Leave Management App - Project Report

Prepared by: SHAIK BAJI
Project Title: Leave Management App
Industry: Human Resources (HR) / Workforce Management
Project Type: Custom CRM Application (Built on Salesforce Platform using Lightning Web Components, custom objects, and Apex)
Target Users: Employees, managers, and HR administrators

Problem Statement

Traditional leave management processes are often manual, fragmented, and inefficient, leading to challenges such as delayed leave approvals, lack of visibility into employee leave statuses, inconsistent tracking of leave requests, and difficulties in ensuring compliance with organizational policies. These inefficiencies can result in employee dissatisfaction, administrative errors, and disrupted workforce planning.

Phase 1: Project Setup and Initialization

Objective: Set up the development environment and create the Salesforce project structure.

Tasks:

- Installed Visual Studio Code (VS Code) with Salesforce CLI and extensions
- Created a new Salesforce project using the command palette (Ctrl+Shift+P) and selected the "SFDX: Create Project" option with the standard template
- Named the project "LeaveTrackerApp"
- Authorized the Salesforce org using the command palette (SFDX: Authorize an Org) with the alias "LiveProject" and default developer org settings
- Verified successful org connection in VS Code

Phase 2: Object Creation

Objective: Create the LeaveRequest__c custom object to store leave request data.

Tasks:

- Logged into Salesforce Developer Org in web browser
- Created a custom object named LeaveRequest (API name: LeaveRequest__c)
- Opened Setup (gear icon top right) and navigated to Object Manager
- Clicked Create > Custom Object and filled in required details:
    - Label: Leave Request
    - Plural Label: Leave Requests
    - Object Name: LeaveRequest (Salesforce appends __c)
    - Record Name: Auto Number (format: LR{0000})

    - Start Number: 1
- Created custom fields for the object:
    - From_Date__c — Date, required
    - To_Date__c — Date, required
    - Reason__c — Long Text Area
    - Status__c — Picklist with values "Pending", "Approved", "Rejected" (default: Pending)
    - User__c — Lookup(User)
    - Manager_Comment__c — Long Text Area
- Set object and field level security and permissions as needed
- Created test LeaveRequest record to verify correctness

Phase 3: Apex Classes Development

Objective: Develop Apex classes to handle data operations and business logic.

Tasks:

- Created LeaveRequestController.cls with @AuraEnabled methods
- Implemented getMyLeaves() method to retrieve leave requests for current user with fields Id, Name, From_Date__c, To_Date__c, Reason__c, Status__c, and Manager_Comment__c
- Added conditional logic to check user profile for manager access
- Deployed Apex classes to the org using SFDX: Deploy Source to Org
- Tested methods using Developer Console to ensure proper functionality

Phase 4: Lightning Web Components Creation

Objective: Create LWC components for the application's user interface.

Tasks:

- In VS Code explorer, right-clicked force-app/main/default/lwc/
- Selected SFDX Create Lightning Web Component
- Named the component: leaveTracker (exactly, no spaces)
- Created new folder leaveTracker with files:
    - leaveTracker.html
    - leaveTracker.js
    - leaveTracker.js-meta.xml
    - leaveTracker.test.js
- Built HTML template with lightning-card and lightning-record-edit-form
- Implemented JavaScript logic with wire service and data handling
- Configured component metadata for Lightning App Builder exposure

- Added optional styling with SLDS classes
- Deployed component to org and verified successful deployment

Phase 5: Lightning App and Page Setup

Objective: Create a Lightning App and App Page to host the LWC components.

Tasks:

- Navigated to Setup > App Manager and created new Lightning App named "Leave Tracker App"
- Selected the System Administrator profile for access and skipped optional settings
- In Lightning App Builder, created new App Page named "Leave Tracker" with single-column layout
- Dragged the leaveTracker component onto the page, saved, and activated it
- Added the page to the "Leave Tracker App" as a tab
- Verified the app in the App Launcher, ensuring the leaveTracker component displayed correctly

Phase 6: User Acceptance Testing (UAT)

Objective: Validate application functionality from end-user perspective.

Tasks:

- Logged in with non-Admin user license in dev org (used standard profile)
- Navigated to the Leave Tracker app via the App Launcher
- Submitted new leave request by filling From Date, To Date, and Reason fields
- Clicked Submit and confirmed form reset with new request appearing in data table
- Tested edit functionality by clicking Edit icon on table row
- Changed field values and clicked Save
- Confirmed table updates reflected changes correctly

Phase 7: Manager Approval Workflow

Objective: Implement manager approval functionality for leave requests.

Tasks:

- Created manager user in Setup > Users with required permissions
- Filled in required fields (First Name, Last Name, Email, Username)
- Assigned System Administrator profile for Edit permissions on LeaveRequest__c
- Logged in as manager user using Login action from Users list
- Navigated to Leave Tracker app and located pending requests
- Used Approve/Reject buttons on data table rows

- Verified status updates and confirmation messages
- Confirmed Status column showed "Approved" or "Rejected" appropriately

Phase 8: Apex Controller Enhancement

Objective: Update LeaveRequestController to support manager and user views.

Tasks:

- Modified getMyLeaves() method in LeaveRequestController.cls
- Added logic to check current user's profile name
- Implemented conditional data retrieval:
  - If user profile contains 'Manager' or 'System Administrator': return all leave requests
  - Otherwise: return only current user's requests
- Updated SOQL query to handle both scenarios efficiently
- Saved and deployed Apex class changes to org
- Tested in Lightning Experience to verify managers see all requests while regular users see only their own

Phase 9: Performance & Debugging

Objective: Ensure application performance and resolve any technical issues.

Tasks:

- Opened Browser Console (F12 → Console) for JavaScript debugging
- Exercised component functionality (submit, edit, approve operations)
- Verified no JavaScript errors or warnings in console
- Checked Salesforce Developer Console logs for server-side errors
- Monitored page load times and response performance
- Validated SOQL query efficiency and data retrieval speed

Phase 10: Testing & Verification

Objective: Comprehensive testing and final validation of the Leave Tracker application.

User Acceptance Testing as Regular User (BAJI.SHAIK):

- Successfully logged into development org as standard user
- Created new leave requests with From Date, To Date, and Reason
- Confirmed form clears after submission and new requests appear in table
- Successfully edited existing requests and verified table updates
- Verified only personal leave requests are visible to regular users

Manager User Testing (EPIC.OrgFarm):

- Successfully logged in with manager profile (Marketing Team role)
- Confirmed all leave requests from all users are visible
- Successfully approved and rejected pending requests
- Verified real-time status changes reflected in data table
- Added manager comments during approval/rejection process

Apex Controller Testing:

- Updated LeaveRequestController.cls with enhanced functionality
- Verified getMyLeaves(): Conditional data retrieval based on user profile
- Tested saveLeaveRequest(): New request creation with validation
- Confirmed updateLeaveStatus(): Manager approval/rejection functionality
- Validated getLeaveRequestsForApproval(): Pending requests for managers

User Management Testing:

- Created "Robert Manager" user with Marketing Team role
- Assigned System Administrator profile for full permissions
- Set up email configuration for user activation
- Confirmed manager can see all requests while users see only their own

UI/UX Testing:

- Data Table Display: All leave requests displayed correctly in tabular format
- Form Functionality: Submit request form works without errors
- Responsive Design: Interface works properly in Lightning Experience
- Real-time Updates: Status changes reflect immediately in the UI
- Error Handling: Proper error messages for validation failures

Data Integrity Testing:

- Record Creation: Leave requests created with proper field values
- Status Management: Status updates (Pending → Approved/Rejected) working correctly
- User Association: Requests properly linked to submitting users
- Audit Trail: CreatedDate and other system fields populated correctly

Sample Data Verified:

- LR0001 Testing request - Status: Approved
- LR0002 Testing request - Status: Pending
- LR0003 Testing and checking - Status: Pending
- LR0004 Going outside - Status: Pending
- LR0005 Going outside for eating food - Status: Approved
- LR0006 Other reason - Status: Approved

Security Testing:

- Role-Based Access: Managers see all requests, users see own requests only
- Permission Validation: CRUD permissions working as expected
- Profile Security: Different profiles have appropriate access levels
- Field-Level Security: Sensitive fields protected appropriately

Performance Testing:

- Page Load Speed: Lightning page loads quickly
- Data Retrieval: SOQL queries execute efficiently
- Browser Console: No JavaScript errors detected
- Server Response: Apex methods respond promptly

End-to-End Workflow Testing:

- Request Submission: User creates leave request → Status: Pending
- Manager Review: Manager views pending request in dashboard
- Approval Decision: Manager approves/rejects with comments
- Status Update: Request status changes to Approved/Rejected
- Final Verification: Updated status visible to both user and manager

Browser Compatibility:

- Chrome: Full functionality working
- Lightning Experience: Optimal performance confirmed

Deployment Verification:

- Code Quality: Clean, well-documented Apex code
- Error Handling: Comprehensive exception handling implemented
- User Experience: Intuitive interface for both users and managers
- Scalability: Architecture supports multiple users and requests

Project Completion Summary

Final Test Results: LEAVE TRACKER PROJECT SUCCESSFULLY COMPLETED

Project Credentials:

- Username: bajishaikproject@gmail.com
- Org Type: Salesforce Developer Edition
- Project Name: LeaveTrackerApp

Key Achievements:

- Successfully implemented complete leave management workflow
- Created secure role-based access control

- Developed intuitive user interface with real-time updates
- Implemented comprehensive testing across all user personas
- Achieved 100% functional requirements satisfaction
- Delivered production-ready Salesforce application

Technical Stack:

- Platform: Salesforce Lightning Platform
- Frontend: Lightning Web Components (LWC)
- Backend: Apex Classes
- Database: Salesforce Custom Objects
- UI Framework: Salesforce Lightning Design System (SLDS)
- Development Environment: Visual Studio Code with Salesforce CLI