PROJECT Design Documentation

> *The following template provides the headings for your Design Documentation. As you edit each section make sure you remove these commentary 'blockquotes'; the lines that start with a > character and appear in the generated PDF in italics.*

Team Information

- Team name: Sultans of Scrum
- Team members
  - Garrett Geyer
  - Michael DiBiase
  - Carla Lopez
  - Rachel Adkins
  - Cheyenne Zhang

Executive Summary

This is a summary of the project.

Purpose

> ***[Sprint 2 & 4]*** *Provide a very brief statement about the project and the most important user group and user goals.* The Mia foundation is a animal shelter dedicated to helping pets with disabilities. The manager is able to asses the needs of the shelter, and provide an up to date list of needs into the website to recieve funding from various helpers. Helpers can browse the available needs and can add to their funding baskets any needs they have an interest in and would like to support.

Glossary and Acronyms

> ***[Sprint 2 & 4]*** *Provide a table of terms and acronyms.*

Term	Definition
SPA	Single Page
Helper	A user using the application with intent to give support
Manager	A user using the application with intent to manage the organization

Requirements

This section describes the features of the application.

> *In this section you do not need to be exhaustive and list every story. Focus on top-level features from the Vision document and maybe Epics and critical Stories.*

Definition of MVP

> ***[Sprint 2 & 4]*** *Provide a simple description of the Minimum Viable Product.* Each user logs in with their username, the manager of the system can log in with the username 'admin' which is reserved for them.

The manager can add, remove and change needs stored in their cupboard to reflect the needs of the organization.

A helper can view that cupboard and select needs to add to their funding basket, which is hidden from the manager.

MVP Features

> A user logs in with a username, going to their page A U-fund Manager logs in with username 'admin' A Helper can see a list of needs A Helper can search through the list of needs A Helper can add/remove a need from their Funding Basket A Helper can Check-out and fund all needs in their Funding Basket A U-fund Manager can add, remove, and edit the list of needs A U-fund manager cannot see each user's funding basket

Enhancements

> ***[Sprint 4]*** *Describe what enhancements you have implemented for the project.*

Application Domain

This section describes the application domain.

Domain Model

> ***[Sprint 2 & 4]*** *Provide a high-level overview of the domain for this application. You can discuss the more important domain entities and their relationship to each other.*

Architecture and Design

This section describes the application architecture.

Summary

The following Tiers/Layers model shows a high-level view of the webapp's architecture. **NOTE**: detailed diagrams are required in later sections of this document. (*When requested, replace this diagram with your **own** rendition and representations of sample classes of your system*.)

The Tiers & Layers of the Architecture

The web application, is built using the Model–View–ViewModel (MVVM) architecture pattern.

The Model stores the application data objects including any functionality to provide persistance.

The View is the client-side SPA built with Angular utilizing HTML, CSS and TypeScript. The ViewModel provides RESTful APIs to the client (View) as well as any logic required to manipulate the data objects from the Model.

Both the ViewModel and Model are built using Java and Spring Framework. Details of the components within these tiers are supplied below.

Overview of User Interface

This section describes the web interface flow; this is how the user views and interacts with the web application.

> *Provide a summary of the application's user interface. Describe, from the user's perspective, the flow of the pages in the web application.*

View Tier

> ***[Sprint 4]** Provide a summary of the View Tier UI of your architecture. Describe the types of components in the tier and describe their responsibilities. This should be a narrative description, i.e. it has a flow or "story line" that the reader can follow.*

> ***[Sprint 4]** You must provide at least **2 sequence diagrams** as is relevant to a particular aspects of the design that you are describing. (**For example**, in a shopping experience application you might create a sequence diagram of a customer searching for an item and adding to their cart.) As these can span multiple tiers, be sure to include an relevant HTTP requests from the client-side to the server-side to help illustrate the end-to-end flow.*

> ***[Sprint 4]** To adequately show your system, you will need to present the **class diagrams** where relevant in your design. Some additional tips:*
>
> - *Class diagrams only apply to the **ViewModel** and **Model** Tier*
> - *A single class diagram of the entire system will not be effective. You may start with one, but will be need to break it down into smaller sections to account for requirements of each of the Tier static models below.*
> - *Correct labeling of relationships with proper notation for the relationship type, multiplicities, and navigation information will be important.*
> - *Include other details such as attributes and method signatures that you think are needed to support the level of detail in your discussion.*

ViewModel Tier

> ***[Sprint 4]** Provide a summary of this tier of your architecture. This section will follow the same instructions that are given for the View Tier above.*

> *At appropriate places as part of this narrative provide **one** or more updated and **properly labeled** static models (UML class diagrams) with some details such as critical attributes and methods.*

Replace with your ViewModel Tier class diagram 1, etc.

Model Tier

> ***[Sprint 2, 3 & 4]** Provide a summary of this tier of your architecture. This section will follow the same instructions that are given for the View Tier above.*

> *At appropriate places as part of this narrative provide **one** or more updated and **properly labeled** static models (UML class diagrams) with some details such as critical attributes and methods.*

Replace with your Model Tier class diagram 1, etc.

OO Design Principles

> ***[Sprint 2, 3 & 4]*** *Will eventually address upto* ***4 key OO Principles*** *in your final design. Follow guidance in augmenting those completed in previous Sprints as indicated to you by instructor. Be sure to include any diagrams (or clearly refer to ones elsewhere in your Tier sections above) to support your claims.*

Principles Chosen for Sprint 2: High Cohesion Single Responsibility

> ***[Sprint 3 & 4]*** *OO Design Principles should span across* ***all tiers.***

Static Code Analysis/Future Design Improvements

> ***[Sprint 4]*** *With the results from the Static Code Analysis exercise,* ***Identify 3-4*** *areas within your code that have been flagged by the Static Code Analysis Tool (SonarQube) and provide your analysis and recommendations.*
> *Include any relevant screenshot(s) with each area.*

> ***[Sprint 4]*** *Discuss* ***future*** *refactoring and other design improvements your team would explore if the team had additional time.*

Testing

> *This section will provide information about the testing performed and the results of the testing.*

Acceptance Testing

> ***[Sprint 2 & 4]*** *Report on the number of user stories that have passed all their acceptance criteria tests, the number that have some acceptance criteria tests failing, and the number of user stories that have not had any testing yet. Highlight the issues found during acceptance testing and if there are any concerns.*
> 19 user stories will be finished.

Unit Testing and Code Coverage

> ***[Sprint 4]*** *Discuss your unit testing strategy. Report on the code coverage achieved from unit testing of the code base. Discuss the team's coverage targets, why you selected those values, and how well your code coverage met your targets.*

> ***[Sprint 2 & 4] Include images of your code coverage report.*** *If there are any anomalies, discuss those.*

Use of Postman API Platform for Sprint 1 RestAPI Demo

> The Postman API Platform will be used to manage and document the Restful API's of our application. Postman is a development tool which helps build, test and modify API's. Postman offers a usr-friendly graphical interface that makes it easier for the team of developers, as well as the product owner, to interact with API's. The team "Sultans of Scrum" has a "workspace" in which the Restful API's of our applciation will be stored. The workspace facilitates collaboration in the team by allowing us to share our collection of API's and maintain a record of them.