Project 2

PROJECT OVERVIEW

USE CASE:

The associate squad should create a custom Salesforce application based on a business of their choosing.

CORE REQUIREMENTS:

The following are minimum requirements for the project and will result in a 70% score if completed. Implementing peripheral requirements will add to this baseline to increase the overall score up to 100%. Projects represent most of performance evaluations in terms of weight so consider goals for peripheral requirements and further customizations carefully.

GIT:

- The team should maintain a group GitHub repo that implements sound branching strategies for feature development.

SDLC:

- While working on this project, the squad should maintain a Scrumban workflow using daily standups to discuss and keep all squad members apprised of everyone's status.
- A Kanban board should be utilized to track the team's tasks and progress throughout the project.
  - The Kanban board should be on the squad GitHub repo.
- Standup notes should be maintained in a markdown file on the repo.

PRESENTATION:

- The team should create and practice a two-part presentation alongside their squad mates.
  - Part one should be a carefully considered slide deck that details the following:
    - An introduction of the squad.
    - The provided use case.
    - The SDLC practices leveraged to complete it.

        - The tools used to complete the project.
        - Any portions of the project that do not lend themselves to the live demo.
    - Part two should be a live demo of the project for the audience.
- Following the demo, the squad should have a wrap up section that thanks the audience and offers a chance for questions.
- The presentation should last no longer than five minutes per member of the squad.

ERD:

- Prior to beginning work on the org, the team should create an ERD mockup of the org they intend to create. This original mockup should be uploaded to the squad repo.

Core Declarative Requirements

DATA MODEL:

- The data model should offer a reasonable interpretation of the selected business.
- Relationships, including Lookup & Master-Detail, should be carefully considered and implemented.

SECURITY:

- Org level security should be configured for at least two distinct profiles.
- Object level security should be configured for at least three distinct user types.
- Record level security & access should be configured for the organization, including Org-Wide Defaults, a customized Role Hierarchy, Sharing Rules, and the ability for Manual Sharing.
- Field level security should be configured.

DECLARATIVE UI (USER INTERFACE) CUSTOMIZATION:

- There should be at least one custom app which includes a custom home page.
- Custom and standard objects should have configured page layouts.

PROCESS AUTOMATION:

- There should be a minimum of one of each of the following:
    - Screen Flow
    - Before-save Flow
    - After-save Flow
    - Autolaunched Flow
- Flows should feature interaction with the database and should demonstrate an understanding of working with single and bulk records.

REPORTS & DASHBOARDS

- Use of both a Standard Report Type & Custom Report Type.

Core Programmatic Requirements

APEX TESTING

- The team should ensure that their Apex code has adequate test coverage – both individually and collectively – and that their tests follow testing best practices.

APEX TRIGGERS

- At minimum, each squad member should create an Apex trigger and associated handler that follows best practices and is relevant to the business use case.

PLATFORM EVENTS

- Collectively, the team should implement each of the following:
    - A Platform Event.
    - Apex code or a flow that fires an instance of the above Platform Event.
    - An Apex Trigger or Platform-Event Triggered Flow that is invoked by the reception of the above Platform Event.

VISUALFORCE

- At minimum, each squad member should create the following:
    - A Visualforce page relevant to the business use case.
    - Associated custom Apex code (either through a Custom Controller or Controller Extension).

COMMENTING

- Following good practices, ensure the code (Apex and Visualforce) is well documented, including:
    - Header comments that detail the name of the file, author(s), created date, last modified date, an overview of the purpose of the file.
    - Comments for each function outlining the purpose of the function.
    - Self-documenting code practices that include proper variable and function names.
    - Use of camelCase naming practices.

PERIPHERAL REQUIREMENTS:

While the core requirements will reach 70%, the team can go beyond that with their own customizations and ideas. Below are a list of peripheral items that can be implemented, but feel free to come up with original, unique additions as well.

SALES CLOUD:

- Leads & Opportunities should be customized as needed to reflect the business being modeled.
    - Leads should have multiple rule entries for both Assignment Rules & Auto-Response Rules.
    - Web-to-lead should be set up to allow a customer to submit a lead.
    - An Opportunity path should provide step by step guidance and a celebratory effect upon completion.

SERVICE CLOUD:

- Ensure cases are configured to accurately represent business needs.
    - Cases should have multiple rule entries for both Assignment Rules and Auto-Response Rules.
    - Web-to-Case should be set up to allow a customer to submit a case.
    - Case paths should offer step by step guidance on cases.

PROCESS AUTOMATION (PERIPHERAL)

- The following process automation tools should be implemented:
    - Workflow Rule
    - Process configured through Process Builder
    - Multi-step Approval Process
    - Schedule-Triggered Flow
    - Platform Event-Triggered Flow

EXPERIENCE CLOUD

- The squad should create a customer-facing Experience Cloud site.

INTEGRATION

- The team should make use of an External API by writing Apex code that consumes the selected webservice.
- The chosen API should be relevant to the business use case.
- The response from the API should be parsed and displayed to the user.

PACKAGE BASED DEVELOPMENT

- The team should implement Package Based Development by creating multiple unlocked packages from a single Dev Hub org with logical dependencies.
- The group should be able to demonstrate the successful installation of all unlocked packages.