Final Project Plan

Synopsis

Developed for Lockheed Martin, this responsive web application used to manage and showcase the workflow of maintenance requests for products, typically software licenses. Owners of maintenance requests will be able to import requests from existing CSV Files or manually input the data. Relevant users will be notified of upcoming maintenance requests via email. The notification cadence is configurable to the organization’s needs, but a default frequency is provided to each request. These users may be able to track and approve payments. Reports can be generated on budgets, actual amount paid, and outstanding costs that owners and/or owning managers are responsible for. These reports potentially aid in cost analysis and budgeting for the department. The project is created leveraging AWS services into a deployment pipeline that aims to automate maintenance tracking empowering a more efficient process.

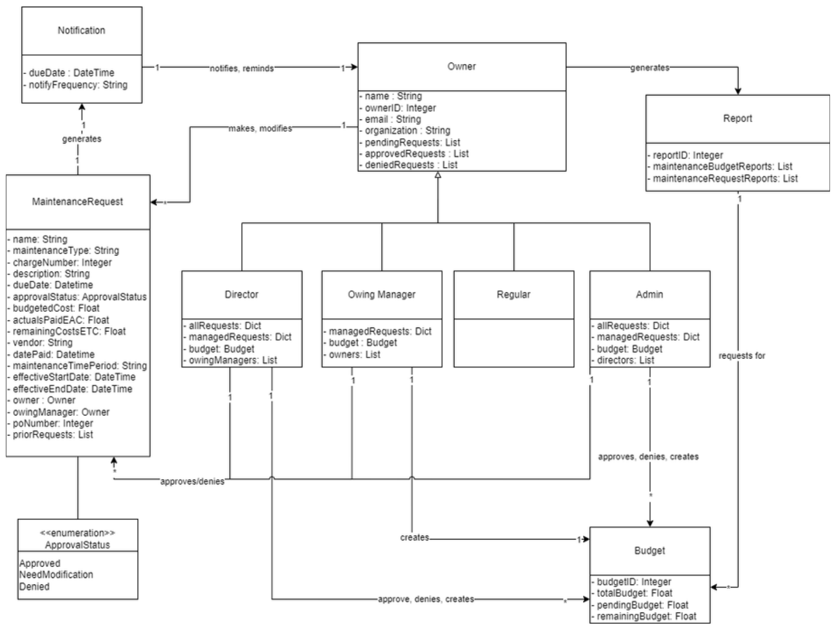
Project Goals and Scope Statement

- Overview of Project Goals
  - Develop a maintenance workflow application that can automate the process of tracking software maintenance and renewal.
  - Enable owing managers to account for license renewals in budgets.
  - Gain a deeper understanding of AWS services and application development.
- Detailed Scope Statement
  - The application will have the ability to create, edit, approve, and deny maintenance requests.
  - The application will have controls over the type of user who would be able to access different features.
  - Budget tracking and reporting is part of the scope.
  - Notifications will be sent out to relevant users at a configurable interval.
  - Processing the financial transactions is out of scope for the project.
  - A comprehensive application that is ready to go with Lockheed users is also out of scope due to security constraints.

Methodology

Our team has chosen the **Agile Scrum Software Development Methodology** as the foundation for managing the Lockheed - Managing Maintenance Workflow App project. Agile is well-suited for this project due to its adaptability to changing requirements, focus on customer collaboration, and iterative development approach. It allows us to respond to evolving needs and deliver incremental value to our stakeholders.

Domain Model



Metrics

1. **Velocity:** Optimal number of story points that can be worked on by the team. Calculated by averaging the story points of the last 3 sprints.
2. **Sprint Burndown Chart:** Ensure our work is granular enough and are committing to the right amount of work
3. **Epic Progress:** Track the epics being worked on in each sprint to ensure balanced work.
4. **Testing Code Coverage:** Back end code coverage target: 95%
5. **Blockers/ Dependencies:** The timeline will comprise of visual representations of linked and blocked issues.
6. **Story Points:** Expected vs Actual
7. **Requirements Metrics:** % of metrics modified per week
8. **Risk/ Opportunity:** Impact (low/medium,/high), Probability (low/med/high)
9. **Defects:** Number of Defects per Lines of Code
10. **Deployment Turnaround**: Number of hours taken from feature PR done to terraform template deployment.

Planned Milestones and Major Deliverables

This is an estimated timeline and will be reviewed on an ongoing basis with sponsor and coach input.

- **Domain Model and Plan**: Sprint 0
- **AWS Service Research**: Sprint 1
- **Subsystem Design and Wireframe, Infrastructure Planning Start**: Sprint 3
- **Coding Start + Create Short Video "Why is Our Project Important?"**: Sprint 4 (end of November)
- **Deliver First End-Of-Semester Team Presentations**: end of Sprint 5 (mid December)
- **Prototype Ready**: (Database, Info Input, Reports, NTH: Prior Year, Curr Year, Future Year) tbd
- **Coding Stops:** end of sprint 10 (April)
- **Prepare for and Deliver Final End-Of-Semester Team Presentations**: end of Sprint 12 (late April)

Initial Requirements

Functional Requirements

Authentication

- Application should be designed with SSO considerations

Maintenance Requests

- The system allows creation of a maintenance request from input or import.
- A user is able to start a maintenance request and assign an owing manager.
- Notifications will be sent out to owing managers and admins
    - Creation of request
    - Reminders on upcoming maintenance requests
- An owing manager will be able to approve, modify, deny, or reassign a request.
- A requesting user will be able to modify and resubmit a request.
- The cost of a maintenance request is calculated from two fields: upfront cost and maintenance cost.

Budget

- A budget will be created and set for owing managers
- A budget will be tracked for each owing manager and will be impacted by maintenance requests
- A budget should show how much is left currently and estimated to be left
- An owing manager or admin will be able to generate budget reports based on the maintenance requests.

Non-functional Requirements

- The application will run using AWS services.
- The application will be usable in mobile and desktop view port sizes (responsive web design).
- The application complies with Lockheed Martin’s style guide.

User Requirements

Roles:

- Admin
- Owing Manager
- Directio
- Requester- anyone in the company

Every role can be considered an owner

Work Breakdown Structure (WBS)

- Hierarchical Task Breakdown
    - Epics: An overarching feature or category of stories (Ex: SCRUM, Design, Process)
    - Story: A unit of work that can reasonably be completed within the sprint.
        - Comprises of a user story and acceptance criteria when deemed helpful.
    - Tasks: Steps to complete the Story in an effective manner.
- Accountability
    - Team members are accountable for tasks based on their role.

        - Accountability entails ensuring that a story is being worked on and is on track to be completed in the sprint.
        - It does not mean that the team member accountable starts and completes the task (responsible).
    - Although there are roles for each team member, tasks aren't strictly divided by these roles.
    - Everyone on the team will jump in on stories and tasks based on their bandwidth on tasks when there is a need.

Communication and Stakeholder Management Plan

Communication Channels and Frequency

- Slack for day-to-day team communication.
- Email for formal announcements and external communication.
- Jira for tracking tasks and progress.
- Weekly sponsor meetings on Thursdays for project updates.

Communication frequency:

- Daily stand-up meetings before other scheduled meetings.
- Weekly 4-Up meetings on Thursdays.
- Sponsor meetings every other Thursday.
- Backlog refinement and Sprint Planning on Mondays.
- Retrospectives at the end of every other Sprint.

Stakeholder Identification and Analysis

- Stakeholders:
    - Primary Sponsor: Jaime Wightman
    - Sponsor: Gregg Doppelheuer
    - Sponsor: Ellie Stevens
    - Project Manager: Saranya Dadi
    - Scrum Master: Nikkia Theodule
    - Sponsor Communications Lead: Aby Tiet
    - Front End Lead: Annie Tiet
    - Back End Lead: Addo Davies Jr.
    - Website Manager: Annie Tiet
- Stakeholder Engagement Strategy
    - Sponsor-Team Meeting: Thursdays at 5:15 PM over Zoom
    - Sponsor-Team Email: Sent at least 24 hours prior to the meeting
    - Sponsor input on Backlog Refinement: Thursdays during the middle of a sprint
    - Sponsor Demos: Thursdays at the end of each sprint
    - Team Communication:
        - Slack
        - Email
        - SCRUM ceremonies