icketing flow + look for other ticketing systems for functionality

DeepSite to look for examples for ticketing system

Try to separate the flow for post life system (maintenance) and stuff

**Jira and Bugzilla for issue ticketing.**

**Unit tests are by dev side. Test scenario is for UAT and SIT**

**Login Flow (all users):**

The user shall be able to Log into their account with their assigned role using email and password.

The system shall notify the user if the user’s credentials are not matching the database.

The system shall display the Home page once the user has correctly entered the sign in credentials.

**User View after login flow (all users):**

The System shall display all the Latest projects from the Home Page with a greeting message to the PM immediately after login.

The PM shall be able to choose the projects that he wants to navigate to in order to see the project details.

**User After choosing a project flow (all users):**

The system shall display a method for the PM to switch between different projects.

The system shall display a navigational tab, including Overview, Activity, Issues, New Issues, Gantt, Calendar, Documents, and Settings.

The system shall display the Overview tab by default after PM selects a project after signing in:

* Overview displays the members within a project as well as their roles within the project

**User choosing the activity tab:**

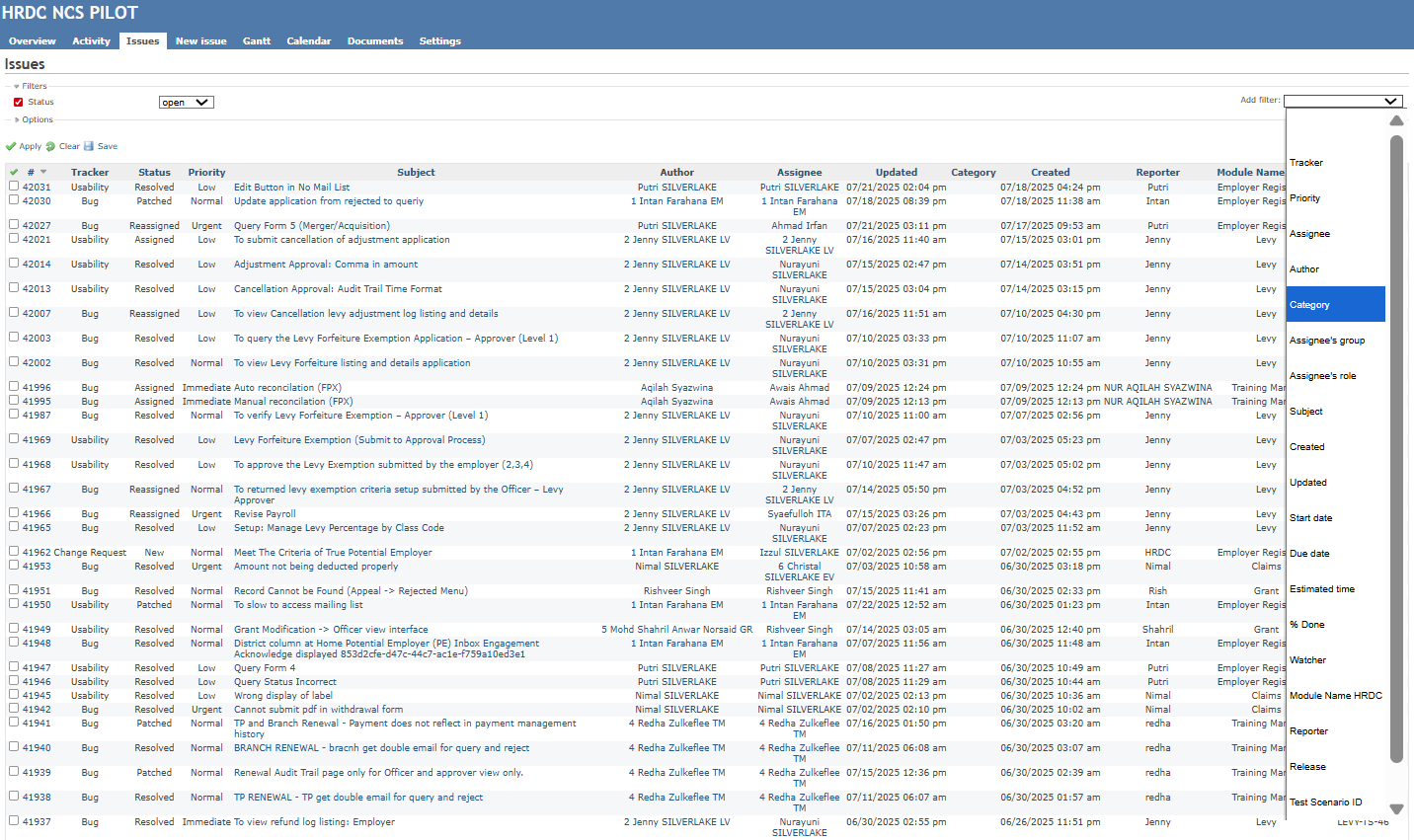
The system shall display all the tickets starting from Newest to oldest updates, each shows date, tracker type (bug, usability, change request), ticket id, status (patched, assigned, reassigned, closed etc), subject, environment patched in

**User Choosing the issues tab:**

The system will display the option to choose your filters such as Tracker, Priority, Assignee, Author, Category, Assignee's group, Assignee's role, Subject, Created, Updated, Start date, Due date, Estimated time, % Done, Watcher, Module Name HRDC, Reporter.

* + After setting the options to choose the filters, the user can select clear / apply / save.
    - Clear: clears all the filters available
    - Apply: applies filters to search for issue tickets
    - Save: save a custom filter style that will replace the current filter style where it defaults to only showing status open filters.

The system will display the table for viewing all of the issues with columns that can be edited to display whatever information the user wants including Tracker, priority, status, % done, Subject, author, assignee, updated, category, reporter, module name, test scenario ID, project, parent task, target version, start date (can be changed, so created date shows actual creation date of the ticket) , due date, estimated time, created (initial creation of ticket), spent time, reviewed by, release.

* Issue screen Christal suggestion:
  + You need to do control click to select multiple filters so that you can apply to not show / show stuff in the drop down tab (maybe you can use tick box instead)
  + Maybe display using hamburger tab instead of having all the filters in dropdown.
  + After that can export all the issues as pdf, csv etc.
  + 

**User clicking the ticket from the Activity, Issues tab, calendar tab, gantt tab.**

The system shall display the ticket ID and title, e.g., ug #41062 - Jompay Force Reconcile not call upfront payment amount.

The system shall display who created the ticket, including the name, role, and time of creation (e.g., “Added by 7 Nik Mahirah Zainal CR 3 months ago”).

The system shall display when the ticket was last updated, e.g., “Updated 34 minutes ago”.

The system shall show the current status of the ticket, e.g., Reassigned.

The system shall show the start date and due date of the ticket.

The system shall show the current assignee (which means at the end when the bug has been rechecked by the QA and confirmed to be fixed, then it will be shown as the reporter’s name, cuz the reporter will send to guy to fix, and will be the last double check to make sure it’s correct) and allow reassignment of the ticket.

The system shall display the priority level of the ticket.

The system shall display the percentage of completion (e.g., % Done: 100%).

The system shall display the following optional metadata fields:

- Category

- Spent Time

- Target Version

- Reporter

- Reviewed By

- Release

- Test Scenario ID

- Module Name (e.g., Course Registration)

The system shall allow for related issues to be added onto the current ticket.

The system shall display the history of assignee and status and description of the ticket, as well as what environment has been changed, and what is the % completion.

**User Choosing the New Issue tab to create a new ticket for an issue:**

The system shall allow the user to create a new ticket based on:

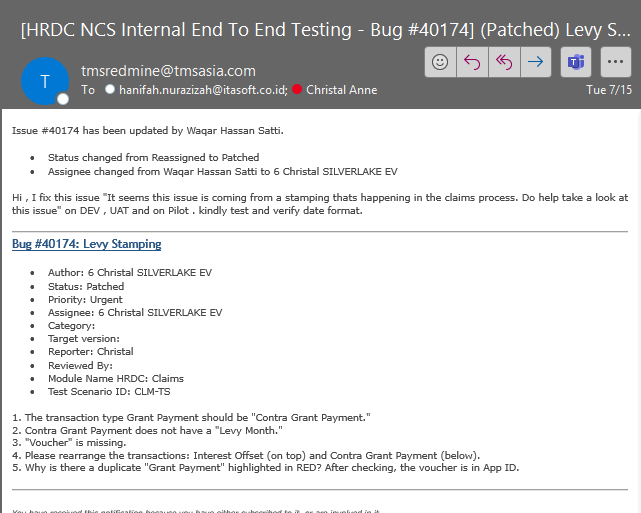
* Tracker
* Subject (of the issue)
* Description (additional text, including:
  + Issue
  + Steps to replicate issue
  + Credentials (which account used to find the bug eg. Officer / employer)
  + Environment (eg. Pilot, dev, uat <they are like branches for testing>)
  + Expected behavior (original function)
* Status:
  + Assigned: send to developer
  + Patched: Dev fix issue and assign back to QA to verify the bug
  + Reassigned: I think it just means you assigned to someone second time (meaning the qa checked after the first patch, still got issue, and reassigned back to the prev dev that “solved” the issue)
  + Resolved: QA will resolve the ticket if the bug is fixed.
  + Closed: irrelevant / tested by user and then no problem
  + Rejected: if the ticket is irrelevant
  + Add On Hold / to be dealt with later, for low priority ticket for developers to work on multiple tickets, they can put one on hold to work on another higher priority ticket.
  + (add an in progress status for developers to show that they are working on it.)

|  |
| --- |
| **Standard Ticket Status Meanings**   1. **New** → The issue or ticket has just been created. No one has acted on it yet. 2. **Assigned** → The ticket has been assigned to a specific person (usually a developer or team) to investigate or fix. 3. **Reassigned**    * The ticket has been transferred from one assignee to another because unavailable / left the team / outside their scope / better for another person or team / task moved between teams (from qa to dev, from frontend to backend) 4. **Accepted** → The assignee has acknowledged the ticket and agreed to work on it. 5. **Resolved** → The issue has been addressed or fixed by the developer, but it has **not yet been verified** by QA/testers. 6. **Patched** → A fix (patch) has been made and possibly deployed to a test environment. This often comes between *Resolved* and *Verified*. 7. **Verified** → QA/testers have tested the patch or fix, and confirmed that the issue is resolved. 8. **To Be Dealt With Later** → The issue is **not urgent or important** right now. It's been deferred to a later phase or backlog. 9. **Closed** → The full lifecycle of the ticket is complete. It’s been resolved and verified, and no further action is needed. 10. **Rejected** → The issue has been reviewed and **declined**, either because it's not a bug, not reproducible, invalid, or out of scope. |

* Priority:
  + Low: can still proceed (typo in non-customer facing message)
  + Normal: can still proceed: eg email template issues, misaligned UI elements (can cause inconvenience, but does not block key functionality)
  + Urgent: impacts key functionality or business flow, but workaround may be possible (bug in payment validation, incorrect user role assignment)
  + Immediate: like can’t proceed to next step (login failure, system crash, thing blocks deployment)
* Start Date:
  + When you create the ticket
  + Can also be configurable
* End Date:
  + Can still be edited even after SLA
* Estimated time: (DEV)
  + To key in how much time it takes to fix the ticket
* % done: (DEV)
  + Reassign will be 100% done.
* Test scenario id:
  + Follow the test scenario google sheet one
* Category: (can customize) add new category
  + Regarding slowness
  + Regarding some approval process
* Reporter:
  + Who raised the ticket.
* Module Name:
  + Which module the issue came from
* Files
  + Attach videos etc. to show how the issue is created
  + What Christal don’t like for redmine – she can only put files one by one, she would like to add like 5 pics at the same time etc.
  + Can also give description for each file.
* Watcher:
  + Add people that get notified when there are updates on the issue such as each time the ticket is edited.
* Release:

**Editing Ticket History:**

Eg:

* + Time stamp when what time how many days ago
  + Updated by developer
  + Status
  + Assignee
  + Percent done
  + Comments:
    - What has been patched
    - Assign back to QA
  + QA check again, then assign back to dev to patch it to the environment if no issues
  + Notification:
  + Email to the assignee
  + 
  + (add, maybe do notification in inbox, like maybe in the hrdc ncs one), inform someone else about updates
  + (add, tagging someone in a ticket comment)

|  |
| --- |
| SLA:  Resolve issue with a deadline (timeframe for different priority)  Defining the timeframe based on client  Urgent / Immediate: severity 1, acknowledge the issue within 15 minutes (make yourself known to the customer that you will solve the issue  SLA can also change over time. (make SLA configurable Customizable)  Eg: Severity 1 is 4 hours (can change with diff company)  Eg: Severity 2 is 44 hours |

**User selects the gantt tab to view the gantt chart of all issues and how they are progressing.**

The system shall display the issues and their completion rate as well as the percentage completion of the total of all issues averaged.

The system shall allow the user to add actions to filter for, such as Tracker, Priority, Assignee, Author, Category, Assignee's group, Assignee's role, Subject, Created, Updated, Start date, Due date, Estimated time, % Done, Watcher, Module Name in Project, Reporter, Release, Test Scenario ID, Reviewed By

The system shall display the issues filtered after applying the filter, displaying by each month of the year and numbering the weeks by the number of weeks in the year and should be able to navigate between previous and future months.

Currently there is no added due date for any of the tickets. May be able to implement SLA’s based on the company and then automatically assign to tickets their due date based on the priority of each ticket

**The user selects the Calendar tab**

The system shall display all the tickets that have the status of open by default (which includes every status other than closed and rejected) based on the day that the ticket was created.

The system shall allow users to apply filters such as Tracker, Priority, Assignee, Author, Category, Assignee's group, Assignee's role, Subject, Created, Updated, Start date, Due date, Estimated time, % Done, Watcher, Module Name in Project, Reporter, Release, Test Scenario ID, Reviewed By as well as choose the month of the year they would like to view the tickets.

**The user selects the Documents tab:**

Normally should display these:

Project specs (FSD, BRD, technical docs)

User manuals

Change request forms

Design diagrams

Test plans or test scripts

Meeting minutes

However current system one is empty, has no documents in the document page.

**The user selects the Settings tab (non PM users)**

**The PM selects the Settings tab of the project**

The system shall display the tabs for Information, Modules, Members, Versions, and Issue categories

**The PM selects the information tab**

The system shall allow the PM to edit the Name of the project

The system shall allow the PM to edit the description of the project

The system shall allow the PM to edit the identifier of the project (all lowercase with no spaces used as URLs and internal references to the project eg. <https://system.yourcompany.com/projects/maybank-ncs-regression>)

The system shall allow the PM to edit the Homepage section (for adding the url for the project’s official site.

The system shall allow for the PM to toggle for the project to be viewed by the public (i.e. anonymous users without any permissions)

The system shall allow for the PM to record the Warranty Expiry of the project as well as the maintenance expiry (warranty is free xxx months after delivery, maintenance is a paid service, if keep paying, it will keep continuing, usually covers general support, patches, and helpdesk)

The system shall allow for dropdown selector options for trackers in ticket creation to be displayed depending on which checkbox has been ticked by the PM. (eg.  Bug, Usability, Post FUAT Findings, FUAT Findings, Testing, Requirement, Change Request, Support, Documentation, Meeting, Internal Training, Not in FSD, Pending Feedback) [admin can add more options for trackers]

The system shall allow for custom fields to be added during ticket creation if ticked by the PM: Reporter, Stage of Discovery, Reviewed By, Reviewed Date, Sprint No., Phase OR CR, App Name, MTC Est. Done Date (estimated completion date for testing master test case) Module Name, Test Cycle, Release (software release version this test / module is tied to, eg. V1.0, release 3.2 etc.)

**The PM selects the Modules tab**

The system displays all the tabs that can be marked as a checkbox to display on the dashboard when a user chooses to view a project (including: Issue tracking, Time tracking, News, Documents, Files, Wiki, Repository, Boards, Calendar, Gantt)

**The PM selects the Members tab:**

The system shall display the user as well as the group and the role of the user for the project.

The system shall display all users registered to the system and allows the PM to assign role / multiple roles to the user.

The system shall allow the PM to edit / delete user roles from the users in the current project as well as to remove them completely from the project.

**The PM clicks onto the Versions tab:**

The system shall display the versions of the project (eg. v1.0, UAT cycle 2, release 2025-Q3 etc.)

The system shall display the status of the version (eg. Open, locked, closed)

The system shall display the start date, due date, % complete of the version (based on number of tasks closed), description, etc.

Flowchart, figma mockup,

--------------------------------------------------------------------

**Authentication & Access**

**UC001 Log in to the System**  
The user enters their credentials to access the ticketing system.

**UC002 Add New User Accounts**  
An admin creates accounts for new users with specific roles and permissions.

**Ticket Management**

**UC003 Create New Ticket**  
A user submits a new issue or request with necessary details.

**UC004 View List of Tickets**  
A user views all tickets assigned to them or available in the system.

**UC005 Search or Filter Tickets**  
A user searches for tickets or applies filters based on status, priority, category, or other attributes.

**UC006 View Ticket by Calendar**  
A user views tickets based on their scheduled dates or deadlines in a calendar format.

**UC007 View Ticket Details**  
A user opens a ticket to see full details, history, and updates.

**UC008 Edit Ticket Details**  
A user updates ticket information such as description, priority, or category.

**UC009 Assign Ticket**  
A project manager or admin assigns a ticket to a specific user or team.

**UC010 Resolve Ticket**  
A user updates a ticket’s status to resolved after completing the work.

**UC011 Add Template for Similar Issue Response**  
A user creates reusable response templates for recurring issues.

**Notification & Communication**

**UC012 View Notifications**  
A user views alerts for ticket assignments, updates, and SLA breaches.

**UC013 Send Notifications**  
The system or user sends alerts to relevant stakeholders about ticket changes or updates.

**Project & Knowledge Management**

**UC014 Create New Project**  
An admin or project manager sets up a new project in the system.

**UC015 Create New Subproject**  
An admin or project manager creates a subproject under an existing project.

**UC016 View Project Documents**  
A user views project-related documents stored in the system.

**UC017 View Knowledge Base**  
A user accesses the knowledge base for troubleshooting guides and FAQs.

**Configuration & Administration**

**UC018 Configure Project Settings**  
An admin updates project-specific configurations.

**UC019 Manage Project Members**  
An admin adds, removes, or changes roles of members in a project.

**UC020 Configure SLA Rules**  
An admin sets up service-level agreement rules for ticket resolution timelines.

**UC021 Set Automation Rules for Ticket Assignment**  
An admin configures automated ticket routing to specific agents based on criteria.

**UC022 Add New Skills**  
An admin updates the skill list for agents to aid in ticket assignment.

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Log in to the System | ID: UC001 | Importance Level: High |
| Primary Actor: User | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  User – All users of the system with a valid account shall be able to log in to the system in order to use it | | |
| Brief Description:  This use case describes about the Log in process into the system. | | |
| Trigger:  User Enters the homepage of the ticketing system and presses the Log in button. | | |
| Relationships:   * Association : User * Include : N/A * Extend : N/A * Generalization : N/A | | |
| Normal Flow of Events   1. The use case begins when the user navigates to the system’s login page. 2. The system displays the login form with fields for username (or email) and password. 3. The user enters their username and password into the respective fields. 4. The user clicks the “Login” button. 5. The system validates that both fields are filled in. 6. The system verifies the entered credentials against the stored user data. 7. The system determines that the credentials are valid. 8. The system grants access and displays the main dashboard according to the user’s role and permissions. | | |
| Sub-flows: | | |
| Alternate/Exceptional Flows:  E-5:   1. If either the username or password field is empty, the system displays an error message: “Please fill in all required fields.” 2. The user can re-enter the missing information and attempt to log in again in step 2.   E-6:   1. If the system detects that the account has been locked (for example, due to too many failed Login attempts or admin suspension), it displays a message: “Your account is locked. Please contact the system administrator.” 2. The user cannot proceed until the account is unlocked.   E-7:   1. If the username is not found or the password does not match, the system displays an error message: “Invalid username or password.” 2. The user can re-enter the correct credentials and continue back from step 4. | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Add New User Accounts | ID: UC002 | Importance Level: High |
| Primary Actor: Admin | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  Admin - Wants to add new users to the system efficiently allowing them to enter the system and also assign roles to them.  New User – Needs their account to be created correctly so that they can log in with their assigned roles.  Project Manager – Ensure that each person has their roles assigned correctly so that they don’t have permission to access places that they aren’t supposed to and edit important information | | |
| Brief Description:  This use case describes how the admin would add in a new user so that they can Login to the system. | | |
| Trigger:  The admin needs to register a new user in the system because a new staff member, agent, or client requires access. | | |
| Relationships:   * Association : Admin * Include : Manage Project Members * Extend : N/A * Generalization : N/A | | |
| Normal Flow of Events   1. The Admin chooses the **“**Add New User” option from the system’s user management menu. 2. The system displays the Add User form with required fields such as full name, username/email, password, role, and any other required details (e.g., department, phone number). 3. The Admin enters the new user’s details in the form fields. 4. The Admin selects the appropriate role (e.g. Project Manager, Agent, BA, QA ). 5. The Admin sets an initial password and selects the “Send Invite” option via email 6. The Admin clicks the Save or Create Account button. 7. The system validates that all required fields are filled in and checks that the username/email is unique. 8. The system creates the new account and stores the information in the database. 9. The system displays a confirmation message: *“User account created successfully.”* | | |
| Sub-flows:  S-1:   1. The Admin clicks Advanced Permissions. 2. The system displays a list of additional permissions (e.g., “Can edit SLA settings”, “Can delete tickets”). 3. The Admin selects any extra permissions required for the new user. | | |
| Alternate/Exceptional Flows:  E-7.1:   1. If one or more required fields are empty, the system displays an error message: “Please fill in all required fields.” 2. The Admin corrects the form and resubmits.   E-7.2:   1. If the entered username or email already exists in the system, the system displays an error message: “This username/email is already in use.” 2. The Admin enters a different username/email and resubmits.   E-8:   1. If there is a database or server error, the system displays an error message: “Unable to create account. Please try again later.” 2. The Admin may retry once the issue is resolved. | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Create New Ticket | ID: UC | Importance Level: High |
| Primary Actor: Admin | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  Users (BU, Agent, Employee, PM, Admin, QA, BA, Developer) – wants to create a ticket so that they can create a record of their issue to make sure it will be resolved later. | | |
| Brief Description:  This use case describes how the user will create a new ticket to report an issue. | | |
| Trigger:  The User needs to create a new ticket because there has been an issue that they encountered and they need to record that issue so that it can be solved later. | | |
| Relationships:   * Association : User * Include : Send Notifications * Extend : Assign Ticket * Generalization : N/A | | |
| Normal Flow of Events   1. The use case begins when the user selects the **“Create New Ticket”** option from the main dashboard of a project’s subproject. 2. The system displays the **New Ticket Form** with fields such as:    * + Tracker      + Subject      + Description      + Status      + Category      + Priority      + start date, end date, assignee, estimated time, %done      + test scenario id      + category      + reporter      + module name      + watchers      + attachments. 3. The user enters a tracker from a drop down menu (Bug, usability, change request) 4. The user enters a **s**ubject summarizing the issue. 5. The user provides a detailed **description** of the issue or request which includes: S5 6. The user selects a **category from a dropdown menu** (e.g., Slowness, Email template, etc). 7. The user sets the assignee of the ticket. 8. The user sets the **priority level** (e.g., Low, Medium, High, Urgent) 9. The user sets the Start and optionally sets the End date 10. The user optionally sets the estimated time to fix the ticket 11. The user optionally sets the % done of the ticket 12. The user sets the test scenario ID 13. The user sets the reporter (themselves, or a customer) 14. The user sets the module name 15. The user optionally attaches files or screenshots to support the issue. 16. The user optionally sets the watchers for the ticket. 17. The user clicks the **Submit** button. 18. The system validates that all required fields are filled in. 19. The system creates the new ticket and assigns it a **unique ticket ID**. 20. The system stores the ticket in the database and applies any relevant SLA or automation rules (e.g., auto-assigning to a support team). 21. The system displays a confirmation message: “Your ticket has been created successfully. Ticket ID: [XXXX].” | | |
| Sub-flows:  S-5   1. what is the issue 2. how to replicate the issue 3. Credentials for the account that encountered the issue 4. Environment where the issue occurred 5. Expected original function of the issue 6. Continue to 6. | | |
| Alternate/Exceptional Flows:  E-15   1. If the attached file exceeds the allowed size limit or has an unsupported format, the system rejects the file and displays an error message: “Attachment is invalid. Please upload a supported file type under the size limit.” 2. The user may remove or replace the attachment and try again.   E-18:   1. If one or more required fields (e.g., title, description, category) are empty, the system displays an error message: “Please complete all required fields.” 2. The user fills in the missing fields and resubmits.   E-19:   1. If the system fails to create the ticket due to a database or server error, the system displays a message: “An error occurred while creating your ticket. Please try again later.” 2. The user may attempt to resubmit once the system is available. | | |
| Additional Notes:   1. Required fields are:  * Subject * Description * Reporter  1. Business Users will only see and be able to edit the following fields:  * Subject * Description * Attachments * Module * Category  1. Users will not be able to create tickets for projects they are not assigned to (except for Admin) | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  View List of Tickets | ID: UC | Importance Level: High |
| Primary Actor: User | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  User – All users will be able to view the list of tickets that have already been created and haven’t been deleted for traceability purposes. | | |
| Brief Description:  This use case describes how the user can view the list of tickets that have been reported. The list provides an overview of existing tickets with relevant details such as ticket ID, subject, status, priority, reporter, assignee, module name, tracker etc. | | |
| Trigger:  After the User has been authenticated and selects a project, and then selects the page they will be able to view all the tickets. | | |
| Relationships:   * Association : User * Include : N/A * Extend : View Ticket Details * Generalization : N/A | | |
| Normal Flow of Events   1. The user selects a ticket from the list of tickets. 2. The system retrieves the details of the selected ticket from the database. 3. The system displays the ticket details, including:    * Ticket ID    * Tracker (bug, change request, etc.)    * Subject/Title    * Description    * Status    * Category    * Priority    * Start Date, End Date, Estimated Time, % Done    * Test Scenario ID    * Reporter    * Module Name    * Watchers    * Attachments 4. The user reviews the ticket details. | | |
| Sub-flows:  S-3:   1. User clicks on the attachments linked to the ticket 2. The system retrieves and opens and allows for downloading the attachment | | |
| Alternate/Exceptional Flows: | | |
| Additional Notes:   * Users are not able to see ticket details that are not within their assigned project (except for Admin) | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Search or Filter Tickets | ID: UC | Importance Level: High |
| Primary Actor: User | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  User – All users will be able to filter or search for a specific ticket as long as they are within their assigned project. (except for Admin) | | |
| Brief Description:  This use case describes how a user can search or apply filters to the list of tickets in order to quickly locate specific tickets. The system allows users to filter tickets by attributes such as status, category, priority, date range, or reporter. It will also allow for searching for specific words on fields like subject or description. The pages that can use filtering are: View Notification Inbox, View Tickets by Calendar, View List of tickets | | |
| Trigger:  After the User has been authenticated and selects a project, and then selects the environment and page they will be search or apply filters to locate specific tickets. | | |
| Relationships:   * Association : User * Include : N/A * Extend : View Ticket Details, View Ticket By Calendar, View Notification * Generalization : N/A | | |
| Normal Flow of Events   1. The user selects a ticket from the list of tickets. 2. The system retrieves the details of the selected ticket from the database. 3. The system displays the ticket details, including:    * Ticket ID    * Tracker (bug, change request, etc.)    * Subject/Title    * Description    * Status    * Category    * Priority    * Start Date, End Date, Estimated Time, % Done    * Test Scenario ID    * Reporter    * Module Name    * Watchers    * Attachments 4. The user reviews the ticket details. | | |
| Sub-flows:  S-3:   1. User clicks on the attachments linked to the ticket 2. The system retrieves and opens and allows for downloading the attachment | | |
| Alternate/Exceptional Flows: | | |
| Additional Notes:   * Users are not able to see ticket details that are not within their assigned project (except for Admin) | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  View List of Tickets | ID: UC | Importance Level: High |
| Primary Actor: User | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  User – All users will be able to view the list of tickets that have already been created and haven’t been deleted for traceability purposes. | | |
| Brief Description:  This use case describes how the user can view the list of tickets that have been reported. The list provides an overview of existing tickets with relevant details such as ticket ID, subject, status, priority, reporter, assignee, module name, tracker etc. | | |
| Trigger:  After the User has been authenticated and selects a project, and then selects the environment and the issue page they will be able to view all the tickets. | | |
| Relationships:   * Association : User * Include : N/A * Extend : View Ticket Details * Generalization : N/A | | |
| Normal Flow of Events   1. The user selects a ticket from the list of tickets. 2. The system retrieves the details of the selected ticket from the database. 3. The system displays the ticket details, including:    * Ticket ID    * Tracker (bug, change request, etc.)    * Subject/Title    * Description    * Status    * Category    * Priority    * Start Date, End Date, Estimated Time, % Done    * Test Scenario ID    * Reporter    * Module Name    * Watchers    * Attachments 4. The user reviews the ticket details. | | |
| Sub-flows:  S-3:   1. User clicks on the attachments linked to the ticket 2. The system retrieves and opens and allows for downloading the attachment | | |
| Alternate/Exceptional Flows: | | |
| Additional Notes:   * Users are not able to see ticket details that are not within their assigned project (except for Admin) | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  View Ticket by Calendar | ID: UC | Importance Level: High |
| Primary Actor: User | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  User – All users will be able to view ticket by calendar within their respective projects | | |
| Brief Description:  This use case describes how a user can View the ticket by calendar. The view will be a calendar and the user can see what day and date each ticket was created more clearly. | | |
| Trigger:  After the User has been authenticated and selects a project, and then selects the environment and calendar tab, then they can view the calendar and all the associated tickets on where they were created. | | |
| Relationships:   * Association : User * Include : N/A * Extend : View Ticket Details, Apply Filter / Search Tickets * Generalization : N/A | | |
| Normal Flow of Events   1. The user navigates to the Calendar tab. 2. The system displays a calendar view with dates highlighted where the user has reported tickets. | | |
| Sub-flows: | | |
| Alternate/Exceptional Flows:  A-2   1. User can apply filters (e.g., status, category, priority) from the calendar tab. 2. Continue back to Flow 2. | | |
| Additional Notes:   * Users are not able to see calendar ticket that are not within their assigned project (except for Admin) | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  View Ticket Details | ID: UC | Importance Level: High |
| Primary Actor: User | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  User – All users will be able to view ticket details within their respective projects | | |
| Brief Description:  This use case describes how a user can View Ticket details such as tracker, status, %done etc. These are all tickets of each individual ticket and you can see the entire history of the entire ticket as well, this includes changes being made to any field of any part of the ticket. There is also an option for you to leave comments and write solutions to the ticket after it has been solved. | | |
| Trigger:  After the User has been authenticated and selects a project, and then selects the environment and view ticket list tab, and click on a specific ticket. | | |
| Relationships:   * Association : User * Include : N/A * Extend : Resolve Ticket, Edit Ticket Details, Assign Ticket, Comment on ticket * Generalization : N/A | | |
| Normal Flow of Events   1. The user selects a ticket from the available list (or calendar). 2. The system retrieves the ticket information from the database. 3. The system displays the ticket details, including:    * + Tracker (bug, change request, etc.)      + Status      + % Done      + Category, Priority, Module name      + Start Date, End Date, Estimated Time      + Reporter, Watchers, Assignee.      + Test Scenario ID      + Attachments 4. The system also displays the full ticket history, showing changes made to fields over time. 5. The system also displays tabs within the ticket for comments and solution. 6. The user reviews the details and history of the ticket. | | |
| Sub-flows: | | |
| Alternate/Exceptional Flows: | | |
| Additional Notes: | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Edit Ticket Details | ID: UC | Importance Level: High |
| Primary Actor: User | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  User – All users will be able to view ticket details to edit within their respective projects, unless certain permissions are set to disable them from editing certain fields. | | |
| Brief Description:  This use case describes how a user can View Ticket details such as tracker, status, %done etc. These are all tickets of each individual ticket and you can see the entire history of the entire ticket as well, this includes changes being made to any field of any part of the ticket. There is also an option for you to leave comments and write solutions to the ticket after it has been solved. | | |
| Trigger:  After the User has been authenticated > and selects a project > and then selects the environment > and view ticket list tab > and click on a specific ticket > then they can edit the ticket details. | | |
| Relationships:   * Association : User * Include : N/A * Extend : Resolve Ticket, Edit Ticket Details, Assign Ticket, Comment on ticket * Generalization : N/A | | |
| Normal Flow of Events:   1. The system displays the fields for the ticket.  * Tracker (bug, change request, etc.) * Status * % Done * Category, Priority, Module name * Start Date, End Date, Estimated Time * Reporter * Test Scenario ID * Attachments  1. The user updates one or more of the available fields. 2. The user confirms and submits the changes. 3. The system validates the new values (e.g., dates are valid, required fields are filled). 4. The system saves the updated ticket details into the database. 5. The system updates the ticket history log to record the changes. 6. The system refreshes and displays the updated ticket details to the user | | |
| Sub-flows: | | |
| Alternate/Exceptional Flows:  E-4   1. If the user enters invalid or incomplete data, the system displays an error message. 2. The user corrects the data and resubmits.   A-3   1. The user cancels before saving changes. 2. The system discards edits and returns to the ticket details view. | | |
| Additional Notes:   1. Users can only Edit Ticket Details if within their own project 2. Some users like BU can be given the permission to not be able to edit certain / all field details, but any user can be prevented from editing any field from settings. | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Assign Ticket | ID: UC | Importance Level: High |
| Primary Actor:  User | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  QA / BA (UAT Team) – Assigns tickets to developers after testing or validation.  PM (Project Manager) – Oversees assignments, balances workload, and ensures progress.  Maintenance Staff / Admins – Assign tickets related to system maintenance, infrastructure, or urgent fixes.  Developers – Receive tickets assigned to them and perform tasks.  Reporter / BU – May be notified of assignment for transparency. | | |
| Brief Description:  This use case describes how an authorized user to assign tickets to the appropriate staff or team to resolve the issue. It helps keep track and the completion of internal project tasks and maintenance tasks are notified properly. | | |
| Trigger:   * 1. A new ticket is created (internal or BU reported) and needs assignment.   2. A ticket needs reassignment due to workload, bug fix, priority change, or escalation. | | |
| Relationships:   * Association : User * Include : Send Notifications * Extend : N/A * Generalization : N/A | | |
| Normal Flow of Events:   1. User selects a ticket that needs assignment. 2. User chooses a user or team as the assignee. 3. System validates the assignee’s role and availability. 4. System updates the ticket record with the assigned staff member/team. 5. System sends a notification to the assignee. 6. Assigning user receives confirmation of the assignment. | | |
| Sub-flows: | | |
| Alternate/Exceptional Flows:  A-1:   1. System automatically assigns tickets based on rules (priority, workload, expertise). 2. Human actors (QA/BA, Agent, PM) can override if necessary. | | |
| Additional Notes   * + 1. BU are not allowed to assign tickets unless given permission to do so. | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Resolve Ticket | ID: UC | Importance Level: High |
| Primary Actor:  Agent, BA, QA, PM, Admin | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  Agent / QA / BA - Documents how the issue was resolved, attaches screenshots or files, and closes the ticket.  Developer - Indirect interest, their fix is recorded as the resolution.  PM - Wants visibility of final closure and historical reference for similar issues.  BU - Wants confirmation that the problem has been fixed and clarity on how it was solved. | | |
| Brief Description:  This use case describes how authorized staff (Agent, QA/BA, or PM) provide resolution details for a completed issue. The actor enters notes, attaches supporting evidence (e.g., screenshots, logs), and updates the ticket status to Closed. They may also choose to create a template to be stored within the knowledge base of the project which can help solve similar issues in the future. | | |
| Trigger:  Ticket is confirmed fixed (developer’s work + QA retesting completed). The responsible staff member is ready to document the resolution and close the ticket.  Or if there is already a similar issue that happened before, the user can take the template from the knowledge base and copy it to the resolution tab of the ticket and close it. | | |
| Relationships:   * Association : User * Include : Send notifications. * Extend : Adding Template For Similar Issue * Generalization : N/A | | |
| Normal Flow of Events:   1. User (Agent/QA/BA/PM/Admin) opens a ticket that is marked as “Resolved” 2. User navigates to the Resolution tab. 3. User enters resolution details like description of how the issue was fixed and steps taken. 4. User attaches any supporting files like screenshots, logs, documentation or videos. 5. User updates the ticket status to Closed. 6. The system saves resolution details and stores attachments. 7. The system notifies the client (and relevant stakeholders) that the issue has been resolved and closed. | | |
| Sub-flows: | | |
| Alternate/Exceptional Flows: | | |
| Additional Notes:   * 1. A BU is not allowed to enter a resolution or close the tickets after resolved. Instead they can comment on the ticket that the issue has already been fixed, and a person who has the authorization will review it and close the ticket.   2. A BU may reopen the ticket if they are still encountering the recurring issue | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Add Template for Similar Issue Response | ID: UC | Importance Level: High |
| Primary Actor:  Agent / BA / QA / PM / Admin | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  Agent - Main user who creates templates to speed up responses to recurring issues.  QA / BA - Can create or validate templates based on common scenarios for technical correctness and usability of template content.  Project Manager (PM) - Oversees the use of templates, ensuring they are standardized and align with company policy.  Developers - Not users, but benefits because fewer unnecessary repetitive tickets come their way if agents can resolve common issues with templates.  BU - Benefits by receiving fast and accurate responses. | | |
| Brief Description:  This use case describes how to write a new template for common issues to be stored in the knowledge base. | | |
| Trigger:   1. When the ticket has been resolved and the authorized user can write a template tied to a ticket that will be stored into the knowledge base. 2. When the authorized user wants to write a new note for the knowledge base. | | |
| Relationships:   * Association : User * Include : N/A * Extend : N/A * Generalization : N/A | | |
| Normal Flow of Events:  Path A: Add Template Directly to Knowledge Base   1. User (Agent / QA / BA / PM) logs into the ticketing system. 2. User navigates to the Knowledge Base / Template Management section. 3. User clicks “Add New Template.” 4. The system displays the Add Template form. 5. User enters template details such as title, content, category, display permission, tags, meta keywords separated by commas, meta description and attachments if needed. 6. User saves the template. 7. The system validates input and stores it in the knowledge base. 8. The system confirms successful creation with a message: “The template has been created successfully, Title: XXX”. 9. The new template is available for reuse when responding to similar issues.   Path B: Create Template After Resolving a Ticket   1. User resolves a ticket and writes a resolution (steps taken, screenshots, notes). 2. Before closing the ticket, the actor selects “Save as Template.” 3. The system copies the resolution details into a New Template Form. 4. User edits the text, adds categories, display permission, tags, meta keywords separated by commas, and meta description attaches supporting files. 5. User saves the template. 6. The system validates input and stores the template in the knowledge base. 7. The system confirms successful creation with a message: “The template has been created successfully, Title: XXX”. 8. The template is now linked to the closed ticket (for reference) and available for reuse. | | |
| Sub-flows: | | |
| Alternate/Exceptional Flows: | | |
| Additional Notes: | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  View Notification Inbox | ID: UC | Importance Level: High |
| Primary Actor:  User | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  Agent / PM / BA / Developer / QA - receives notifications when assigned or watched tickets are updated.  Client - receives notifications on updates to their submitted tickets.  Administrator - manages notification rules and settings. | | |
| Brief Description:  This use case describes how users can view notifications whenever a ticket is created or updated. Users can view these notifications in their inbox. | | |
| Trigger:  A ticket is created or edited (status change, comment added, reassignment, etc.). | | |
| Relationships:   * Association : User * Include : N/A * Extend : View Ticket Details, Search / Apply Filter tickets * Generalization : N/A | | |
| Normal Flow of Events:   1. The system detects a change due to ticket creation or editing. 2. The system generates notification. 3. The system delivers the notification to relevant stakeholders such as the assignee and the PM if the severity of the ticket is high. 4. Stakeholders view the notification in their inbox. 5. The system will display all the tickets that are relevant to the stakeholders in their inbox. | | |
| Sub-flows: | | |
| Alternate/Exceptional Flows:  E-3   1. Notification delivery fails due to a system error or external channel issue such eg: network | | |
| Additional Notes: | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Create New Project | ID: UC | Importance Level: High |
| Primary Actor:  Admin | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  Administrator - creates the project for the BA, QA, PM, Agent and BU.  Other Users – assigned to the project after creation to keep track of tickets that are related to that project. | | |
| Brief Description:  This use case describes how the Admin of the system creates a new project for a project team to keep track of their tickets, knowledge bases and other tasks so that the relevant activities can be separately stored in their respective projects | | |
| Trigger:  When there is a new development project the Admin will create a new project that is tied to the new development project to assign team members and store relevant information. | | |
| Relationships:   * Association : Admin * Include : N/A * Extend : N/A * Generalization : N/A | | |
| Normal Flow of Events:   1. Administrator selects "Create New Project." 2. System displays the project creation form. 3. Administrator enters project details (name, description, dates, roles, team members, etc.). 4. The system validates the input. 5. The system saves the project information. 6. The system confirms successful creation and displays the new project. | | |
| Sub-flows: | | |
| Alternate/Exceptional Flows:  E-4   1. If validation fails (e.g., missing required fields, invalid dates), the system prompts the admin to correct errors. 2. Admin Enters the correct details to proceed to step 5.   E-5   1. System error occurs during saving 2. The project is not created, and an error message is displayed. | | |
| Additional Notes: | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  Create New Subproject | ID: UC | Importance Level: High |
| Primary Actor:  Admin | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  Administrator – Defines permissions for subprojects. The subprojects are created under a main project.  Project Members (Agent, BA, QA, PM, Dev)– Assigned to work on the subproject. | | |
| Brief Description:  This use case describes how an Admin would add a subproject. A subproject is created under an existing main project to represent a specific phase (e.g., Regression Testing), or environment (e.g., UAT, Pilot, Development). The Project Manager defines subproject details and assigns relevant team members. | | |
| Trigger:  The Admin receives requests to create a new subproject or a list of subproject from the Project Manager. | | |
| Relationships:   * Association : Admin * Include : N/A * Extend : N/A * Generalization : N/A | | |
| Normal Flow of Events:   1. Project Manager selects a main project. 2. System displays the option to create a subproject. 3. Project Manager enters subproject details (name, type: phase/environment, description, dates, assigned team). 4. The system validates the details. 5. The system links the subproject to the parent project. 6. The system saves and confirms successful creation with a message: “Subproject has been successfully created, Name: XXX“. | | |
| Sub-flows: | | |
| Alternate/Exceptional Flows:  E-4:   1. If validation fails (e.g., missing required fields, invalid dates), the system prompts corrections. 2. The Admin will have to fill in the correct details to proceed to step 5   E-6:   1. System error while saving, will display error message “The subproject has failed to be created”. | | |
| Additional Notes:  The Admin may allow permissions for the Project manager of the Project to create their own subproject if the project doesn’t require too much restrictions and controls. | | |

|  |  |  |
| --- | --- | --- |
| Use Case Name:  View Knowledge Base | ID: UC | Importance Level: High |
| Primary Actor:  User | | Use Case Type: Detail, Essential |
| Stakeholders and Interests:  Business User – views articles to resolve issues themselves so that they don’t need to raise unnecessary tickets.  Agent – views knowledge base to assist BU if the BU aren’t able to find the articles themselves.  Admin – Make sure knowledge base is accessible to BU as well as the agents and all the team members in the project. | | |
| Brief Description:  This use case describes how users are able to view the knowledge base for articles, frequently asked questions or guides. | | |
| Trigger:   * The BU has a question but feels like it should be simple enough that he can look up frequently asked questions to solve it. * The Agent needs to answer a question by the BU so they go through the Knowledge Base to find an answer template that will help clarify the BU’s doubts. | | |
| Relationships:   * Association : User * Include : N/A * Extend : * Generalization : N/A | | |
| Normal Flow of Events:   1. User navigates to the Knowledge Base 2. The system displays the main Knowledge Base homepage with categories (e.g., *Getting Started, Troubleshooting, FAQs*). 3. The system displays a list of articles 4. User clicks the article title from the results list. 5. The system displays the article’s title, body text, images, and attachments. 6. The user can choose to write comments and to rate the article with thumbs up / down. | | |
| Sub-flows: | | |
| Alternate/Exceptional Flows:  A-3   1. User enters a keyword into the search bar (e.g., “Reset password”). 2. User presses enter and the flow proceeds to Step 3.   A-3  1. User clicks into a category (e.g. troubleshooting), then a section (e.g. Login issues)  2. Flow proceeds to step 3. | | |
| Additional Notes: | | |