|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Register Client | | **USE CASE TYPE** |
| **USE CASE ID:** | 3.1 | | Business Requirements:🞎 |
| **PRIORITY:** | High | | System Analysis: **🗹** |
| **SOURCE:** | Minions – Requirement List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Client | | |
| **PRIMARY SYSTEM ACTOR** | Operational Head | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * None | | |
| **DESCRIPTION:** | This Use Case describes the event where a first time Client goes directly onto the system to register him/herself to gain access to all the functionalities available to him/her, or calls in instead for the Operational Head to add him/her onto the system first before performing any other task. | | |
| **PRE-CONDITION:** | The client is not a registered on the system. | | |
| **TRIGGER:** | The client wants to register on the system. | | |
| **TYPICAL COURSE**  **OF EVENTS:** | Actor Action | System Response | |
| **Step 1**: The client goes on the system and selects to register. | **Step 2:** The system requests the client’s details. | |
| **Step 3:** The client enters their full:   * Name(s) and surname(s) * Physical address * Contact details * Email address * Password and the confirmation password. | **Step 4:** The system captures and validates the new Client’s details. | |
|  | S**tep 5:** On successful validation, a new client is added to the ‘**Client Table’.** | |
|  | **Step 6:** Notification is sent to the client to confirm successful registration. | |
|  |  | |
| **ALTERNATE COURSES:** | **Alt Step 1:** The client calls in and gets the Operational Head on the line. | | |
| **Alt Step 2:** The Operational Head prompts the client for his details | | |
| **Alt Step 3:** The client provides name, surname, location, contact details and email address. | | |
| **Alt Step 4a**: The Operational Head captures the details onto the system and the system checks if client already exists, if not the operational registers the Client and a notification is automatically sent to the client with his temporary login details. | | |
| **Alt Step 5:** New Client Details cannot be validated, and error message is displayed. | | |
|  | | |
| **CONCLUSION:** | The system saves the client’s details and transaction information in the **Audit Trail Table** and notifies Client. | | |
| **POST-CONDITION:** | * The client is a member of the system. | | |
| **BUSINESS RULES** | * Credentials are confidential, meaning client should not share their login details. | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** | * None | | |
| **ASSUMPTIONS:** | * Details provided by the client are truthful | | |
| **OPEN ISSUES:** | * None | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | View profile | | **USE CASE TYPE** |
| **USE CASE ID:** | 3.2 | | Business Requirements:🞎 |
| **PRIORITY:** | High | | System Analysis: **🗹** |
| **SOURCE:** | Minions – Requirement List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Client | | |
| **PRIMARY SYSTEM ACTOR** | None | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * None | | |
| **DESCRIPTION:** | This use case describes the event where a Client or the Operational Head would like to view details about the profile he/she created on the system. | | |
| **PRE-CONDITION:** | The client must be registered into the system. | | |
| **TRIGGER:** | The user logs in to the system. | | |
| **TYPICAL COURSE**  **OF EVENTS:** | Actor Action | System Response | |
| **Step 1**: On the system, the Client selects to View his/her Profile. | Step 2: The system retrieves information from the ‘Client Table’. | |
|  | **Step 3:** The System displays the Client’s profile. | |
| **ALTERNATE COURSES:** |  | | |
|  |  | | |
|  |  | | |
| **CONCLUSION:** | The system displays the current user’s profile with all his personal information. | | |
| **POST-CONDITION:** | * The user has successfully viewed his profile. | | |
| **BUSINESS RULES** | * None | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** | * None | | |
| **ASSUMPTIONS:** | * None | | |
| **OPEN ISSUES:** | * None | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | 3.3 | | **USE CASE TYPE** |
| **USE CASE ID:** | Maintain Client | | Business Requirements: 🞎 |
| **PRIORITY:** | High | | System Analysis:🗹 |
| **SOURCE:** | Minions – Requirements List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Client | | |
| **PRIMARY SYSTEM ACTOR** | Operational Head | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * None | | |
| **DESCRIPTION:** | This use case describes the event in which a Client either goes on to the system to update information about his/her personal details or instead, calls in the Operational Head who will make those changes for him/her. | | |
| **PRE-CONDITION:** | The Client is registered on the system. | | |
| **TRIGGER:** | The Client is logged in and wants to update his/her details. | | |
| **TYPICAL COURSE** | Actor Action | System Response | |
| **OF EVENTS:** | **Step 1**: On the System, the client selects to Maintain Client. | **Step 2:** The system retrieves the Client’s details and prompts him/her to make necessary changes where possible. | |
|  | **Step 3:** The Client enters the updated information. | **Step 4:** The system captures the updated details**.** | |
|  |  | **Step 5:** The system saves the updated details and updates the ‘**Client Table’.** | |
|  |  |  | |
| **ALTERNATE COURSES:** | **Alt-Step 1:** The Client calls in to request profile update. | | |
|  | **Alt Step 2:** The Operational Head prompts the Client for the changes he/she wishes to change. | | |
|  | **Alt Step 3:** The Client provides the necessary details to be updated | | |
|  | **Alt Step 4:** The Operational Head captures these details and Saves the information onto the system which will then update the **Client Table.** | | |
|  |  | | |
|  |  | | |
| **CONCLUSION:** | The system successfully updates the client’s information and send an email notification. | | |
| **POST-CONDITION:** | The updated details must be added to the **Client Table** and a transaction trail is created in the **Audit Trail table.** | | |
| **BUSINESS RULES** | * None | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** | * None | | |
| **ASSUMPTIONS:** | * None | | |
| **OPEN ISSUES:** |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | 3.4 | | **USE CASE TYPE** |
| **USE CASE ID:** | Place Job Request | | Business Requirements: 🞎 |
| **PRIORITY:** | High | | System Analysis:🗹 |
| **SOURCE:** | Minions – Requirements List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Client | | |
| **PRIMARY SYSTEM ACTOR** | Operational Head | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * None | | |
| **DESCRIPTION:** | This use case describes the event in which a Client wants to request some task(s) being performed for him/her by Minions**.**  The client will either log onto the system or call in, provide which of the task(s) he/she wishes to be performed for him/her from the predefined set of tasks available, provide additional descriptions to these tasks if necessary and submit his/her request. | | |
| **PRE-CONDITION:** | The Client is registered on the system. | | |
| **TRIGGER:** | The Client is logged in and wants to place a job request. | | |
| **TYPICAL COURSE**  **OF EVENTS:** | Actor Action | System Response | |
| **Step 1:** The Client logs in the system and selects to Place Job Request. | **Step 2:** The System retrieves all the available categories of tasks under the **‘Task Type Table’** and prompts the Client to select the task(s) he would like to request. | |
| **Step 3:** The Client chooses from the categories of task, the one(s) he/she would like to place the order for. He has to indicate the:   * date * time * location for the request, and * name of preferred minion if applicable   He/she has to further provide specifications on the task(s) if necessary and submit his request. | **Step 4:** The system captures the Client’s selection. | |
|  | **Step 5:** The system saves this request in the **Booking Table.** | |
|  | **Step 6:** A notification is sent to the Client on successful job request made. | |
| **ALTERNATE COURSES:** | **Alt Step 1:** The Client calls in to place request | | |
| **Alt Step 2:** The Operational Head prompts the Client to provide description about his/her job request. | | |
| **Alt Step 4:** The Operational Head captures the Client request onto the System and saves it to the **Booking Table.** | | |
|  | | |
|  | | |
| **CONCLUSION:** | The Use Case concludes when a notification has been sent to the client about his request details. | | |
| **POST-CONDITION:** | Job request has been placed successfully and notification is sent to Client. | | |
| **BUSINESS RULES** | * Thorough and truthful details have to been given by the clients with regards to the task(s) requested, reason being that the predefined categories are generic and require more specifications so that the Operational Head could examine the request and come up with a quote. * Request must be placed at least 3 days before the date of the required task to be done, to give sufficient time to the Operational Head to assign (a) Minion(s) and to generate a quote. | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** |  | | |
| **ASSUMPTIONS:** |  | | |
| **OPEN ISSUES:** |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | View Quote | | **USE CASE TYPE** |
| **USE CASE ID:** | 3.5 | | **Business Requirements: 🞎** |
| **PRIORITY:** | High | | **System Analysis: 🗹** |
| **SOURCE:** | Minions – Requirements List | | **System Design: 🞎** |
| **PRIMARY BUSINESS ACTOR** | Client | | |
| **PRIMARY SYSTEM ACTOR** | None | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * None | | |
| **DESCRIPTION:** | After a job request has been placed, the Operational Head drafts a quote according to the client’s specifications.  This use case describes the event in which a Client wishes to view that quote before proceeding to **3.6.** | | |
| **PRE-CONDITION:** | The Client must have requested for a job to be done. | | |
| **TRIGGER:** | The Client wants to view the quote and is logged in already. | | |
| **TYPICAL COURSE**  **OF EVENTS:** | Actor Action | System Response | |
| **Step 1**: The Client selects to View the Quote | **Step 2:** The System retrieves all the quotes from the **‘Quote Table’** and prompts the Client to choose which one he wishes to view. | |
| **Step 3:** The Client selects the quote he wishes to view. | **Step 4:** The System displays the selected quote details such as:   * **Quote Number** * **Customer reference number** * **Total Amount payable** * **Company’s Bank Details** * **Payment Options** * **Client’s Information** | |
|  |  | |
| **ALTERNATE COURSES:** |  | | |  |
|  | | |
| **CONCLUSION:** | The system concludes when quote has been viewed by the Client. | | |
| **POST-CONDITION:** | Client has successfully viewed the Quote. | | |
| **BUSINESS RULES** | * Quotes are valid for 7 days | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** |  | | |
| **ASSUMPTIONS:** |  | | |
| **OPEN ISSUES:** |  | | |
|  | | | |
| **USE CASE NAME:** | 3.6 | | **USE CASE TYPE** |
| **USE CASE ID:** | Accept/Reject Quote | | **Business Requirements: 🞎** |
| **PRIORITY:** | High | | **System Analysis: 🗹** |
| **SOURCE:** | Minions – Requirements List | | **System Design: 🞎** |
| **PRIMARY BUSINESS ACTOR** | Client | | |
| **PRIMARY SYSTEM ACTOR** | None | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * Operational Head | | |
| **DESCRIPTION:** | This use case describes the event in which a Client either proceeds to payment or declines the quote leading to job request cancellation.  Accepting a quote here refers to selecting all or some of the tasks the client is wishing to pay for and actually going to pay for them. | | |
| **PRE-CONDITION:** | The Client must be logged in the system and must have viewed the quote. | | |
| **TRIGGER:** | The Client wishes to proceed to payment or reject the quote**.** | | |
| **TYPICAL COURSE**  **OF EVENTS:**  **ALTERNATE COURSES:** | Actor Action | System Response | |
| **Step 1**: The Client selects to Proceed To Payment on the quotation page. | **Step 2**: The system displays the quote with all the task(s) requested by the Client in **3.4 Place Job Request** alongside with their single costs and prompts the Client to select the task(s) he is willing to pay for. | |
| **Step 3:** The Client selects the single task available, or instead all or some of the multiple tasks listed. | **Step 4:** The system captures the Client selection and provides him/her with payment specifications. | |
| **Step 5:** The Client makes and sends proof of payment to Operational Head. |  | |
| **Step 6:** The Operational Head examines proof of payment and if successful, requests to update quotation status. | **Step 7:** prompts the Operational Head to select the suitable status for the quote. | |
| **Step 8:** The Operational Head changes the status to **‘Accepted’**. | **Step 9:** The System updates the quotation status on th**e** **‘Quote Table’** and notifies Client that the job should commence. | |
| **Alt Step 1:** The Client selects to Decline Quote option. | | |
| **Alt Step 2:** The System prompts the Client to confirm if he really want to pursue with quote rejection. | | |
| **Alt Step 3:** The Client confirms his/her selection. | | |
| **Alt Step 4:** The System validates input and updates the quotation status to ‘declined’ on the **‘Quote Table’**. | | |
| **Alt Step 6:** The proof of payment is invalid and the Operational Head prompts the Client to fix issues. | | |
| **CONCLUSION:** | The system concludes when a quote is converted into an invoice or when the client has rejected the quote. | | |
| **POST-CONDITION:** | The quote status is successfully updated and the Invoice generated if applicable. | | |
| **BUSINESS RULES** | * The Client has 7 days to accept the quote otherwise it gets expired and considered rejected. | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** |  | | |
| **ASSUMPTIONS:** | * None | | |
| **OPEN ISSUES:** | * None | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | 3.7 | | **USE CASE TYPE** |
| **USE CASE ID:** | View Requested Job | | Business Requirements: 🞎 |
| **PRIORITY:** | High | | System Analysis:🗹 |
| **SOURCE:** | Minions – Requirements List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Client | | |
| **PRIMARY SYSTEM ACTOR** | None | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * None | | |
| **DESCRIPTION:** | This use case describes the event in which a client wishes to see information about a requested job that already exists on the **‘Job Table’**.  He/she might want to check if his/her task(s) has (have) been allocated to (a) minion(s) already, if yes which one(s) or to check if the job is actually what they requested in the first place. | | |
| **PRE-CONDITION:** | The Client is logged in the website. | | |
| **TRIGGER:** | The Client wants to view requested tasks. | | |
| **TYPICAL COURSE** | Actor Action | System Response | |
| **OF EVENTS:** | **Step 1**: The Client selects to View Requested Task on the System. | **Step 2**: The system retrieves information from the ‘**Job Table’.** | |
|  |  | **Step 3:** The System displays the details about all the task(s) requested by that particular client. Information about :   * Job status * Job details * Minion(s) allocated to task(s)   Will be displayed. | |
| **ALTERNATE COURSES:** |  | | |
|  | | |
| **CONCLUSION:** | The system displays the all correct details about the job requested by the client. | | |
| **POST-CONDITION:** | The Client has viewed the necessary information and exits the page. | | |
| **BUSINESS RULES** |  | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** |  | | |
| **ASSUMPTIONS:** |  | | |
| **OPEN ISSUES:** |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | 3.8 | | **USE CASE TYPE** |
| **USE CASE ID:** | Rate Minion | | Business Requirements: 🞎 |
| **PRIORITY:** | Medium | | System Analysis:🗹 |
| **SOURCE:** | Minions – Requirements List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Client | | |
| **PRIMARY SYSTEM ACTOR** | None | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * None | | |
| **DESCRIPTION:** | This use case describes the event in which Clients provides their input regarding the service that has been provided to them.  The Client will rate the Minion using a 5 star rating scale and provide additional details if he/she wishes. | | |
| **PRE-CONDITION:** | The job status from the ‘**Job Table’** is updated to ‘completed’**.**  The Client is logged in. | | |
| **TRIGGER:** | The Client wants to provide feedback. | | |
| **TYPICAL COURSE** | Actor Action | System Response | |
| **OF EVENTS:** | **Step 1:** The Client selects to Provide Feedback from the System. | **Step 2:** The System prompts the Client to rate the quality of the job performed using the 5 star rating scale and put in additional comments. | |
|  | **Step 3:** The Client fills in the 5 star rating scale according to his satisfaction level and writes a brief review on the minion. | **Step 4:** The system captures this information and saves it on the **‘Minion Table’**. | |
|  |  |  | |
|  |  |  | |
| **ALTERNATE COURSES:** |  | | |
|  |  | | |
| **CONCLUSION:** | The system concludes when the Minion rating has been captured and the Client exits the system. | | |
| **POST-CONDITION:** | All the ratings are saved on the **‘Minion Rating Table’**. | | |
| **BUSINESS RULES** | * All the clients have the obligation to rate the minion(s) that performed the task(s) in order to improve service offered. | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** | * Not every Client might want to rate minions * This use case should be enabled only when the job is completed according to the status update | | |
| **ASSUMPTIONS:** | * None | | |
| **OPEN ISSUES:** | * None | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | 4.1 | | **USE CASE TYPE** |
| **USE CASE ID:** | Sign Up | | Business Requirements: 🞎 |
| **PRIORITY:** | Medium | | System Analysis:🗹 |
| **SOURCE:** | Minions – Requirements List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Applicant | | |
| **PRIMARY SYSTEM ACTOR** | Operational Head | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * Operational Head | | |
| **DESCRIPTION:** | This use case describes the event in which a first time applicant wishes to become a Minion. They will browse for the Minion system and fill in an application form or physically submit their application form at the company’s office where the Operational Head will capture the data into the system.  By Minion here, we refer to any individuals \_usually students \_that are willing to perform certain tasks, given some remuneration. They are considered as independent contractors who are external to the company. | | |
| **PRE-CONDITION:** | The prospective Minion shouldn’t exist in the system already. | | |
| **TRIGGER:** | The prospective minion wishes to sign up. | | |
| **TYPICAL COURSE** | Actor Action | System Response | |
| **OF EVENTS:** | **Step 1**: The applicant opens selects to Sign Up. | **Step 2:** The system prompts the applicant to enter his personal details, such as:   * ID number/Passport Number * Name/Surname * Physical address * Email Address * Social Media Profile(s) * Driving License if any | |
|  | **Step 3:** The applicant will fill in his personal details and submit the application form. | **Step 4:** The system validates if all information has been entered correctly. | |
|  |  | **Step 5:** If yes, the System captures all his/her details and saves them in the **‘Application Table’**. | |
|  |  | **Step 6:** The System notifies the applicant that application has been received successfully and a temporary password is provided to them. | |
|  |  |  | |
|  |  |  | |
| **ALTERNATE COURSES:** | **Alt Step 1:** The applicant physically goes and submits application forms at the company premises. | | |
| **Alt Step 2:** The Operational Head captures the applicant details onto the system and it saves it directly in the **‘Application Table’**. | | |
| **Alt Step 4:** Information has not been entered correctly and system prompts applicant to re-enter correct details. | | |
| **CONCLUSION:** | The system concludes when applicant has been successfully added on the database. | | |
| **POST-CONDITION:** | A notification has been sent to the applicant to confirm successful application. | | |
| **BUSINESS RULES** |  | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** |  | | |
| **ASSUMPTIONS:** |  | | |
| **OPEN ISSUES:** |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Check Application Status | | **USE CASE TYPE** |
| **USE CASE ID:** | 4.2 | | Business Requirements: 🞎 |
| **PRIORITY:** | High | | System Analysis:🗹 |
| **SOURCE:** | Minions – Requirement List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Applicant | | |
| **PRIMARY SYSTEM ACTOR** | None | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * None | | |
| **DESCRIPTION:** | This use case describes the event in which an applicant would like to see whether his/her application has been successful or not.  The Minion to be, will either log into the system using the temporary login details provided to him/her to view the application status, or will call in instead the Operational Head who will check the status for him/her. | | |
| **PRE-CONDITION:** | The Minions must have applied online or physically and went through an interview. | | |
| **TRIGGER:** | The Minion wants to check his/her application status. | | |
| **TYPICAL COURSE** | Actor Action | System Response | |
| **OF EVENTS:** | **Step 1**: The Applicant logs in to the system and selects to View his/her Application Status. | **Step 2:** The system retrieves information from the **‘Application Table’.** | |
|  |  | **Step 3:** The System displays the result corresponding to that minion. The possible statuses displayable are :   * Pending * Accepted * Rejected | |
|  |  |  | |
| **ALTERNATE COURSES:** | **Alt Step 1:** The Applicant calls in and provide his application number | | |
|  | **Alt Step 2:** The Operational Head will log in and retrieve the status corresponding to that application number from the status. | | |
|  |  | | |
|  |  | | |
|  |  | | |
| **CONCLUSION:** | The Use Case concludes when the Minion has viewed/been told his application status. | | |
| **POST-CONDITION:** | * The Minion has successfully viewed his status. | | |
| **BUSINESS RULES** | * None | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** |  | | |
| **ASSUMPTIONS:** | * None | | |
| **OPEN ISSUES:** | * None | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Add Working Schedule | | **USE CASE TYPE** |
| **USE CASE ID:** | 4.3 | | Business Requirements: 🞎 |
| **PRIORITY:** | High | | System Analysis:🗹 |
| **SOURCE:** | Minions – Requirement List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Minion | | |
| **PRIMARY SYSTEM ACTOR** | None | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * Operational Head | | |
| **DESCRIPTION:** | The Minion is requested to log the days and hours he/she would be available to perform tasks into the system, after he/she has been accepted as a Minion.  This use case describes the event of how Minion will go about adding their timetable onto the system. | | |
| **PRE-CONDITION:** | The Minion is logged in the System and must have been accepted. | | |
| **TRIGGER:** | The Minion wants to add his availability timetable. | | |
| **TYPICAL COURSE** | Actor Action | System Response | |
| **OF EVENTS:** | **Step 1**: The Minion selects to add a schedule. | **Step 2**: The system prompts the Minion to fill in his availability hours on the 7 day timetable, fill in a small biography about him/herself and select which task(s) he is interested in. | |
|  | **Step 3**: The Minion logs in his/her hours, completes the small biography about him/herself and selects which task(s) he is interested in. | **Step 4:** The system will save this information in the **‘Schedule Table’** and send a notification to the Minion about the changes made. | |
|  |  |  | |
| **ALTERNATE COURSES:** |  | | |
| **CONCLUSION:** | The use case concludes when the Minion has added his timetable and exits the page. | | |
| **POST-CONDITION:** | * The system has successfully saved the information and transaction appears on Audit Trail. | | |
| **BUSINESS RULES** | * None | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** | * None | | |
| **ASSUMPTIONS:** | * None | | |
| **OPEN ISSUES:** | * None | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Maintain Working Schedule | | **USE CASE TYPE** |
| **USE CASE ID:** | 4.4 | | Business Requirements: 🞎 |
| **PRIORITY:** | Medium | | System Analysis:🗹 |
| **SOURCE:** | Minions – Requirement List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Minion | | |
| **PRIMARY SYSTEM ACTOR** | None | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * Operational Head | | |
| **DESCRIPTION:** | This use case describes the event in which a Minion logs in to the website to make some changes to his/her timetable, or the task(s) he/she is interested in. | | |
| **PRE-CONDITION:** | The Minion is logged in the System and must have added a working schedule prior. | | |
| **TRIGGER:** | The Minion selects to maintain his working schedule. | | |
| **TYPICAL COURSE**  **OF EVENTS:** | Actor Action | System Response | |
| **Step 1**: The Minion selects to Maintain Working Schedule from the Homepage | **Step 2**: The System will retrieve the schedule details from the **‘Schedule Table’.** | |
|  | **Step 3:** The System will display the current Minion schedule, prompting him/her to update the fields he/she wants to. | |
| **Step 4**: The Minion will either increase or decrease his availability times and/or add or remove task(s) from his schedule. | **Step 5:** The System captures the new schedule details. | |
|  | **Step 6:** The System will update the **‘Schedule Table’.** | |
| **ALTERNATE COURSES:** |  | | |
|  | | |
| **CONCLUSION:** | The Use Case concludes when the Minion has updated his information and exits the page. | | |
| **POST-CONDITION:** | * The system updates the changes successfully in the **‘Schedule Table’**. | | |
| **BUSINESS RULES** | * These changes can only be made if the Minion doesn’t intend to change the dates and times a task has been assigned to him/her already. | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** | * None | | |
| **ASSUMPTIONS:** | * None | | |
| **OPEN ISSUES:** | * None | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Update Availability Status | | **USE CASE TYPE** |
| **USE CASE ID:** | 4.5 | | Business Requirements: 🞎 |
| **PRIORITY:** | Low | | System Analysis:🗹 |
| **SOURCE:** | Minions – Requirement List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Minion | | |
| **PRIMARY SYSTEM ACTOR** | None | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * Operational Head * Client | | |
| **DESCRIPTION:** | Besides uploading his timetable, a Minion has the opportunity to state whether he/she is available or not for a certain period of time. This would remove them from the list of available minions even if their schedule says otherwise.  This status is by default set to ‘available’ when the Minion is accepted.  This use case describes how the Minion will go about updating his/her availability status. | | |
| **PRE-CONDITION:** | The Minion is logged in the System and must not have any task assigned already. | | |
| **TRIGGER:** | The Minions wants to update availability status. | | |
| **TYPICAL COURSE** | Actor Action | System Response | |
| **OF EVENTS:** | **Step 1**: The Minion selects to Update Availability Status option from the homepage. | **Step 2:** The System retrieves the Minion’s availability status from the **‘Minion Table’** and requests the Minion to choose which status reflects his/her current situation. The statuses available for selection are:   * Available * Not Available | |
|  | **Step 3:** The Minion selects a corresponding status from the statuses list. | **Step 4:** The System checks if any task has been assigned to the current Minion. If not, the System proceeds and captures the Minion’s selection and saves it under the **‘Minion Table’**. | |
|  |  |  | |
| **ALTERNATE COURSES:** | **Alt Step 4:** The System realizes a task has been assigned to the Minion already and returns an Error Message preventing him to change his status. | | |
| **CONCLUSION:** | The Use Case concludes when the Minion has changed his status given that he was able to and he/she exits the page or when an error message is sent to notify him/her that the action cannot be processed | | |
| **POST-CONDITION:** | The Minion’s status has been saved on to the system. | | |
| **BUSINESS RULES** | * Availability Status can only be changed when there is no task assigned to the Minion already. | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** |  | | |
| **ASSUMPTIONS:** | * None | | |
| **OPEN ISSUES:** | * None | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | View Assigned Task Request | | **USE CASE TYPE** |
| **USE CASE ID:** | 4.6 | | Business Requirements: 🞎 |
| **PRIORITY:** | Medium | | System Analysis:🗹 |
| **SOURCE:** | Minions – Requirement List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Minion | | |
| **PRIMARY SYSTEM ACTOR** | None | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * None | | |
| **DESCRIPTION:** | After the Operational Head has assigned the task(s) to all available minions, each one of them can individually view what the task(s) is exactly about.  This use case describes the event of how a Minion goes about viewing all those tasks that have been assigned to him/her. | | |
| **PRE-CONDITION:** | The Minion is logged in the System and a task must have been assigned. | | |
| **TRIGGER:** | The Minion wants to view the task assigned to him. | | |
| **TYPICAL COURSE** | Actor Action | System Response | |
| **OF EVENTS:** | **Step 1**: The Minion selects to View Assigned Task from the Menu. | **Step 2**: The System retrieves information from the **‘Task Request Table’** and displays it according to the Minion’s profile. | |
|  |  |  | |
|  |  |  | |
| **ALTERNATE COURSES:** |  | | |
|  |  | | |
| **CONCLUSION:** | The Use Case concludes when the Minion has viewed the details about the task(s) assigned to him and he either exits the page or proceeds to 4.7. | | |
| **POST-CONDITION:** | * The Minion has successfully viewed what task(s) is (are) being assigned to him/her. | | |
| **BUSINESS RULES** | * None | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** |  | | |
| **ASSUMPTIONS:** | * None | | |
| **OPEN ISSUES:** | * None | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Accept/Decline Task | | **USE CASE TYPE** |
| **USE CASE ID:** | 4.7 | | Business Requirements: 🞎 |
| **PRIORITY:** | High | | System Analysis:🗹 |
| **SOURCE:** | Minions – Requirement List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Minion | | |
| **PRIMARY SYSTEM ACTOR** | None | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * Operational Head | | |
| **DESCRIPTION:** | After receiving the task request from the Operational Head, a Minion is then required to make himself available for the task or not.  This use case describe the events of how a Minion will accept or decline any task request sent to him/her using the system. | | |
| **PRE-CONDITION:** | The Minion is logged in the System and must have viewed the task details before. | | |
| **TRIGGER:** | The Minion wants to accept or decline the task(s). | | |
| **TYPICAL COURSE**  **OF EVENTS:** | Actor Action | System Response | |
| **Step 1**: The Minion selects to Accept the Task. | **Step 2**: The System will save the Minion’s selection and update the **‘Task Request Table’.** | |
|  | **Step 3:** The System will notify the Operational Head that a task has been accepted. | |
|  |  | |
|  |  | |
| **ALTERNATE COURSES:** | **Alt step 1**: The Minion selects to Decline Task. | | |
| **Alt Step 3:** The System will notify the Operational Head that a task has been declined. | | |
| **CONCLUSION:** | The use case concludes when the minion has either accepted or rejected a task and wishes to exit the page. | | |
| **POST-CONDITION:** | The Minion’s selection has been saved and stored for the Operational Head to decide on which Minion to pick. | | |
| **BUSINESS RULES** | * Once Accepted, a task cannot be cancelled * After a task request has been sent, Minion have 24h to either accept or decline | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** |  | | |
| **ASSUMPTIONS:** | * None | | |
| **OPEN ISSUES:** | * None | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Update Assigned Job Status | | **USE CASE TYPE** |
| **USE CASE ID:** | 4.8 | | Business Requirements: 🞎 |
| **PRIORITY:** | High | | System Analysis:🗹 |
| **SOURCE:** | Minions – Requirement List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Minion | | |
| **PRIMARY SYSTEM ACTOR** | None | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * Operational Head | | |
| **DESCRIPTION:** | The Minion is required to keep the task status up to date every time a change occurs. For instance, he/she is required to say if the task is started, in progress or completed.  This use case describes the event of how a Minion updates the task status of any task(s) assigned to him/her. | | |
| **PRE-CONDITION:** | The Minion must have been assigned to the actual task he is updating the status for and logged into the system. | | |
| **TRIGGER:** | The Minion wants to update the task(s) statuses. | | |
| **TYPICAL COURSE** | Actor Action | System Response | |
| **OF EVENTS:** | **Step 1**: The Minion selects to Update the Task Status. | **Step 2**: The System retrieves information from the ‘**Job Table’** and displays (all) the task(s) assigned to the Minion and prompts him/her to select which task he/she wants to update the status of**.** | |
|  | **Step 3:** The Minion selects the Task which requires an update. | **Step 4:** The System prompts the Minion to choose from the status list which one is relevant to the actual state of the task. The list contains the following possible statuses:   * Started * In Progress * Completed | |
|  | **Step 5:** The Minion selects a status from the status list | **Step 6:** The system saves the Minion’s selection and updates the **‘Job Table’**. | |
|  |  |  | |
| **ALTERNATE COURSES:** |  | | |
|  |  | | |
| **CONCLUSION:** | The use case concludes when the minion has updated the task status with its appropriate state and wishes to exit the page. | | |
| **POST-CONDITION:** | The Task status has been successfully updated and saved onto the **‘Job Table’**. | | |
| **BUSINESS RULES** | * A Minion is deemed to be truthful about the task status * Acting otherwise will see the Minion go through a disciplinary process which could lead to his/her eviction from the company. | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** |  | | |
| **ASSUMPTIONS:** | * None | | |
| **OPEN ISSUES:** | * The task might have the status ‘Completed’ but the Client is still unsatisfied about the job performed. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | 4.9 | | **USE CASE TYPE** |
| **USE CASE ID:** | Add Feedback on Completed Task(s) | | Business Requirements: 🞎 |
| **PRIORITY:** | High | | System Analysis:🗹 |
| **SOURCE:** | Minions – Requirements List | | System Design: 🞎 |
| **PRIMARY BUSINESS ACTOR** | Minion | | |
| **PRIMARY SYSTEM ACTOR** | None | | |
| **OTHER PARTICIPATING ACTORS:** | * None | | |
| **OTHER INTERESTED STAKEHOLDERS:** | * Operational Head | | |
| **DESCRIPTION:** | This use case describes the event in which a Minion adds his/her own impression about the task(s) that he has performed. This information might serve the Company to know who their Client are and to protect their Minions. | | |
| **PRE-CONDITION:** | The task has been completed an Minion is logged in. | | |
| **TRIGGER:** | The Minion wants to add feedback on task(s). | | |
| **TYPICAL COURSE** | Actor Action | System Response | |
| **OF EVENTS:** | **Step 1**: The Minion selects to Add Feedback. | **Step 2:** The System prompts the Minion to Add his impression. | |
|  | **Step 3:** The Minion types in his feedback and submits it. | **Step 4:** The System saves the feedback and updates the **‘Job Table’**. | |
|  |  |  | |
| **ALTERNATE COURSES:** |  | | |
|  |  | | |
| **CONCLUSION:** | The Use Case concludes on submission of feedback by the Minion. | | |
| **POST-CONDITION:** | The **‘Job Table’** is updated with the Minion’s feedback. | | |
| **BUSINESS RULES** |  | | |
|  |  | | |
|  |  | | |
|  |  | | |
|  |  | | |
|  |  | | |
| **IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS** |  | | |
| **ASSUMPTIONS:** |  | | |
| **OPEN ISSUES:** |  | | |