

ANIMAL AUXILIARIEs 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: 2231-swen-261-05-2b-animal-auxiliaries
- Team members
 - Chase Balmer
 - Sarah Payne
 - Marisa Ortiz
 - Tyler Combs
 - Tszfai Choy

Executive Summary

This project is a U-Fund website that is to be used by a local animal shelter/humane society. This website provides tools for a welcoming and supportive community with the purpose of helping homeless animals. Managers run the needs page, indicating the current needs of the organization. This may include funding equipment purchases, purchasing food, etc. Helpers may view these needs and contribute through funding. We hope that our product will enable members of our community to help these innocent animals and provide them with the things they need, or even a home. The websites backend is built in Java-Spring, the frontend with Angular.

Purpose

[Sprint 2 & 4] *Provide a very brief statement about the project and the most important user group and user goals.*

The Animal Auxiliaries U-Fund website exists to allow an animal shelter to virtually fundraise for their non-profit by making the process of donating to the animal shelter simple and easy.

Glossary and Acronyms

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

Term	Definition
Need	A single item or action that the non-profit needs to be funded. Needs are created by <i>Managers</i> and funded by <i>Helpers</i> .
Cupboard	Virtual storage for the many needs of the non-profit. The cupboard is edited by <i>Managers</i> and viewed by <i>Helpers</i> .
Funding Basket	Virtual shopping basket. Each <i>Helper</i> has a personal Funding Basket, they can add and remove <i>Needs</i> from their Funding Basket as well as Check-out or buy those <i>Needs</i> .

Term	Definition
Manager	A website User that represents a person who is responsible for the needs of a non-profit. The Manager has the ability to edit the <i>Cupboard</i> .
Helper	A website User that represents a person who is donating to the non-profit. The Helper has the ability to search the <i>Cupboard</i> and fund <i>Needs</i> .

Requirements

Webpage displaying and enabling all of the following things:

Authentication for Helper/U-fund Manager login & logout will enable necessary privledges for a user.

Helpers have the following abilities: Helper can see list of needs Helper can search for a need Helper can add/remove an need to their funding basket Helper can proceed to check-out and fund all needs they are supporting

Managers have the following abilities: Manager(s) can add, remove and edit the data of all their needs stored in their needs cupboard U-fund Manager cannot see contents of funding basket(s)

Data Persistence The system saves all information to files so that changes are reflected for all users.

Your 10% additional feature enchantment(s) Adoption Site Community Board

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.

- **Minimal Authentication** --> Minimal Authentication allows any user to login to the website without creating an account. Any user attempting to login with the **admin** username will be logged in as a *Manager*. Any user attempting to login with any other username will be logged in as a *Helper*.
- **Helper Functionality** --> Minimal Helper Functionality allows the *Helper* to **search** for a *need*. As well as **Add** and **Remove** *Needs* from their *Funding Basket*.
- **U-fund Manager Functionality** --> Minimal Manager Functionality allows a *Manager* to **add**, **remove**, and **edit** the *needs* in the *cupboard*. And prevent the *Manager* from viewing the *funding baskets*.

MVP Features

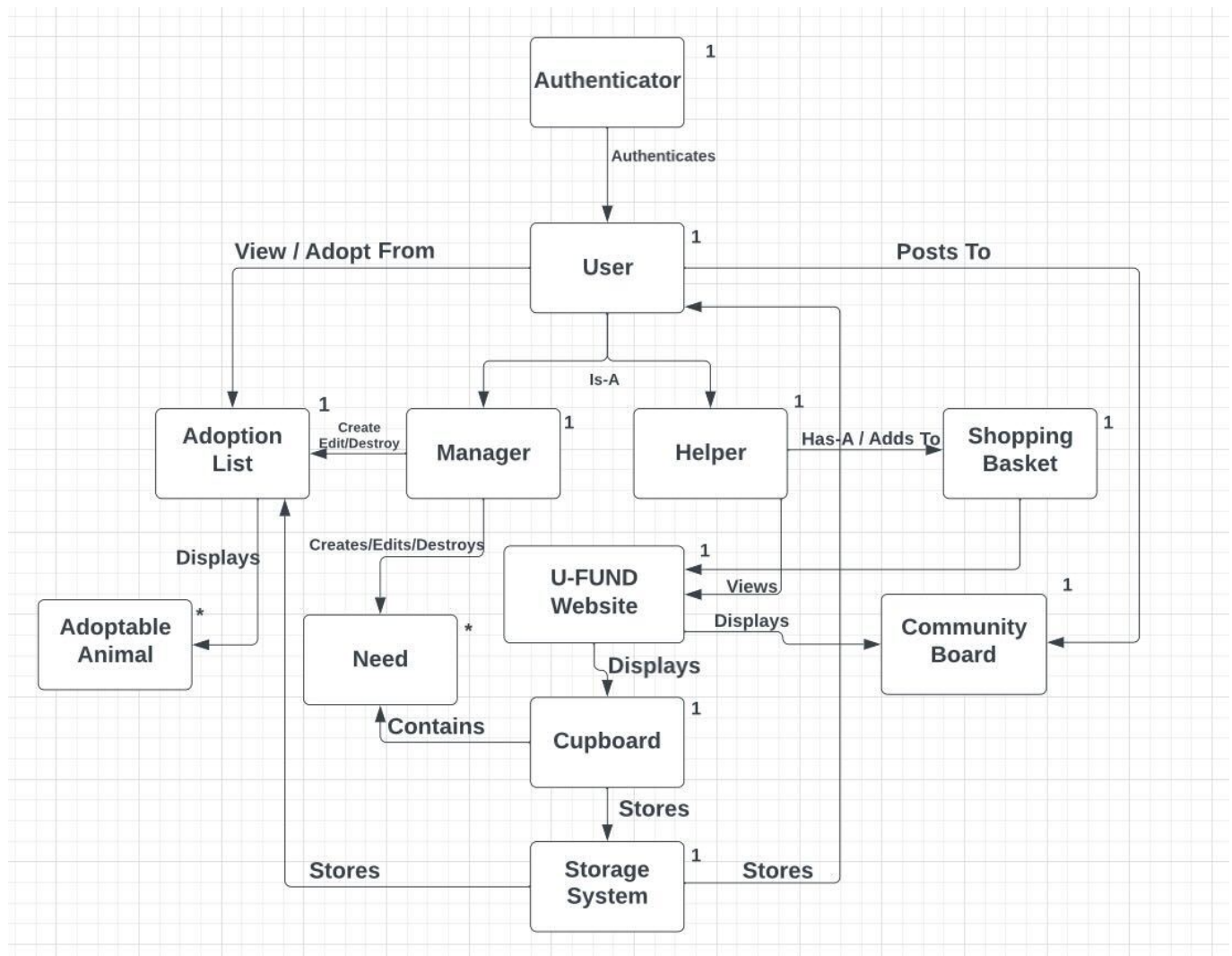
[Sprint 4] Provide a list of top-level Epics and/or Stories of the MVP.

Enhancements

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

Application Domain

This section describes the application domain.



[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.

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 persistence.

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.

[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.

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.