Test Identifier	Test Summery	Test Description	Precondition	Test Steps	Expected Result	Actual Results	UI / API	Status	Tester	Defect ID
TC-M1-CAT-UI-001	Verify Admin can view 'Add Category' button	Validate that the 'Add Category' button is visible and enabled only for users with Admin privileges.	1. Logged in as Admin 2. Navigate to Category List Page	1. Observe the Category List page header. 2. Check for the presence of the "Add Category" button.	The "Add Category" button is visible and clickable for admin users.		UI	Not Executed	QA1-Inupama	
TC-M1-CAT-UI-002	Verify 'No category found' message when no records exist	Validate that pagination controls appear when total categories are more than page limit.	1. Logged in as Admin 2. Database has no categories OR search returns no results 3. Navigate to Category List Page	1. Login as Admin 2. Navigate to Category List page with no categories or Enter search term with no matches	Message 'No category found' should be displayed		UI	Not Executed	QA1-Inupama	
TC-M1-CAT-UI-003	Verify Admin can search categories by name	Validate that entering a name in 'Search sub category' and clicking 'Search' filters the results correctly.	1. Logged in as Admin 2. At least 3 categories with different names exist	1. Enter an existing category name in 'Search sub category' field 2. Click 'Search' button 3. Observe filtered results	Only categories whose Name matches or contains the search term are displayed in the table		UI	Not Executed	QA1-Inupama	
TC-M1-CAT-UI-004	Verify Reset button clears search and filter for Admin	Validate that clicking 'Reset' clears the search box, parent filter dropdown, and reloads the complete list.	1. Logged in as Admin 2. Search term and/or parent filter already applied and results are filtered	1. With filtered results displayed, click 'Reset' button 2. Observe the search field, dropdown, and category list	Search field becomes empty, parent dropdown resets to 'All Parents', and all categories are listed again		UI	Not Executed	QA1-Inupama	
TC-M1-CAT-UI-005	Verify Admin can filter categories by parent	Validate that selecting a parent category in the dropdown filters the list to show only its sub-categories.	1. Logged in as Admin 2. At least one main category and several sub-categories mapped to it exist	1. Click on 'All Parents' dropdown 2. Select a specific parent category 3. Click 'Search' button 4. Observe filtered list	Only categories whose Parent equals the selected parent are displayed in the results		UI	Not Executed	QA1-Inupama	
TC-M1-CAT-UI-006	Verify User can use search by category name	Validate that User role can search categories using the 'Search sub category' field.	1. Logged in as User 2. Multiple categories exist with different names	1. Enter an existing category name in search field 2. Click 'Search' button 3. Observe filtered results	List displays only categories matching the search term, search functionality works for User role		UI	Not Executed	QA1-Inupama	
TC-M1-CAT-UI-007	Verify sorting by Name ascending for User	Validate that clicking the Name column header sorts category names alphabetically	1. Logged in as User 2. At least 3 categories with different names exist.	1. Navigate to Category List page. 2. Click on Name column header once. 3. Observe name order.	Category names appear in ascending alphabetical order		UI	Not Executed	QA1-Inupama	
TC-M1-CAT-UI-008	Verify User can use parent filter	Validate that User role can filter categories by selecting a parent category from the dropdown.	1. Logged in as User 2. At least one parent and sub-categories exist	1. Click on 'All Parents' dropdown 2. Select a specific parent category 3. Click 'Search' button 4. Observe the filtered list	Only categories with the selected Parent are displayed.		UI	Not Executed	QA1-Inupama	
TC-M1-CAT-UI-009	Verify "Add Category" button hidden for User	Validate that the "Add Category" button is not visible for regular users	1. Logged in as User 2. User is on Category List page	1. Login as User 2. Navigate to category list page 3. Observe button area	The 'Add Category' button is not visible for regular users		UI	Not Executed	QA1-Inupama	
TC-M1-CAT-UI-010	Verify pagination visible for User	Validate that pagination controls are visible and functional for User	1. Logged in as User 2. Categories exist that span multiple pages	1. Navigate to Categories page 2. Scroll to the bottom of the category list 3. Inspect pagination controls	Pagination controls (Previous/Next or page numbers) are visible and usable for User role		UI	Not Executed	QA1-Inupama	
TC-M1-CAT-API-01	Verify Admin search categories by name returns matching results.	Validate that GET /api/categories returns only categories matching the name parameter.	1. API running 2. Logged in as Admin 3. Categories with different names exist in database	1. Authenticate as Admin and obtain valid bearer token 2. Send GET /api/categories?search=<existingCategoryName> with token 3. Inspect response status and body	Status Code: 200 Response body contains only categories where Name matches or contains <existingCategoryName>		API	Not Executed	QA1-Inupama	
TC-M1-CAT-API-02	Verify Admin gets all categories without filters	Validate that GET /api/categories returns full list of all categories for Admin.	1. API running 2. Logged in as Admin 3. Multiple categories exist in database	1. Authenticate as Admin and obtain bearer token 2. Send GET /api/categories without any query parameters 3. Verify response status and count	Status Code: 200 Response body list size equals or represents the total categories in database		API	Not Executed	QA1-Inupama	
TC-M1-CAT-API-03	Verify Admin sort by ID ascending	Validate that /api/categories?sort=id,asc returns categories sorted by ID	1. API is running 2. Authenticated with Admin role 3. At least 3 categories with different IDs exist	1. Authenticate as Admin 2. Send GET /api/categories?sort=id,asc 3. Verify the order of IDs in response array	Status Code: 200 IDs in response are in ascending order		API	Not Executed	QA1-Inupama	
TC-M1-CAT-API-04	Verify categories search with non-existent name returns empty	Validate that searching for a string that matches no categories returns an empty list but a successful status.	1. API is running 2. Authenticated with Admin role 2. Multiple categories exist in database	1. Send GET request to /api/categories?search. 2. Check response body.	Status Code 200. Response body is an empty list/array [].		API	Not Executed	QA1-Inupama	
TC-M1-CAT-API-05	Verify GET categories returns error for unauthenticated requests	Validate that accessing the endpoint without an auth token returns 401 Unauthorized.	1. API is running 2. No authentication token provided	1. Send GET request to /api/categories without auth token 2. Verify response status code 3. Verify error message	Status Code 401 Unauthorized. Response contains unauthorized error message		API	Not Executed	QA1-Inupama	
TC-M1-CAT-API-06	Verify User can GET categories as read-only	Validate that User role can call GET /api/categories with read-only access	1. API running 2. Logged in as a user User	1. Authenticate as User 2. Send GET /api/categories 3. Verify response status and data received	Status Code: 200. List of categories returned with read-only access, No modification errors		API	Not Executed	QA1-Inupama	
TC-M1-CAT-API-07	Verify search with non-existent category name returns empty	Validate that search with non-existent category name returns empty list	1. User is authenticated 2. Categories exist, but none match the test search term.	1. . Send GET request with search parameter that doesn't match any category 2. Verify response status code 3. Verify empty result set	Status Code 200. An empty list of categories		API	Not Executed	QA1-Inupama	
TC-M1-CAT-API-08	Verify sorting categories by name (ascending)	Validate that the sort parameter correctly orders the returned list alphabetically.	1. Multiple categories exist 2. API is running	1. Send GET request to /api/categories?sort=name,asc. 2. Check order of first and last item.	Status Code 200. List is sorted alphabetically by Name (A-Z).		API	Not Executed	QA1-Inupama	
TC-M1-CAT-API-09	Verify filtering categories by parentId	Verify filtering categories by parent category ID returns correct sub-categories	1.API is running 2.User is authenticated 3. Parent-child category relationships exist	1. Send GET request to /api/categories with parent filter 2. Verify response status code 3. Verify only sub-categories of specified parent are returned	Status Code 200. Response contains only categories where parentId matches the filter		API	Not Executed	QA1-Inupama	
TC-M1-CAT-API-10	Verify GET categories with multiple filter parameters works correctly	Validate that User can combine search by name and parentId filter to retrieve only matching sub-categories.	1. API is running 2. User is authenticated 3. Parent category ID=2 exists with sub-categories 4. Some sub-categories of parent ID=2 have "Tree" in name	1. Authenticate as user 2. Send GET /api/categories?search=Tree&parentId=2 3. Verify response status and data received	Status Code 200 Response contains only categories matching search term AND specified parent category		API	Not Executed	QA1-Inupama	

Test Identifier	Test Summery	Test Description	Precondition	Test Steps	Expected Result	Actual Results	UI / API	Status	Tester	Defect ID
TC-M2-CAT-UI-001	Verify Admin can access Add Category page	Validate that Admin users can open the Add Category page.	1. Logged in as Admin 2. Navigate to Category List page	1. Go to /ui/categories 2. Click on Add Category button	Add Category page is displayed.		UI	Not Executed	QA2-Maheepa	
TC-M2-CAT-UI-002	Verify Admin can add a category with valid name	Validate successful category creation with valid data.	1. Logged in as Admin 2. On Add Category page	1. Enter Category Name = 'Plants' 2. Click Save	Category is created and appears in category list.		UI	Not Executed	QA2-Maheepa	
TC-M2-CAT-UI-003	Verify error when Category Name is empty	Validate mandatory field validation for Category Name.	1. Logged in as Admin 2. On Add Category page	1. Leave Category Name empty 2. Click Save"	Error message displayed -: "Category name is required"		UI	Not Executed	QA2-Maheepa	
TC-M2-CAT-UI-004	Verify error when Category Name length is less than 3	Validate minimum length validation.	1. Logged in as Admin 2. On Add Category page	1. Enter Category Name = 'AB' 2. Click Save	Error message -: "Category name must be between 3 and 10 characters"		UI	Not Executed	QA2-Maheepa	
TC-M2-CAT-UI-005	Verify error when Category Name length is more than 10	Validate maximum length validation.	1. Logged in as Admin 2. On Add Category page	1. Enter Category Name = 'VeryLongName123' 2. Click Save	Error -: "Category name must be between 3 and 10 characters"		UI	Not Executed	QA2-Maheepa	
TC-M2-CAT-UI-006	Verify User cannot see Add Category button	Validate that Add Category is hidden for User role.	1. Logged in as User 2. Navigate to Category List page	1. Observe page header 2. Check for Add Category button	Add Category button is not visible.		UI	Not Executed	QA2-Maheepa	
TC-M2-CAT-UI-007	Verify User cannot edit a category	Validate edit option restriction for User role.	1. Logged in as User 2. Category list available	1. Try to find Edit button for a category	Edit option is hidden or disabled.		UI	Not Executed	QA2-Maheepa	
TC-M2-CAT-UI-008	Verify User cannot delete a category	Validate delete option restriction for User role.	1. Logged in as User 2. Category list available	1. Try to find Delete button for a category	Delete option is hidden or disabled.		UI	Not Executed	QA2-Maheepa	
TC-M2-CAT-UI-009	Verify Admin can edit a category	Validate successful category update.	1. Logged in as Admin 2. Existing category available	1. Click Edit 2. Change name 3. Click Save	Category details are updated successfully.		UI	Not Executed	QA2-Maheepa	
TC-M2-CAT-UI-010	Verify Admin can delete a category	Validate successful deletion of a category.	1. Logged in as Admin 2. Existing category available	1. Click Delete 2. Confirm deletion	Category is removed from category list.		UI	Not Executed	QA2-Maheepa	
TC-M2-CAT-API-001	Verify Admin can create category using API	Validate category creation with valid data.	1. Logged in as Admin in Swagger 2. Authorization applied	1. POST /categories 2. Body -: { 'name':'Flowers' }	Status 200/201 and category created.		API	Not Executed	QA2-Maheepa	
TC-M2-CAT-API-002	Verify API validation for empty category name	Validate mandatory field check via API.	1. Logged in as Admin in Swagger	1. POST /categories 2. Body -: { 'name':'' }	Status 400 with validation error message.		API	Not Executed	QA2-Maheepa	
TC-M2-CAT-API-003	Verify API validation for name length < 3	Validate minimum length rule.	1. Logged in as Admin in Swagger	1. POST /categories 2. Body: { 'name':'AB' }	Status 400 returned.		API	Not Executed	QA2-Maheepa	
TC-M2-CAT-API-004	Verify API validation for name length > 10	Validate maximum length rule.	1. Logged in as Admin in Swagger	1. POST /categories 2. Body: { 'name':'VeryLongName123' }	Status 400 returned.		API	Not Executed	QA2-Maheepa	
TC-M2-CAT-API-005	Verify Admin can delete category via API	Validate deletion operation.	1. Logged in as Admin 2. Category ID exists	1. DELETE /categories/{id}	Status 200 and category deleted.		API	Not Executed	QA2-Maheepa	
TC-M2-CAT-API-006	Verify User cannot create category via API	Validate authorization control.	1. Logged in as User in Swagger	1. POST /categories	Status 403 Forbidden.		API	Not Executed	QA2-Maheepa	
TC-M2-CAT-API-007	Verify User cannot update category via API	Validate authorization control.	1. Logged in as User 2. Category ID exists	1. PUT /categories/{id}	Status 403 Forbidden.		API	Not Executed	QA2-Maheepa	
TC-M2-CAT-API-008	Verify User cannot delete category via API	Validate authorization control.	1. Logged in as User 2. Category ID exists	1. DELETE /categories/{id}	Status 403 Forbidden.		API	Not Executed	QA2-Maheepa	
TC-M2-CAT-API-009	Verify Admin can update category via API	Validate update operation.	1. Logged in as Admin 2. Category ID exists	1. PUT /categories/{id} 2. Body: { 'name':'NewName' }	Status 200 and category updated.		API	Not Executed	QA2-Maheepa	
TC-M2-CAT-API-010	Verify User cannot access Admin-only category APIs	Validate restricted endpoint protection.	1. Logged in as User in Swagger	1. Try POST or DELETE /categories	Status 403 Forbidden.		API	Not Executed	QA2-Maheepa	

Test Identifier	Test Summery	Test Description	Precondition	Test Steps	Expected Result	Actual Results	UI / API	Status	Tester	Defect ID
TC-M3-PLANT-UI-001	Verify plant list page loads for User	Verify that User can view the plant list page with all plants displayed	1. User is logged in with 'User' role 2. Plants exist in the system	1. Navigate to /ui/plants 2. Observe the page loading	1. Plant list page loads successfully 2. All plants are displayed in a table 3. Columns show: Name, Category, Price, Stock, Actions 4. Pagination controls are visible if records > page size 5. 'Add Plant' button is NOT visible for User		UI	Not Executed	QA3-Umesha	
TC-M3-PLANT-UI-002	Verify search functionality by plant name for User	Verify that User can search plants by entering plant name in search box	1. User is logged in with 'User' role 2. Multiple plants exist in the system	1. Navigate to /ui/plants 2. Enter a valid plant name in 'Search plant' field 3. Click 'Search' button	1. Only plants matching the search term are displayed 2. Non-matching plants are filtered out 3. Search is case-insensitive 4. If no match found, 'No plants found' message is displayed		UI	Not Executed	QA3-Umesha	
TC-M3-PLANT-UI-003	Verify filter by category functionality for User	Verify that User can filter plants by selecting a category from dropdown	1. User is logged in with 'User' role 2. Plants exist under multiple categories	1. Navigate to /ui/plants 2. Click on 'All Categories' dropdown 3. Select a specific category 4. Click 'Search' button	1. Dropdown shows all available categories 2. Only plants belonging to selected category are displayed 3. Other plants are filtered out 4. Plant count updates accordingly		UI	Not Executed	QA3-Umesha	
TC-M3-PLANT-UI-004	Verify Low stock badge display for User	Verify that 'Low' badge is displayed when plant quantity is below 5	1. User is logged in with 'User' role 2. Plants exist with quantity < 5 and quantity >= 5	1. Navigate to /ui/plants 2. Observe the Stock column for all plants	1. Plants with quantity < 5 display 'Low' badge in Stock column 2. Plants with quantity >= 5 do NOT display 'Low' badge 3. Badge is visually distinct (different color/style)		UI	Not Executed	QA3-Umesha	
TC-M3-PLANT-UI-005	Verify sorting functionality on Price column for User	Verify that User can sort plants by Price in ascending and descending order	1. User is logged in with 'User' role 2. Multiple plants with different prices exist	1. Navigate to /ui/plants 2. Click on 'Price' column header once 3. Observe the sorting order 4. Click on 'Price' column header again	1. First click sorts plants by Price in ascending order 2. Second click sorts plants by Price in descending order 3. Sorting indicator (arrow) shows current sort direction 4. Plants are correctly ordered by price value		UI	Not Executed	QA3-Umesha	
TC-M3-PLANT-UI-006	Verify plant list page loads for Admin	Verify that Admin can view the plant list page with admin-specific features	1. User is logged in with 'Admin' role 2. Plants exist in the system	1. Navigate to /ui/plants 2. Observe the page elements	1. Plant list page loads successfully 2. All plants are displayed in a table 3. 'Add Plant' button IS visible for Admin 4. Edit and Delete actions are visible in Actions column 5. All standard features (search, filter, sort) are accessible		UI	Not Executed	QA3-Umesha	
TC-M3-PLANT-UI-007	Verify Reset button clears search and filter for Admin	Verify that clicking Reset button clears all search and filter criteria	1. User is logged in with 'Admin' role 2. Plants exist in the system	1. Navigate to /ui/plants 2. Enter text in search field 3. Select a category from filter dropdown 4. Click 'Search' button 5. Click 'Reset' button	1. Search field is cleared 2. Category filter resets to 'All Categories' 3. All plants are displayed (no filters applied) 4. Page returns to default state		UI	Not Executed	QA3-Umesha	
TC-M3-PLANT-UI-008	Verify sorting on Name column for Admin	Verify that Admin can sort plants alphabetically by Name	1. User is logged in with 'Admin' role 2. Multiple plants with different names exist	1. Navigate to /ui/plants 2. Click on 'Name' column header 3. Observe the sorting 4. Click on 'Name' column header again	1. First click sorts plants by Name in ascending order (A-Z) 2. Second click sorts plants by Name in descending order (Z-A) 3. Sorting indicator shows direction 4. Plants are alphabetically ordered correctly		UI	Not Executed	QA3-Umesha	
TC-M3-PLANT-UI-009	Verify sorting on Quantity column for Admin	Verify that Admin can sort plants by Stock/Quantity	1. User is logged in with 'Admin' role 2. Multiple plants with different quantities exist	1. Navigate to /ui/plants 2. Click on 'Stock' column header 3. Observe the sorting 4. Click on 'Stock' column header again	1. First click sorts plants by Quantity in ascending order 2. Second click sorts plants by Quantity in descending order 3. Low stock items can be easily identified when sorted ascending 4. Plants are correctly ordered by quantity value		UI	Not Executed	QA3-Umesha	
TC-M3-PLANT-UI-010	Verify empty state message for Admin	Verify that appropriate message is shown when no plants exist or match search criteria	1. User is logged in with 'Admin' role 2. No plants exist OR search returns no results	1. Navigate to /ui/plants (when no plants exist)OR 2. Enter a search term that matches no plants 3. Observe the display	1. Message 'No plants found' is displayed 2. Message is clearly visible 3. Table headers remain visible 4. 'Add Plant' button remains accessible for Admin		UI	Not Executed	QA3-Umesha	
TC-M3-PLANT-API-001	Verify GET all plants API returns complete plant list for User	Verify that GET /api/plants returns all plants with correct structure	1. User is authenticated with 'User' role 2. Multiple plants exist in database	1. Send GET request to /api/plants 2. Verify response status code 3. Verify response body structure	1. Response status code is 200 OK2. Response returns array of plant objects3. Each plant contains: id, name, price, quantity, category object4. Category object contains: id, name, parent, subCategories5. All plants in database are returned		API	Not Executed	QA3-Umesha	
TC-M3-PLANT-API-002	Verify GET plant by ID API for User	Verify that GET /api/plants/{id} returns specific plant details	1. User is authenticated with 'User' role 2. Plant with known ID exists	1. Send GET request to /api/plants/{validId} 2. Verify response status and structure	1. Response status code is 200 OK 2. Response returns single plant object 3. Plant object contains: id, name, price, quantity, categoryId 4. Returned plant ID matches requested ID 5. All fields contain valid data		API	Not Executed	QA3-Umesha	
TC-M3-PLANT-API-003	Verify GET plants by category API for User	Verify that GET /api/plants/category/{categoryId} returns plants filtered by category	1. User is authenticated with 'User' role 2. Plants exist under specific category	1. Send GET request to /api/plants/category/{validCategoryId} 2. Verify response contains only plants from that category	1. Response status code is 200 OK 2. Response returns array of plant objects 3. All returned plants belong to the specified category 4. Each plant has complete category object with matching categoryId 5. Plants from other categories are not included		API	Not Executed	QA3-Umesha	
TC-M3-PLANT-API-004	Verify GET plant summary API for User	Verify that GET /api/plants/summary returns correct plant statistics	1. User is authenticated with 'User' role 2. Plants exist with varying quantities	1. Send GET request to /api/plants/summary 2. Verify response structure and values	1. Response status code is 200 OK 2. Response contains: totalPlants, lowStockPlants 3. totalPlants equals total number of plants in database 4. lowStockPlants equals count of plants with quantity < 5 5. Both values are non-negative integers		API	Not Executed	QA3-Umesha	
TC-M3-PLANT-API-005	Verify GET plants with pagination API for User	Verify that GET /api/plants/paged supports pagination parameters	1. User is authenticated with 'User' role 2. More than 10 plants exist in database	1. Send GET request to /api/plants/paged?page=0&size=5 2. Verify pagination metadata 3. Send request for page=1 4. Compare results	1. Response status code is 200 OK 2. Response contains: totalPages, totalElements, size, content array, number, first, last 3. content array contains max 5 plants (per size param) 4. totalElements shows total plant count 5. Page 0 and page 1 return different plants 6. first=true for page 0, last=true for final page		API	Not Executed	QA3-Umesha	
TC-M3-PLANT-API-006	Verify GET plants with sorting API for Admin	Verify that GET /api/plants/paged supports sorting by name and price	1. User is authenticated with 'Admin' role 2. Multiple plants with varying prices exist	1. Send GET request to /api/plants/paged?sort=price,asc 2. Verify ascending price order 3. Send GET request to /api/plants/paged?sort=name,desc 4. Verify descending name order	1. Response status code is 200 OK 2. For price,asc: Plants ordered by price low to high 3. For name,desc: Plants ordered alphabetically Z to A 4. sort array in response reflects requested sort parameters 5. Sorting is applied correctly across all pages		API	Not Executed	QA3-Umesha	
TC-M3-PLANT-API-007	Verify GET plant by invalid ID returns 404 for Admin	Verify that requesting non-existent plant ID returns appropriate error	1. User is authenticated with 'Admin' role 2. Plant ID 99999 does not exist	1. Send GET request to /api/plants/99999 2. Verify error response	1. Response status code is 404 Not Found 2. Response contains appropriate error message 3. No plant data is returned 4. Error message indicates plant not found		API	Not Executed	QA3-Umesha	
TC-M3-PLANT-API-008	Verify GET plants by invalid category returns empty list for Admin	Verify behavior when requesting plants for non-existent or empty category	1. User is authenticated with 'Admin' role 2. Category ID 99999 does not exist OR has no plants	1. Send GET request to /api/plants/category/99999 2. Verify response	1. Response status code is 200 OK (or 404 based on implementation) 2. Response returns empty array []OR 3. Returns 404 with appropriate error message 4. No incorrect plant data is returned		API	Not Executed	QA3-Umesha	
TC-M3-PLANT-API-009	Verify pagination boundary conditions for Admin	Verify behavior at pagination boundaries (first page, last page, invalid page)	1. User is authenticated with 'Admin' role 2. Exactly 12 plants exist in database	1. Send GET request to /api/plants/paged?page=0&size=10 2. Send GET request to /api/plants/paged?page=1&size=10 3. Send GET request to /api/plants/paged?page=5&size=10	1. User is authenticated with 'Admin' role 2. Exactly 12 plants exist in database		API	Not Executed	QA3-Umesha	
TC-M3-PLANT-API-010	Verify combined filter and sort API for Admin	Verify that GET /api/plants/paged works with multiple query parameters	1. User is authenticated with 'Admin' role 2. Plants exist under multiple categories with varying quantities	1. Send GET request to /api/plants/paged?page=0&size=10&sort=quantity,asc 2. Verify combined parameters work correctly	1. Response status code is 200 OK 2. Pagination is applied (max 10 results) 3. Plants are sorted by quantity ascending 4. Response includes both pagination metadata and sort information 5. Results are correctly filtered and ordered 6. Page navigation works with sorting maintained		API	Not Executed	QA3-Umesha	

Test Identifier	Test Summery	Test Description	Precondition	Test Steps	Expected Result	Actual Results	UI / API	Status	Tester	Defect ID
TC-M4-PLANT-UI-001	Verify price validation error message in Add Plant form	Test that the UI displays appropriate error message when price is 0 or negative in the Add Plant form	1. User logged in as admin, on Plants page	1. Navigate to /Plants/ Add a plant 2. Fill Plant Name, Category, Quantity 3. Enter price = 0 Submit form 4. Enter price = -10 Submit form	Error message "Price must be greater than 0" displays below price field, form not submitted, plant not added to table		UI	Not Executed	QA4-Dinuri	
TC-M4-PLANT-UI-002	Verify category dropdown shows only sub-categories	Test that the category dropdown in Add Plant form displays only sub-categories and not main categories	1. User logged in as admin, on Plants page 2. Categories properly configured	1. Click "Add a Plant" button 2. Click on Category dropdown 3. Observe available options 4. Verify only sub-categories listed	Category dropdown displays only sub category options, no main category options visible		UI	Not Executed	QA4-Dinuri	
TC-M4-PLANT-UI-003	Verify plant edit functionality with data pre-population	Test that clicking edit icon opens form with existing plant data and allows successful update	1. User logged in as admin 2. At least one plant exists in the plants table	1. Click edit (pencil) icon for existing plant 2. Verify form opens with pre-populated data 3. Modify plant name and price 4. Submit form 5. Verify success message 6. Verify updated data in table	Edit form opens with current plant data, modifications save successfully, updated values display in plants table		UI	Not Executed	QA4-Dinuri	
TC-M4-PLANT-UI-004	Verify plant deletion from UI	Test that clicking delete icon removes plant from the table after confirmation	1. User logged in as admin 2. At least one plant exists in the plants table	1. Note total number of plants 2. Click delete (trash) icon for a plant 3. Confirm deletion in confirmation dialog 4. Verify success message 5. Verify plant removed from table 6. Verify plant count decreased by 1	Confirmation dialog appears, plant removed from table after confirmation, success message displayed, plant no longer visible in list		UI	Not Executed	QA4-Dinuri	
TC-M4-PLANT-UI-005	Verify required field validation in Add Plant form	Test that all required fields show validation errors when form is submitted empty	1. User logged in as admin, on Plants page	1. Click "Add a Plant" button 2. Leave all fields empty 3. Click Submit 4. Verify validation errors for each required field (Plant Name, Category, Price, Quantity)	Validation error messages display for all required fields: "Name is required", "Category is required", "Price is required", "Stock is required", form not submitted		UI	Not Executed	QA4-Dinuri	
TC-M4-PLANT-UI-006	Verify user can view all plants in table	Test that regular users can view the complete plants list with all details displayed correctly	1. User logged in as regular user (non-admin), on Plants page, multiple plants exist	1. Navigate to Plants page 2. Verify table displays 3. Verify all columns visible (Plant Name, Category, Price, Quantity) 4. Verify all plants listed 5. Verify data accuracy	Plants table displays all plants with correct data.		UI	Not Executed	QA4-Dinuri	
TC-M4-PLANT-UI-007	Verify Plants edit button is hidden for regular users	Test that regular users do not see edit action button in the Actions column	1. User logged in as regular user (non-admin), on Plants page, multiple plants exist	1. Navigate to Plants page 2. Observe table structure 3. Check if Actions column exists 4. Verify no edit (pencil) icon visible	Actions column is either hidden completely or displays no action buttons for regular users, no edit functionality available		UI	Not Executed	QA4-Dinuri	
TC-M4-PLANT-UI-008	Verify Plants delete button is hidden for regular users	Test that regular users do not see delete action button in the Actions column	1. User logged in as regular user (non-admin), on Plants page, multiple plants exist	1. Navigate to Plants page 2. Observe table structure 3. Check if Actions column exists 4. Verify no edit delete (trash) icon visible	Actions column is either hidden completely or displays no action buttons for regular users, no delete functionality available		UI	Not Executed	QA4-Dinuri	
TC-M4-PLANT-UI-009	Verify "Add a Plant" button is hidden for regular users	Test that regular users do not see the "Add a Plant" button on Plants page	1. User logged in as regular user (non-admin), on Plants page	1. Navigate to Plants page 2. Observe top-right section of page 3. Verify "Add a Plant" button is not visible 4. Verify only Search and Reset buttons available	"Add a Plant" button is not displayed for regular users, only Search and Reset buttons visible		UI	Not Executed	QA4-Dinuri	
TC-M4-PLANT-UI-010	Verify read-only view of plant data for regular users	Test that regular users can view plant information but cannot interact with or modify any data fields	1. User logged in as regular user (non-admin), on Plants page 2. Plants displayed in table	1. Navigate to Plants page 2. Attempt to click on plant name in table 3. Verify no edit modal/form opens 4. Attempt to click on price field 5. Verify field is not editable 6. Right-click on plant row 7. Verify no context menu with edit/delete options appears	All plant data displays as read-only text, no clickable edit functionality, no inline editing possible, no context menus with modification options available for regular users		UI	Not Executed	QA4-Dinuri	
TC-M4-PLANT-API-001	Verify successful plant creation with valid data	Test the POST /api/plants/category/{categoryId} endpoint to create a new plant with all valid required fields	1. User authenticated as admin 2. valid sub-category ID available	1. Send POST request to /api/plants/category/{categoryId} with valid data (Plant Name, Category, Price>0, Quantity≥0) 2. Verify response status code 3. Verify plant created in database	API returns 201 Created status, response contains plant details with generated ID, plant appears in plants list		API	Not Executed	QA4-Dinuri	
TC-M4-PLANT-API-002	Verify API validation for invalid price values	Test that API rejects plant creation when price is 0 or negative	1. User authenticated as admin 2. valid sub-category ID available	1. Send POST request with price = 0 2. Verify error response 3. Send POST request with price = -5 4. Verify error response	API returns 400 Bad Request, error message indicates "Price must be greater than 0"		API	Not Executed	QA4-Dinuri	
TC-M4-PLANT-API-003	Verify API rejects main category in plant creation	Test that API validates and rejects attempts to create plant with main category instead of sub-category	1. User authenticated as admin, main category ID available	1. Identify main category ID (not sub-category) 2. Send POST request to /api/plants/category/{mainCategoryId} 3. Verify error response	API returns 400 Bad Request with error message "Plant must be assigned to a sub-category only"		API	Not Executed	QA4-Dinuri	
TC-M4-PLANT-API-004	Verify successful plant update operation	Test the PUT /api/plants/{id} endpoint to update existing plant details	1. User authenticated as admin, existing plant ID available	1. Get existing plant ID 2. Send PUT request with modified data (Plant Name, Category, Price, Quantity) 3. Verify response 4. Verify updated data in GET request	API returns 200 OK, response contains updated plant details, changes reflected in database		API	Not Executed	QA4-Dinuri	
TC-M4-PLANT-API-005	Verify successful plant deletion	Test the DELETE /api/plants/{id} endpoint to remove a plant	1. User authenticated as admin, existing plant ID available for deletion	1. Get existing plant ID 2. Send DELETE request to /api/plants/{id} 3. Verify response status 4. Verify plant removed from GET /api/plants	API returns 200 OK or 204 No Content, plant no longer appears in plants list		API	Not Executed	QA4-Dinuri	
TC-M4-PLANT-API-006	Verify user can retrieve all plants via API	Test that GET /api/plants endpoint returns all available plants for regular users	1. User authenticated as regular user (non-admin) 2. Multiple plants exist in database	1. Send GET request to /api/plants 2. Verify response status code 3. Verify response contains plant list 4. Verify each plant has required fields (id, Plant Name, Category, Price, Quantity)	API returns 200 OK, response contains array of all plants with complete details		API	Not Executed	QA4-Dinuri	
TC-M4-PLANT-API-007	Verify user can retrieve plants by category	Test that GET /api/plants/category/{categoryId} endpoint filters plants by sub-category for users	1. User authenticated as regular user 2. Plants exist in multiple categories	1. Get Culinary category ID 2. Send GET request to /api/plants/category/{culinaryId} 3. Verify response 4. Confirm only Culinary plants returned	API returns 200 OK, response contains only plants from specified category		API	Not Executed	QA4-Dinuri	
TC-M4-PLANT-API-008	Verify user cannot update plant details via API	Test that regular users are restricted from updating plant information using PUT endpoint	1. User authenticated as regular user (non-admin) 2. Existing plant ID available with current data	1. Get existing plant ID 2. Prepare modified plant data 3. Send PUT request to /api/plants/{id} with updated data 4. Verify response status 5. Verify original plant data unchanged	API returns 403 Forbidden or 401 Unauthorized, error message indicates user lacks permission to update plants, plant data remains unchanged at original values		API	Not Executed	QA4-Dinuri	
TC-M4-PLANT-API-009	Verify user cannot create plant via API	Test that regular users are restricted from creating plants using POST endpoint	1. User authenticated as regular user (non-admin) 2. Valid plant data prepared	1. Send POST request to /api/plants/category/{categoryId} with valid plant data 2. Verify response status 3. Verify error message indicates insufficient permissions	API returns 403 Forbidden or 401 Unauthorized, error message indicates user lacks permission to create plants		API	Not Executed	QA4-Dinuri	
TC-M4-PLANT-API-010	Verify user cannot delete plant via API	Test that regular users are restricted from deleting plants using DELETE endpoint	1. User authenticated as regular user (non-admin) 2. Existing plant ID available	1. Get existing plant ID 2. Send DELETE request to /api/plants/{id} 3. Verify response status 4. Verify plant still exists in database	API returns 403 Forbidden or 401 Unauthorized, error message indicates insufficient permissions, plant not deleted		API	Not Executed	QA4-Dinuri	

Test Identifier	Test Summery	Test Description	Precondition	Test Steps	Expected Result	Actual Results	UI / API	Status	Tester	Defect ID
TC-M5-SALE-UI-001	Verify sales list page loads with pagination	Validate that sales list displays all sales records with pagination controls	1. User logged in as Admin 2. Sales records exist in system	1. Navigate to /ui/sales 2. Observe the page layout	Sales list page loads successfully with pagination controls visible at bottom		UI	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-002	Verify Sell Plant button is visible only for Admin	Validate Admin-specific button visibility on sales list page	1. User logged in as Admin 2. Sales records exist in system	1. Navigate to /ui/sales 2. Check for Sell Plant button	Sell Plant button is visible and clickable		UI	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-003	Verify Sell Plant button is hidden for regular User	Validate button visibility restriction for non-admin users	1. User logged in as regular User (not Admin)	1. Navigate to /ui/sales 2. Check for Sell Plant button	Sell Plant button is not visible or disabled		UI	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-004	Verify No sales found message when no records exist	Validate empty state message display	1. User logged in 2. No sales records in system	1. Navigate to /ui/sales 2. Observe the page content	Message No sales found is displayed		UI	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-005	Verify delete action is visible only for Admin	Validate delete button/icon visibility for Admin role	1. User logged in as Admin 2. Sales records exist	1. Navigate to /ui/sales 2. Check each sale record for delete action	Delete action (button/icon) is visible for each sale record		UI	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-006	Verify default sorting by sold date descending	Validate that sales list is sorted by most recent date first by default	1. User logged in 2. Multiple sales records exist with different dates	1. Navigate to /ui/sales 2. Observe the order of records	Sales records are displayed in descending order by sold date (newest first)		UI	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-007	Verify sorting by plant name	Validate sorting functionality for plant name column	1. User logged in 2. Multiple sales records exist with different dates	1. Navigate to /ui/sales 2. Click on Plant Name column header 3. Observe the sorting	Sales records are sorted alphabetically by plant name (A-Z or Z-A)		UI	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-008	Verify sorting by quantity	Validate sorting functionality for quantity column	1. User logged in 2. Multiple sales records exist with different dates	1. Navigate to /ui/sales 2. Click on Quantity column header 3. Observe the sorting	Sales records are sorted by quantity (ascending or descending)		UI	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-009	Verify sorting by total price	Validate sorting functionality for total price column	1. User logged in 2. Multiple sales records exist with different dates	1. Navigate to /ui/sales 2. Click on Total Price column header 3. Observe the sorting	Sales records are sorted by total price (ascending or descending)		UI	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-010	Verify Sell Plant page loads with plant dropdown	Validate sell plant form displays correctly with available plants	1. User logged in as Admin 2. Plants with stock exist	1. Navigate to /ui/sales 2. Click Sell Plant button	Sell Plant page loads at /ui/sales/new with plant dropdown showing available plants with current stock		UI	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-011	Verify validation error when plant is not selected	Validate mandatory field validation for plant selection	1. User on Sell Plant page (/ui/sales/new)	1. Leave Plant dropdown empty 2. Enter quantity value 3. Click Submit	Error message Plant is required displays below plant field in red		UI	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-012	Verify validation error when quantity is zero	Validate quantity validation rule (must be greater than 0)	1. User on Sell Plant page 2. Plant selected	1. Select a plant from dropdown 2. Enter quantity as 0 3. Click Submit	Error message Quantity must be greater than 0 displays below quantity field		UI	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-013	Verify validation error when quantity is negative	Validate negative quantity validation	1. User on Sell Plant page 2. Plant selected	1. Select a plant from dropdown 2. Enter quantity as -5 3. Click Submit	Error message Quantity must be greater than 0 displays below quantity field		UI	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-014	Verify successful sale redirects to sales list	Validate successful sale flow and redirection	1. User on Sell Plant page 2. Plant with sufficient stock exists	1. Select plant from dropdown 2. Enter valid quantity (within stock) 3. Click Submit	Sale is created successfully and user is redirected to /ui/sales page		API	Not Executed	QA5-Chamod	
TC-M5-SALE-UI-015	Verify Cancel button navigates back to sales list	Validate cancel functionality on sell plant form	1. User on Sell Plant page (/ui/sales/new)	1. Click Cancel button	User is redirected to /ui/sales without creating sale		API	Not Executed	QA5-Chamod	
TC-M5-SALE-API-001	Verify GET all sales returns 200 with sales list	Validate GET /api/sales endpoint returns all sales	1. Valid authentication token 2. Sales records exist	1. Send GET request to /api/sales 2. Verify response	Status code: 200 Response body contains array of sales with fields: id, plant object, quantity, totalPrice, soldAt		API	Not Executed	QA5-Chamod	
TC-M5-SALE-API-002	Verify GET sales by ID returns correct sale details	Validate GET /api/sales/{id} endpoint for specific sale	1. Valid authentication token 2. Known sale ID exists	1. Send GET request to /api/sales/{id} 2. Verify response structure	Status code: 200 Response contains: id, plant{id, name, price, quantity, category}, quantity, totalPrice, soldAt in correct format		API	Not Executed	QA5-Chamod	
TC-M5-SALE-API-003	Verify GET sales with pagination parameters	Validate GET /api/sales/page with pagination and sorting	1. Valid authentication token 2. Multiple sales exist	1. Send GET request to /api/sales/page?page=0&size=10&sort=soldAt,desc 2. Verify response	Status code: 200. Response contains paginated sales sorted by soldAt in descending order		API	Not Executed	QA5-Chamod	
TC-M5-SALE-API-004	Verify GET sales returns 404 for non-existent ID	Validate error handling for invalid sale ID	1. Valid authentication token	1. Send GET request to /api/sales/99999 (non-existent ID) 2. Verify response	Status code: 404. Error message indicates sale not found		API	Not Executed	QA5-Chamod	
TC-M5-SALE-API-005	Verify POST sale creates sale and reduces stock	Validate successful sale creation and inventory update	1. Valid Admin authentication token 2. Plant exists with quantity >= 5	1. Note plant's current quantity 2. Send POST to /api/sales/plant/{plantId} with body: {quantity: 5} 3. Verify response 4. GET plant to verify stock	Status code: 201Sale created with correct totalPrice calculationPlant stock reduced by sold quantity		API	Not Executed	QA5-Chamod	
TC-M5-SALE-API-006	Verify POST sale fails when quantity exceeds stock	Validate stock validation during sale creation	1. Valid Admin token2. Plant exists with quantity = 3	1. Send POST to /api/sales/plant/{plantId} with body: {quantity: 10} 2. Verify response	Status code: 400. Error message indicates insufficient stock		API	Not Executed	QA5-Chamod	
TC-M5-SALE-API-007	Verify POST sale fails with invalid quantity (0)	Validate quantity validation rule in API	1. Valid Admin authentication token2. Valid plant ID	1. Send POST to /api/sales/plant/{plantId} with body: {quantity: 0} 2. Verify response	Status code: 400. Error message: Quantity must be greater than 0		API	Not Executed	QA5-Chamod	
TC-M5-SALE-API-008	Verify POST sale fails for non-existent plant	Validate plant existence check during sale	1. Valid Admin authentication token	1. Send POST to /api/sales/plant/99999 with body: {quantity: 5} 2. Verify response	Status code: 404. Error message indicates plant not found		API	Not Executed	QA5-Chamod	
TC-M5-SALE-API-009	Verify DELETE sale removes sale record successfully	Validate successful sale deletion	1. Valid Admin authentication token2. Sale record exists with known ID	1. Send DELETE request to /api/sales/{id} 2. Verify response 3. Send GET to /api/sales/{id} to confirm deletion	Status code: 200 or 204. Subsequent GET returns 404 confirming deletion		API	Not Executed	QA5-Chamod	
TC-M5-SALE-API-010	Verify DELETE sale fails for regular User role	Validate role-based access control for delete operation	1. Valid User (non-Admin) authentication token2. Sale record exists	1. Send DELETE request to /api/sales/{id} as regular User 2. Verify response	Status code: 403 (Forbidden). Error message indicates insufficient permissions		API	Not Executed	QA5-Chamod	