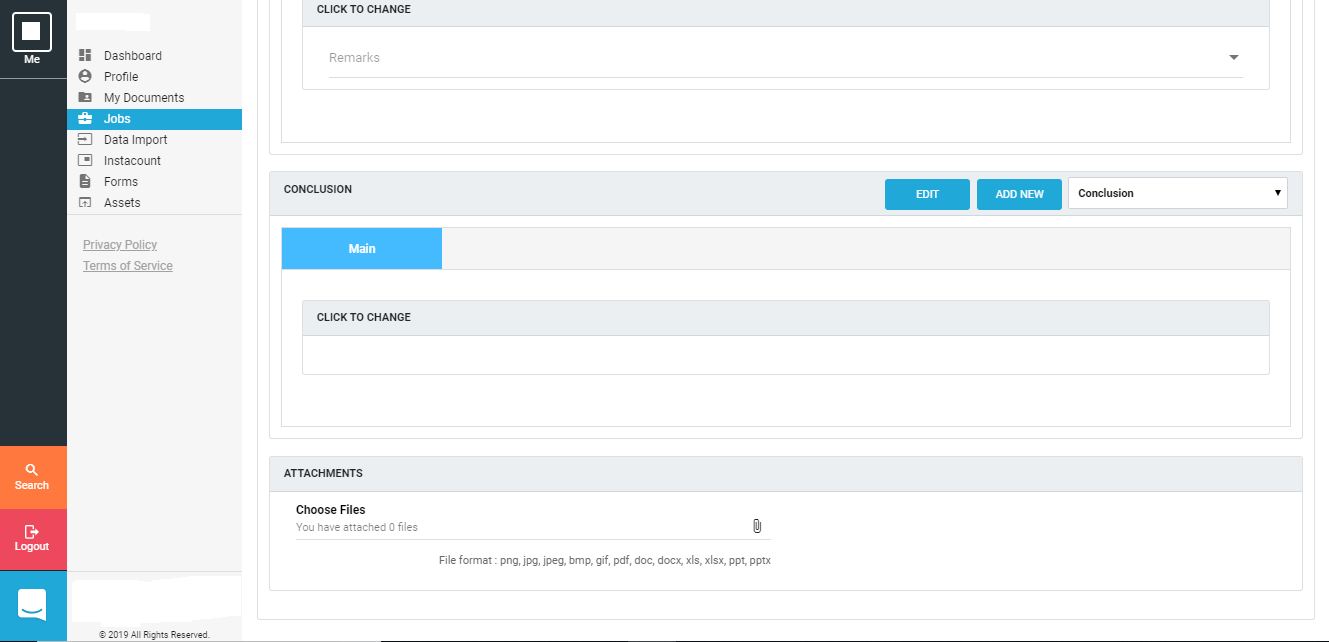


**Edit or Add New Overview/Conclusion form:**

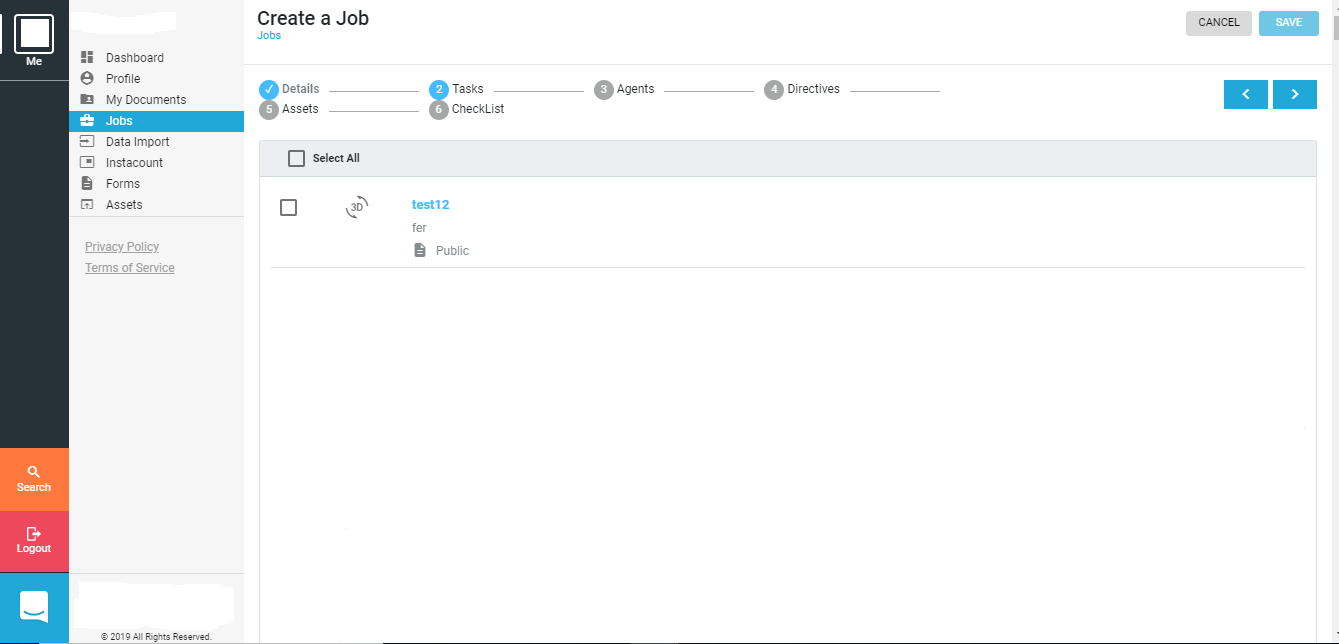




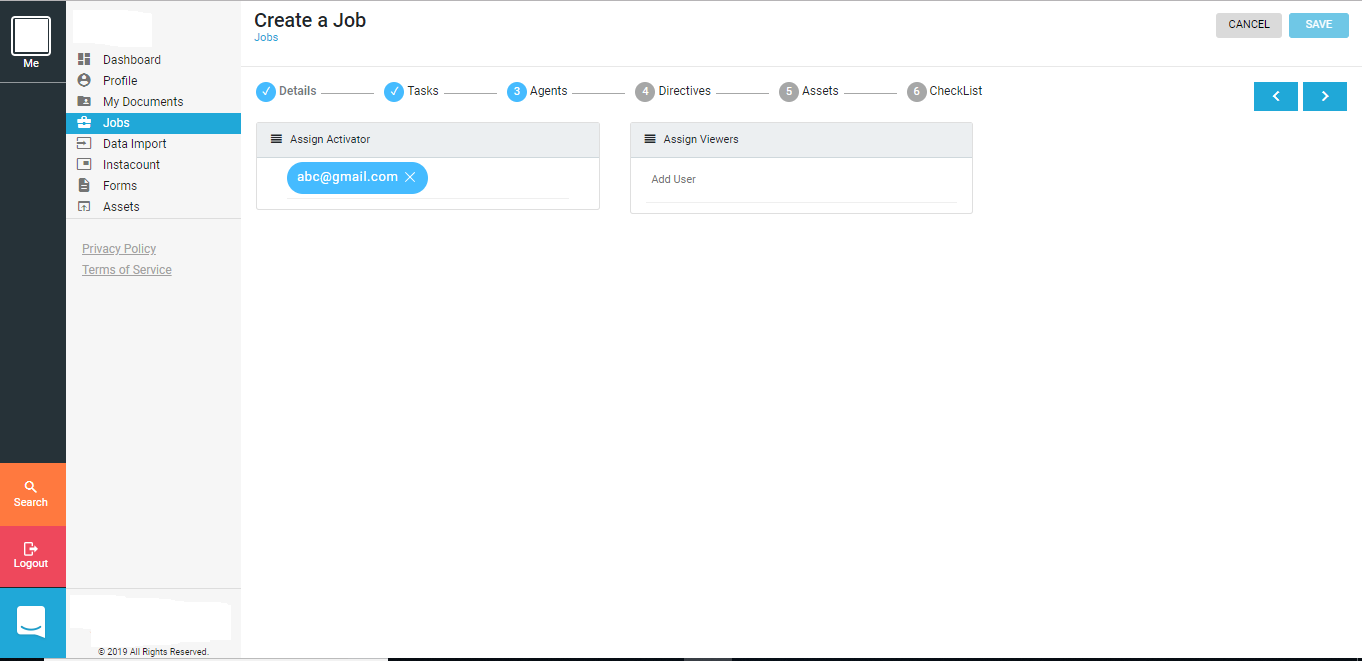


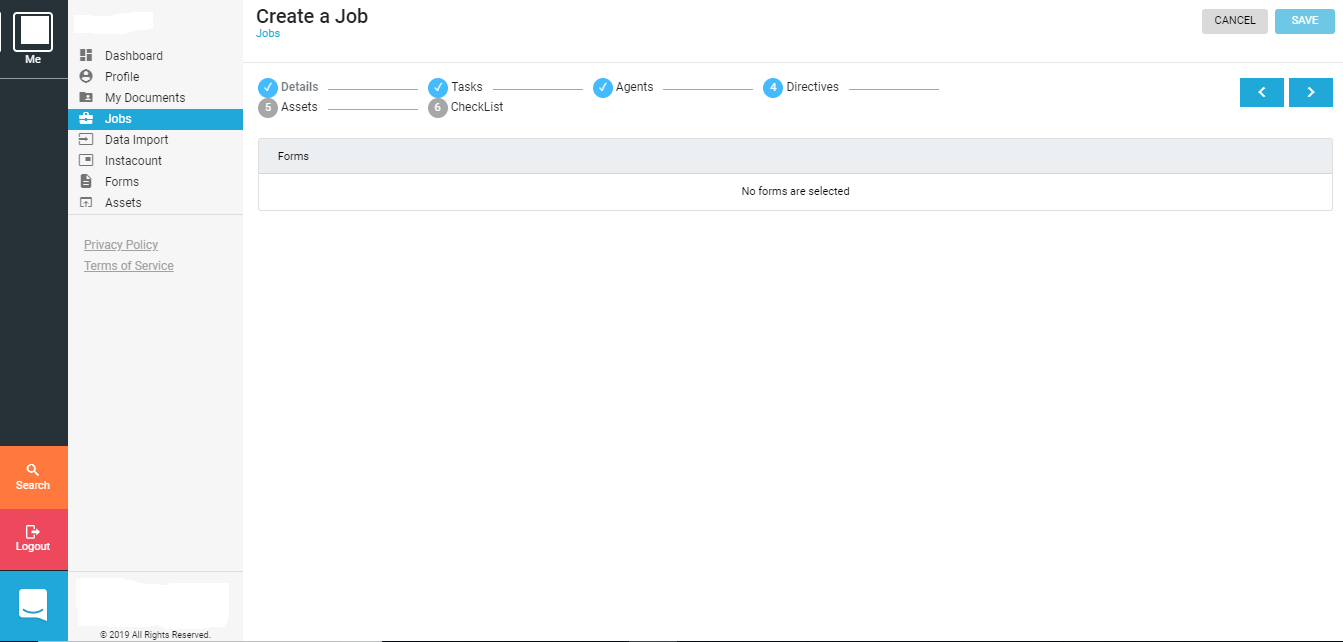


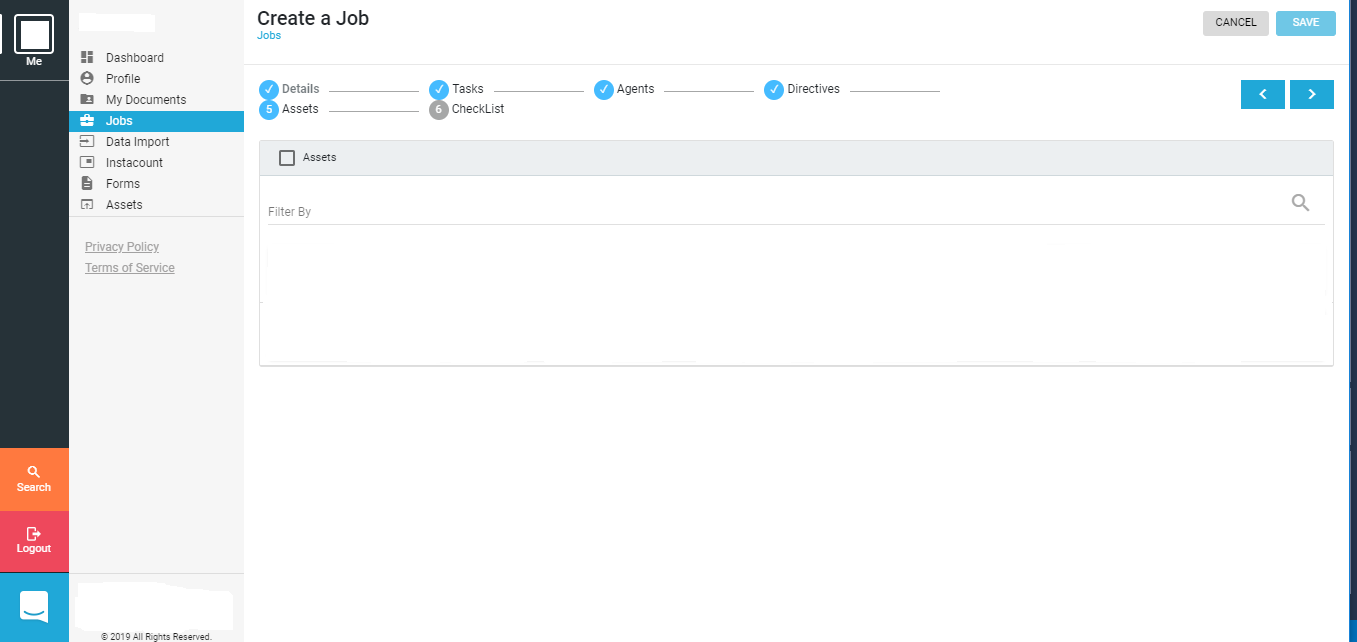
**Tasks:**

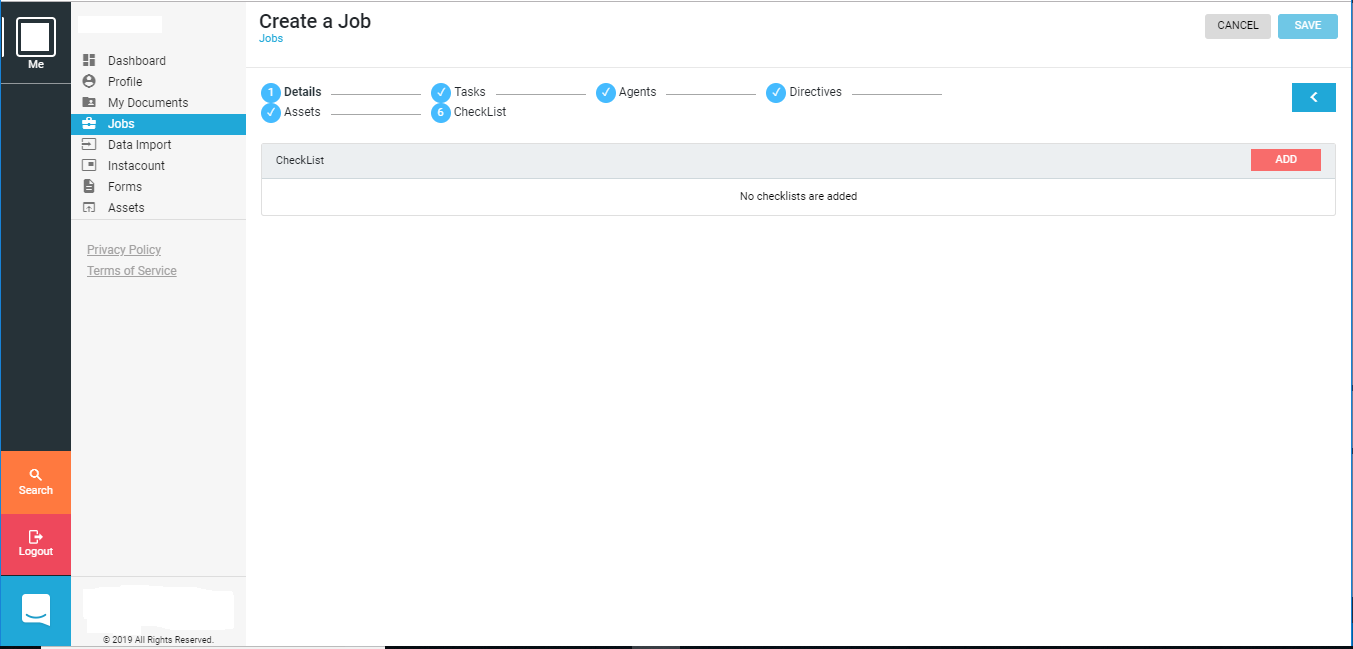


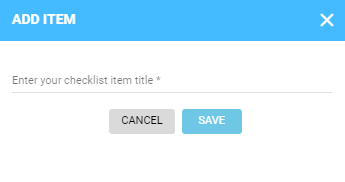
**Agents:**



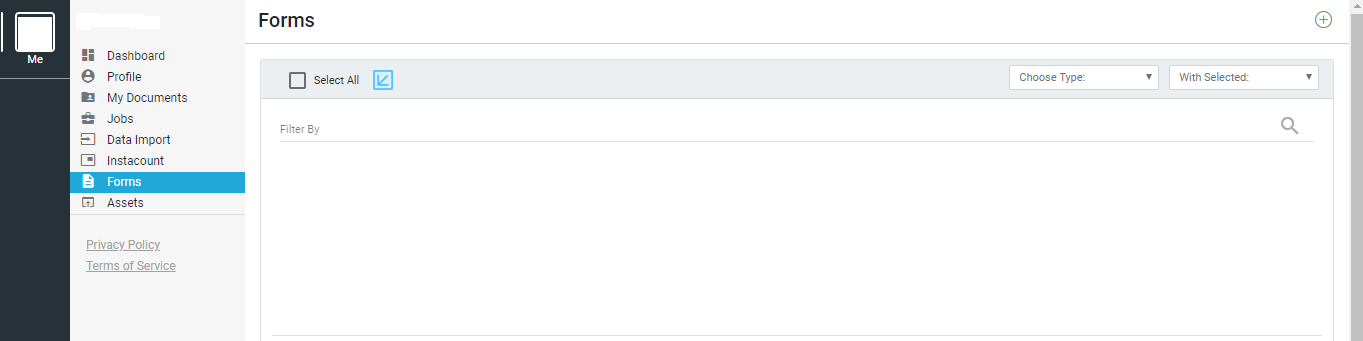


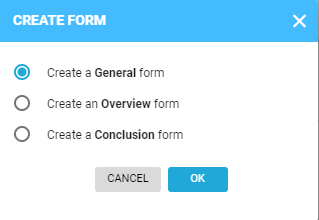




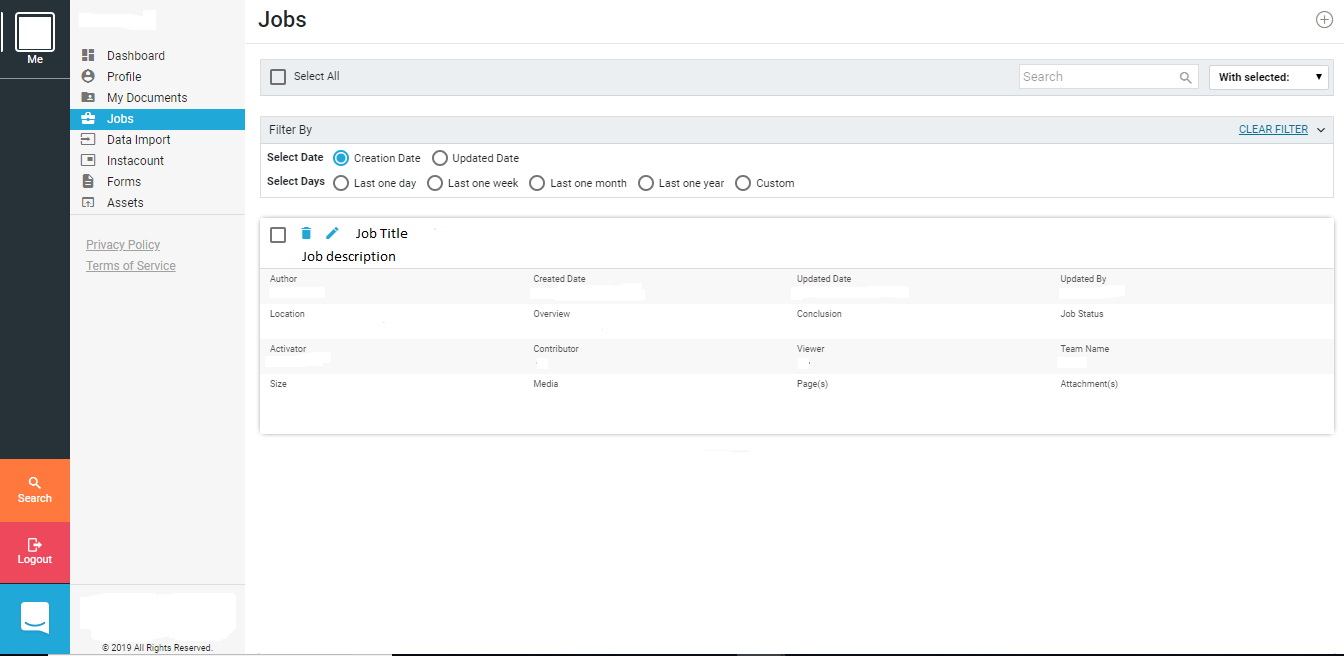


**4.1.9 Form**

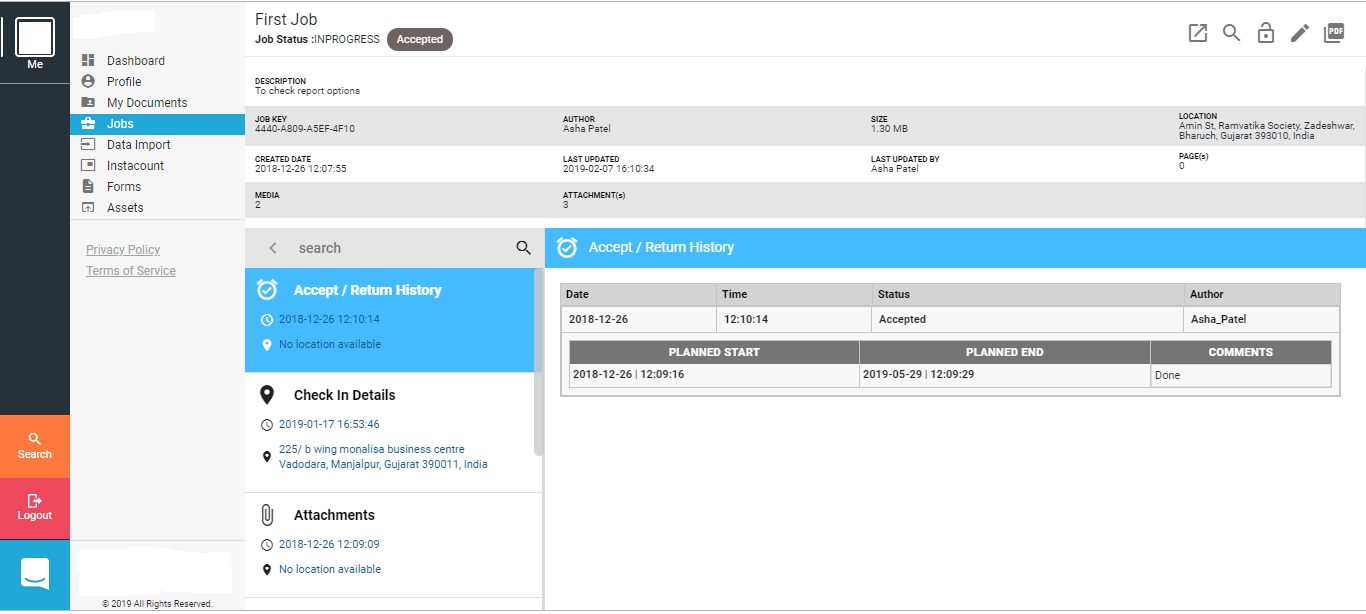




**4.1.10 Jobs Overview**



* + 1. **Edit Job**



* 1. **Implementation Environment**

4.2.1 Hardware Requirement:

* PC/Laptop with internet access.
* Memory:16GB
* Processor: Intel(R) Core i5-7400 CPU @ 3.00GHz
* RAM: 16.0 GB
* System type: 64bit operating system.

4.2.2 Software Requirement:

* + Frontend: Angular, HTML, CSS, Bootstrap
  + Backend: Asp.net Core
  + Database: PostgreSQL
    1. Communication interfaces:
* HTTP, SMTP

**CHAPTER:5**

System Features

**5.1**

**System Feature 1: Registration**

* Name
* Email (Verification required)
* Contact number (Optional verification)
* Password, Confirm Password

**System Feature 2: Login**

* Login using Facebook or Google.
* Login by entering email address and password.

**System Feature 3: Profile Update**

* Edit user details
* Upload Profile picture

**System Feature 4: Job creation**

* Job details
* Add agents (Viewer, Activator)
* Add assets
* Add task
* Create/Add checklist

**System Feature 5: Form creation (Dynamic Form Builder)**

* Create basic form for task
* Create Overview form
* Create Conclusion form

**System Feature 6: Report Generation**

* Generate report that provides summery of the Job.
* Select formatting options for more general describing view of report.

**System Feature 7: Universal search**

* Search across entire application and display result in form of links.
* Module wise search result list.

**System Feature 8: Document and contact management**

* List of all the documents uploaded for the specific Job with require details.
* Stores all the registered contacts.
* Able to create document report same as Job report generation.

**CHAPTER: 6**

Other Nonfunctional Requirements

**6.1 Performance Requirements**

* The program or application should run at such a speed that the user can move at his/her own space, without noticing interruption due to processing.
* Data should flow between different modules before page loading.

**6.2 Safety Requirements**

* This software is intended to communicate over an internal network, therefore security is of little concern.
* The user will have to enter the username and password so the program can connect to the server and allow to use the functionalities provided by the system.

**6.3 Security Requirements**

Some of the factors that are identified to protect the software from accidental or malicious access, use, modification, destruction, or disclosure are described below.

* The system, at any time, should be accessed only by the authenticated users.
* Network communications should use cryptographic protocols such as SSL.
* The system is required to end the session automatically, when an open session is not used for a specific period of time.
* Utilize certain cryptographic techniques
* Keep specific log or history data sets
* Assign certain functions to different modules
* Restrict communications between some areas of the program
* Check data integrity for critical variables

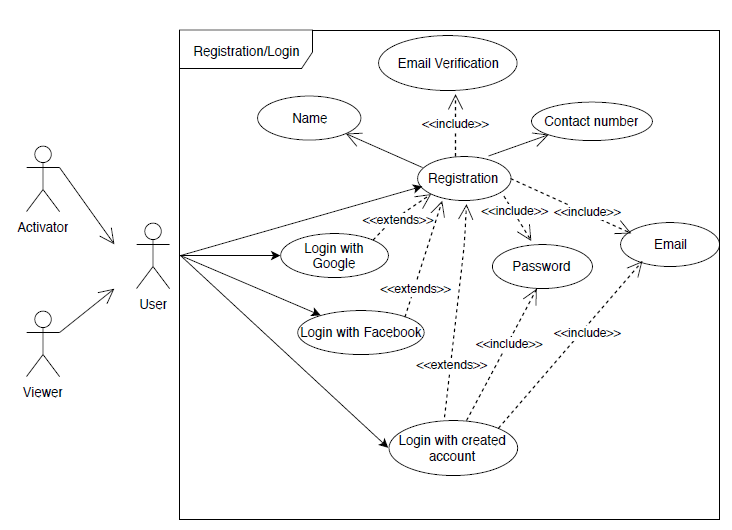
**6.4 Software Quality Attributes**

**CHAPTER: 7**

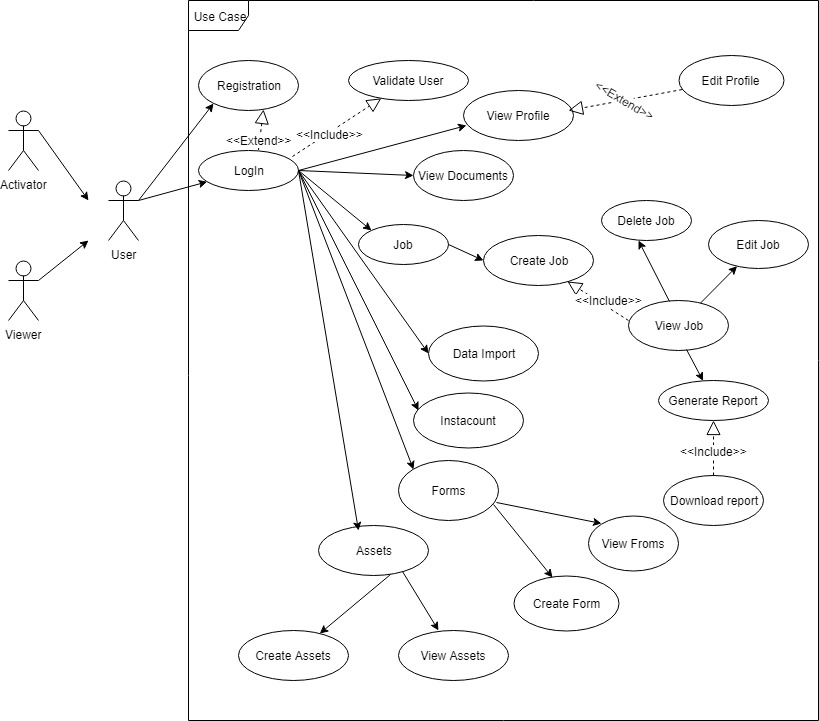
**Future Enhancement**

**Appendix A: Design Documentation**

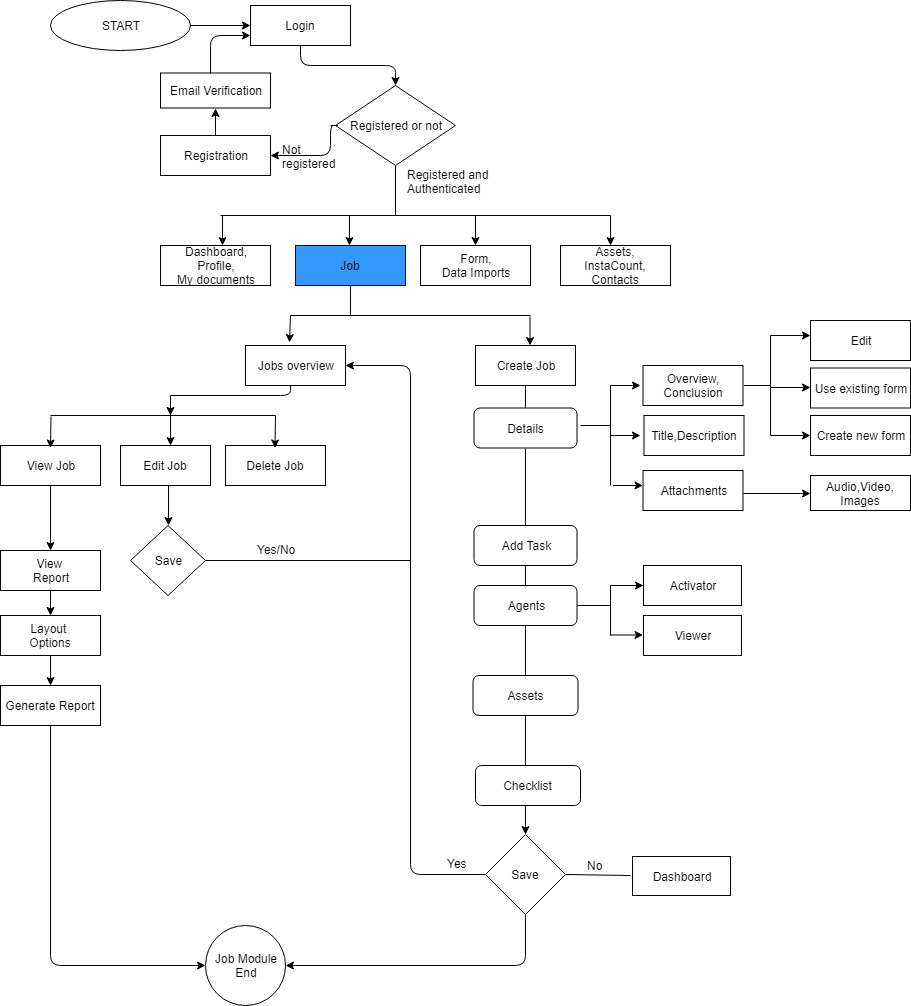
1. **Use Case Diagram:**
2. **Registration and login**

****

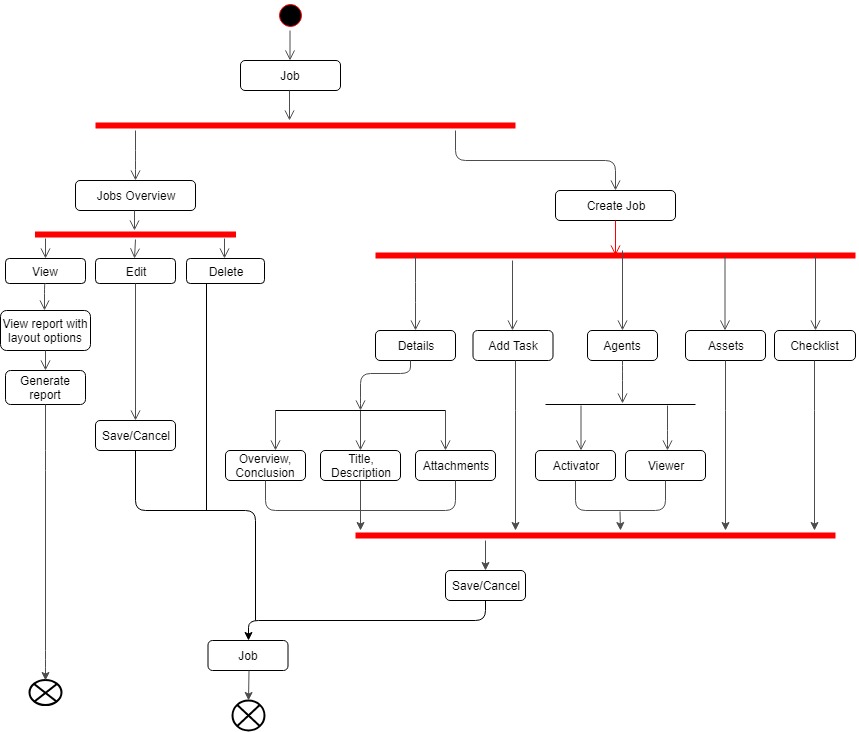
1. **Job Module**



1. **Module Flow:**

****

1. **Activity Diagram:**



1. **Data Dictionary:**
   * 1. **User:**
     2. Contains the details of the user.

|  |  |  |  |
| --- | --- | --- | --- |
| **Fieldname** | **Datatype** | **Size** | **Constraints** |
| id | Varchar | 50 | Primary key |
| name | Varchar | 100 | Not null |
| email | Varchar | 100 | Not null |
| phone | Numeric | 15 | - |
| password | Varchar | 25 | Not null |

**7.4.2 Form:**

* + 1. Contains the details of the form.

|  |  |  |  |
| --- | --- | --- | --- |
| **Fieldname** | **Datatype** | **Size** | **Constraints** |
| id | Varchar | 50 | Primary key |
| user\_id | Varchar | 50 | Foreign key |
| name | Varchar | 100 | Not Null |
| description | Text | - | - |
| file\_url | Text | - | Not null |
| created\_on | Timestamp | - | Not null |
| form\_type | Numeric | 1 | Must be 1/2/3 |

**7.4.3 Job:**

* + 1. Contains the details of the job.

|  |  |  |  |
| --- | --- | --- | --- |
| **Fieldname** | **Datatype** | **Size** | **Constraints** |
| id | Varchar | 50 | Primary key |
| user\_id | Varchar | 50 | Foreign key |
| title | Varchar | 50 | Not null |
| description | Text | - | - |
| overview\_form | Varchar | 50 | Foreign key |
| conclusion\_form | Varchar | 50 | Foreign key |
| created\_on | Timestamp | - | Not null |
| status | Numeric | 1 | Must be 1/2/3 |
| location\_text | Varchar | 100 | Not null |
| location\_longitude | Varchar | 100 | Not null |
| location\_latitude | Varchar | 100 | Not null |
| special\_instruction | Text | - | - |

* + 1. **Job Attachments:**
    2. Contains the details of the form.

|  |  |  |  |
| --- | --- | --- | --- |
| **Fieldname** | **Datatype** | **Size** | **Constraints** |
| id | Varchar | 50 | Primary key |
| job\_id | Varchar | 50 | Foreign key |
| user\_id | Varchar | 50 | Foreign key |
| name | Varchar | 100 | Not Null |
| file\_url | Text | - | Not null |
| created\_on | Timestamp | - | Not null |

**7.4.5 Job Tasks:**

* + 1. Contains the details of the form.

|  |  |  |  |
| --- | --- | --- | --- |
| **Fieldname** | **Datatype** | **Size** | **Constraints** |
| job\_id | Varchar | 50 | Foreign key |
| form\_id | Varchar | 50 | Foreign key |

**7.4.6 Job Agents:**

* + 1. Contains the details of the form.

|  |  |  |  |
| --- | --- | --- | --- |
| **Fieldname** | **Datatype** | **Size** | **Constraints** |
| job\_id | Varchar | 50 | Foreign key |
| user\_id | Varchar | 50 | Foreign key |
| agent\_type | Numeric | 1 | Must be 1/2 |

* + 1. **Job Checklist:**
    2. Contains the details of the form.

|  |  |  |  |
| --- | --- | --- | --- |
| **Fieldname** | **Datatype** | **Size** | **Constraints** |
| job\_id | Varchar | 50 | Foreign key |
| checklist\_item | Varchar | 100 | Not null |

* 1. **Job Assets:**
     1. Contains the details of the form.

|  |  |  |  |
| --- | --- | --- | --- |
| **Fieldname** | **Datatype** | **Size** | **Constraints** |
| job\_id | Varchar | 50 | Foreign key |
| asset\_url | Text | - | Not null |

1. **Class Diagram:**
2. **Test Cases**

**1: Login**

This test case checks whether the login by user will be successfully done or not in the given system.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sr No.** | **Description** | **Test Steps** | **Test data** | **Expected Results** | **Actual Results** | **Pass/**  **Fail** |
| 1. | Check Customer Login with valid Data | 1. Go to Site 2. Enter email 3. Enter password 4. Click Login | email: [abc@xyz.com](mailto:abc@xyz.com)  password:  abc2468@xyz | Login should be successful | As expected | Pass |
| 2. | Check Customer Login with invalid Data | 1. Go to Site 2. Enter email 3. Enter password 4. Click Login | email: [acd@xyz.com](mailto:acd@xyz.com)  password:  abc1357@xyz | User should not Login into an application  Message:  Invalid email or password. | As expected | Pass |

**2: Registration**

This test case checks whether the registration by the user will be successfully done or not in the given system.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sr No.** | **Description** | **Test Steps** | **Test data** | **Expected Results** | **Actual Results** | **Pass/**  **Fail** |
| 1. | Check Customer Registration data with validation | 1. Go to Site 2. Enter required details. 3. Click Create Account | Name: abc  Email: [abc@xyz.com](mailto:abc@xyz.com)  Country:  India(+91)  Phone number:  9978695768  Password:  abc2468@xyz  Confirm Password | Password and Confirm Password should be match.  Phone number according to country code should be match.  Email must be valid. | As expected | Pass |
| 2. | Check invalid format of email | Follow above steps with invalid email | email: [acd@xy.com](mailto:acd@xy.com) | User should not register. Message: Invalid email. | As  expected | Pass |
| 3. | Check different Phone number digits with country code. | Follow above steps with invalid Phone number digits | Country:  India(+91)  Phone number:  997869576 | User should not register. Message: Invalid phone number. | As  expected | Pass |
| 4. | Enter password which does not match with regex. | Follow above steps with wrong password format | Password:  abdhsgj | User should not register.  Message:  Password must be alphanumeric including at least 1 uppercase letter,1 lowercase letter with minimum 8 characters. | As  expected | Pass |
| 5. | Enter different entries in password and confirm password fields. | Follow above steps with different entries in given fields | Password:  asdf1357@asd  Confirm password:  asdf145@asgh | User should not register. Message: ‘Confirm Password' and 'Password' do not match. | As  expected | Pass |

**Appendix B: Glossary**

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Database | Collection of all the information monitored by this system. |
| Software Requirements Specification | A document that completely describes all of the functions of a proposed system and the constraints under which it must operate. For example, this document. |
| Stakeholder | Any person with an interest in the project who is not a developer. |
| User | Reviewer or Author. |