**1. Project Title, and Group Name:**

**Project Title:** Luna’s List

**Group Name:** Software Solution Ltd.

**2. Project Website/Github URL:**

**Github URL:** [**https://github.com/Reiterpallasch/4882-Capstone**](https://github.com/Reiterpallasch/4882-Capstone)

**3. Version:**

**Version:** 1.0

**4. Project Summary:**

**4.1 High-Level Overview:**

Luna’s List is intended to be a platform in which canine lovers may discover and share locations that are dog friendly. In doing so, the platform will allow for the rules of the location to be well laid out, such as if aggressive dogs are allowed, or if a leash need be employed. The platform will also allow for those of the canine affinity to come together in varying ways, whether it be casual or formal. To sum it up simply, one may view it as an initially low-level canine lover’s social network with great potential for expansion. Initially it is to be web based, however, mobile expansion is within its grasp.

**4.2 Goals:**

The goal of this project is to bring information to dog owners in a unified, central location. Dog owners will be able to see tips and tricks on maintaining their dog’s health and behavior. Owners will also be add and see parks, vets, stores and restaurants that allow dogs to come as well as location rules, and a forum to discuss varying topics not covered elsewhere. Admins will be able to view and approved submitted information by users, or be able to add information themselves. Pages will also be able to display maps pinging relevant locations to the page that are near to the user on the page at the time.

**5. Problem Statement:**

**5.1 Original Customer Problem Statement:**

Dogs have been interwoven into our lives at a scale never seen before. They have moved outside of the perimeter of our homes and yards and into our roads, stores and entertainment areas.

More and more questions have flooded businesses, parks, and buildings on when, where and how can I bring my dog?

Lunas List will help answer these questions and more by providing a platform that provides communities with Dog friendly spaces.  It will also bring Dog Owners together in different ways.

**5.2 Expanded Problem Statement:**

Dogs have been interwoven into our lives at a scale never seen before. They have moved outside of the perimeter of our homes and yards and into our roads, stores and entertainment areas. As more dog owners seek to bring their dogs out on more than just walks and car rides: businesses, parks, and buildings have begun to reciprocate by offering themselves as dog friendly places. However, although this is occurring, questions have begun to arrive from these dog owners about when, where, and how they can bring their dogs. These questions that grow more and more make it inconvenient for dog owners to be able to always stay updated on where they can go. This is where Luna’s List comes in. Luna’s List strives to answer the questions that dog owners have by creating a platform that tells its users about upcoming events, dog friendly businesses, and provide tips that will better enable dog owners to prepare them and their dogs to go out to different places. Luna’s List will let dog owners connect with each other and participate in providing information about events and businesses that surround them that are dog friendly and among another range of items will serve to provide: a forum where dog owners can discuss dog habits, doggy trainers that they can go to, an intermediate for adopting dogs or puppies, and other general discussions. Luna’s List will also provide general dog tips such as care, etiquette, and news.

**6. Team Profile:**

**6.1. Team Member Profile:**

Wesley Jones:

My goal is to implement software design and programming into two additional that I posses in Biomedical engineering. I wish to apply lessons learned into building great team foundations, and working towards improving products such as healthcare monitoring devices and prosthetic interfacing. My primary function on this project has been implementing major backend website functions that allow for data addition and display, as well as administrator control. I have also served as Scrum Master, keeping meetings on focus and assisting with team issues that present themselves during the project design and implementation. MY major strengths are quickly understanding a problem and working to find a viable solution, as well as being able to quickly pick up a programming language.

Larz Leonard:

My goals in starting a career is to program and eventually to get into game development and I am very interested in programming and improving AI. For this project, my role is a developer and my strengths are programming in Java, Python, HTML, and Ruby, with Ruby being the primary focus behind the coding here. Another aspect of my strengths is brainstorming different ideas for how to tackle a problem to determine what will work and what will not.

Thi Phan:

My strengths revolve around development in Python, Java. I know some about Web development with Django, Ruby on Rails and iOS development with Swift. For this project, I will work on the back end and some front end aspects.

Vinh Tran:

My main understanding in programming is Java and C/C++ with a minor amount of knowledge in SQL. In this project my role serves as a developer, helping to formulate the issues/tasks that the team requires to build a successful product.

Noah Hanks:

My career goals are less focused on programming and more concentrated on other aspects of the information technology field. More specifically, I would like to pursue a career either in systems administration or possibly digital forensics. Lately, my interests have primarily revolved around building out a “homelab” which is a hobby focused on setting up and using enterprise hardware and software in a home network. While my career goals may not be programming focused, I have experience coding in the following languages: Python, C++, SQL, and Ruby on Rails. In this project, I am filling the roles of developer and product owner. As the product owner, I am responsible for handling all communication with the customer and making sure that this information is conveyed correctly to the rest of the team. As a developer, I am responsible for completing assigned tasks on-time and to the best of my ability.

**6.2 Roles and Responsibilities:**

**Scrum Master:**

Wesley Jonesis our scrum master. He is a keeper of scrum process. Wesley is tasked with leading daily scrum and running the scrum planning meeting. In doing so, it is ensured that the topics stay on track and important issues are discussed as necessary.

**Product Owner:**

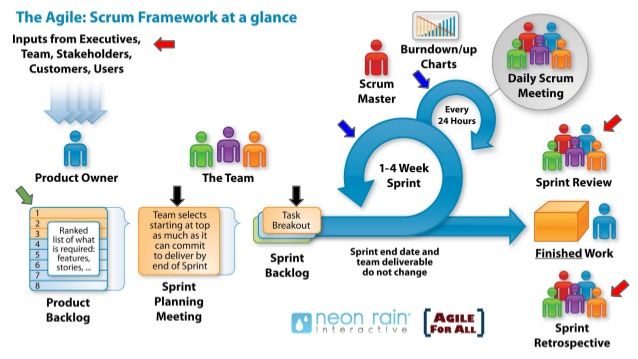
Noah Hanks is Product Owner. He is the key member to communicate with customer and responsible for writing user stories or help others to write user stories.

**Developers:**

Every member has a developer role. Each Team Member is typically in charge of developing a new feature for the website and make sure their codes work.

**7. Development Process:**

**7.1 Agile Scrum Process:**

****

***Figure 1. Agile Scrum Process***

The Agile Scrum method is being implemented to perform this project. Thus far, Sprint planning has consisted of determining what is to be accomplished during the sprint, and scrum meetings used to discuss where the team was in finishing the tasks as well as discuss concerns. We initially began with brainstorming on our product backlog, creating a list of both functional and non-functional user stories that we felt embodied the scope of the project. Once we had our basis, we broke these stories down further into smaller components. Presenting the overall plan to the customer and understanding his needs, we then broke down the current backlog and selected what was needed that would both fit our timeline effectively and match the customer needs. Values to the user stories were determined via a consensus to the effort required for completion. We also decided that we would meet with the customer before and after each sprint, and hold scrum meetings twice a week, with external communication available if needed. Each scrum meeting consisted of ensuring that we understood where we were in regards to completion of tasks, or discuss changes that needed to be implemented to complete a task. We also provided assistance to one another if it was needed. We ended with a retrospective to determine what went well, as well as what we can do better, which is detailed in section 9.

**7.2:**

For our website, we made use of Ruby 2.5.1, Rails 5.2.2, client-side JavaScript, and HTML/CSS. For CSS, we’re additionally using Bootstrap to create a responsive, clean design. We use sqlite3 for database management as it is portable, easy to manage, and straightforward in making changes or pulling information.

**7.3 Communication Method:**

Communication consists of Scrum meetings, Slack channels, Email, and Github. These platforms were felt best to effectively and consistently maintain communication flowing in a manner that all team members are up to date. Slack particularly allowed team members to communicate outside of scrum meeting and implement a system to alert users when a commit was made and pushed to Github.

**8. Glossary of Terms:**

Mobile – Platform description for phones and tablets.

Model - Database table for storing values.

Admin - System administrators capable of reviewing, editing, or deleting added table values.

Routes - File containing paths to pages necessary to reach each page of the website

Controller - File containing primary ruby code that controls how a page functions and calls the associated model for use. It also defines what values are permitted to be added to each database model.

Views - Sub-folders containing website pages. A hierarchical structure allowing for multiple pages of similar name to be implemented and defined in the routes file without conflicting naming schemes.

Bootstrap - An API that allows for graphical enhancement of website pages and layouts.

Gem - A packaged ruby API containing code to allow for functions and features to be implemented.

GUI - Graphical User Interface.

Home - The central location from which the rest of the website structure may be accessed.

Sitemap - A hand drawn description of the website that shows how to reach and page, and the pages primary function.

Index - A page dedicated to displaying all model information. Essentially lists entries in the model.

Registration - The page dedicated to creating a user.

User - A standard person who wishes to have an account on the website and the ability to add information to the models for approval.

UML Diagram - An overarching Database diagram that depicts what each table/model is supposed to contain, and how it is related to another model.

Relation - Describes how a model is related to another such as one to one, one to many, or many to many.

rb - File extension denoting ruby code.

**9. Results:**

**9.1 Requirements, Analysis, Design:**

**9.1.1 Product Backlog:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **User Story** | **Priority** | **Functional/Non-Functional** |
| LL-14 | As a user, I would like to have a list of vet locations in my area. | Highest | Functional |
| LL-15 | As a user, I would like to view relevant events in my area or search by city. | Highest | Functional |
| LL-16 | As a user, I would like to view approved restaurants in my area or search by city. | Highest | Functional |
| LL-17 | As a user, I would like to view park locations in my area or nearby or search by city. | Highest | Functional |
| LL-18 | As a user, I would like to have a profile page | Highest | Non-Functional |
| LL-24 | As a user, I would like to rate vets through a review. | Lowest | Functional |
| LL-25 | As a user, I would like to upload a profile picture to my profile. | Low | Functional |
| LL-26 | As a user, I would like to add my dog/dogs information. | Medium | Functional |
| LL-28 | Format the home page | Medium | Non-Functional |
| LL-29 | Format the Vets page | Medium | Non-Functional |
| LL-30 | Format the restaurants page | Medium | Non-Functional |
| LL-31 | Format the parks page | Medium | Non-Functional |
| LL-32 | Implement Forum | Medium | Non-Functional |
| LL-38 | As a user, I would like to upload pictures of my dogs | Low | Functional |
| LL-39 | As a user, I would like to rate parks through a review. | Lowest | Functional |
| LL-40 | As a user, I would like to rate restaurants through a review. | Lowest | Functional |
| LL-41 | As a user, I would like to rate events through a review. | Lowest | Functional |

Note that the backlog continues to expand as we further understand the customer’s desires and we brainstorm on features not initially mentioned as the project is built as well as further break down stories. Modification of the problem statement as can be seen in the problem statement section does occur.

**9.1.2 Functional Requirements:**

**User Interface Functional Requirements:**

Navigation Bar – Located permanently at the top of the website with with links to the following pages: Home, Stores, Events, Parks, Restaurants, Vets, and Login/Register.

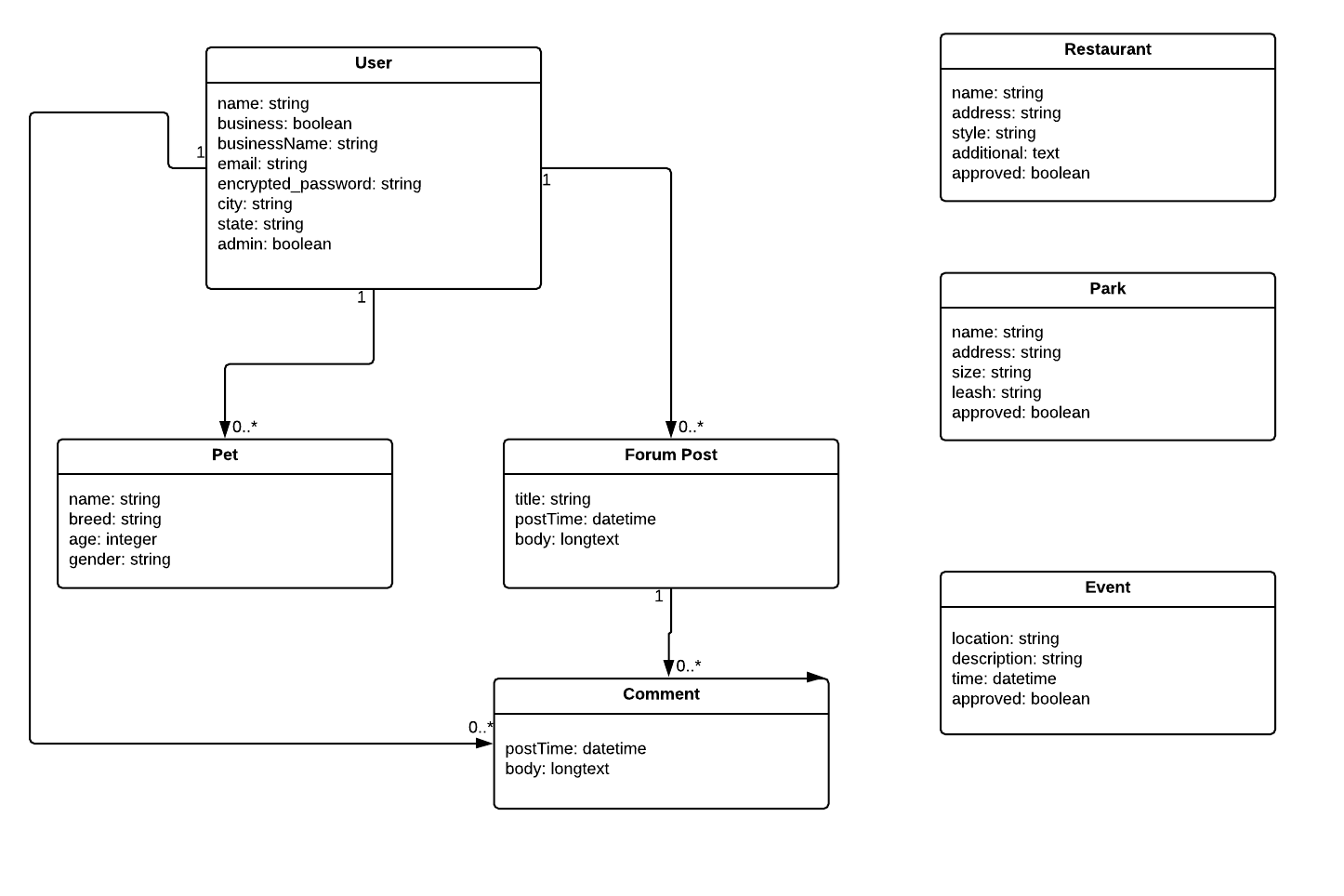
Home Page – Landing page for the website containing a welcome message and links to sign in

Stores Page – List of local businesses that have been approved by a site admin to be displayed publicly

Events Page – List of approved events in a users area

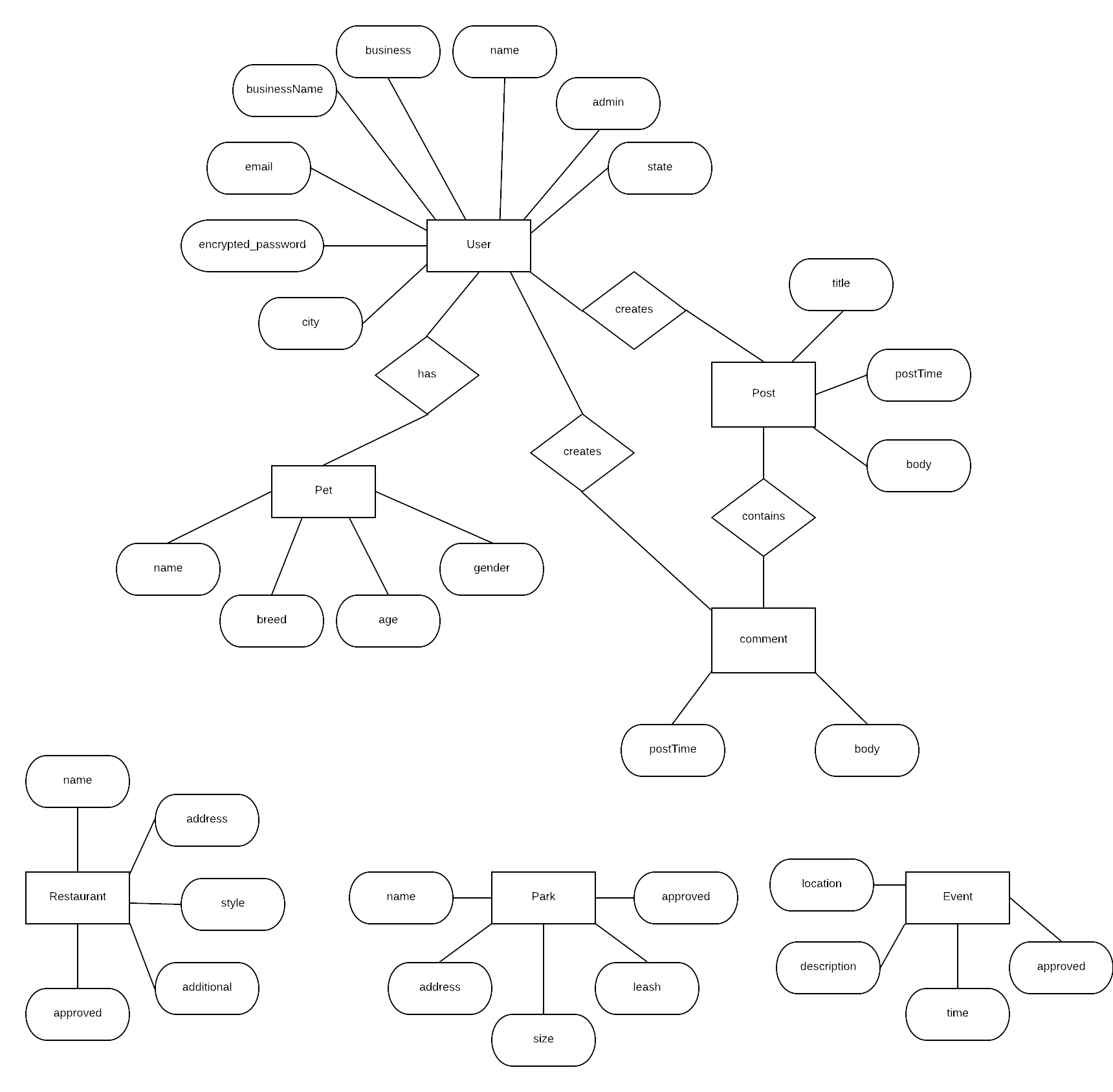
Restaurants Page – List of approved restaurants in a users area

Vets Page – List of approved veterinarians in a users area



**Figure 1:** Class Diagram

This class diagram outlines the database models for the application as well as how information is stored inside of them.



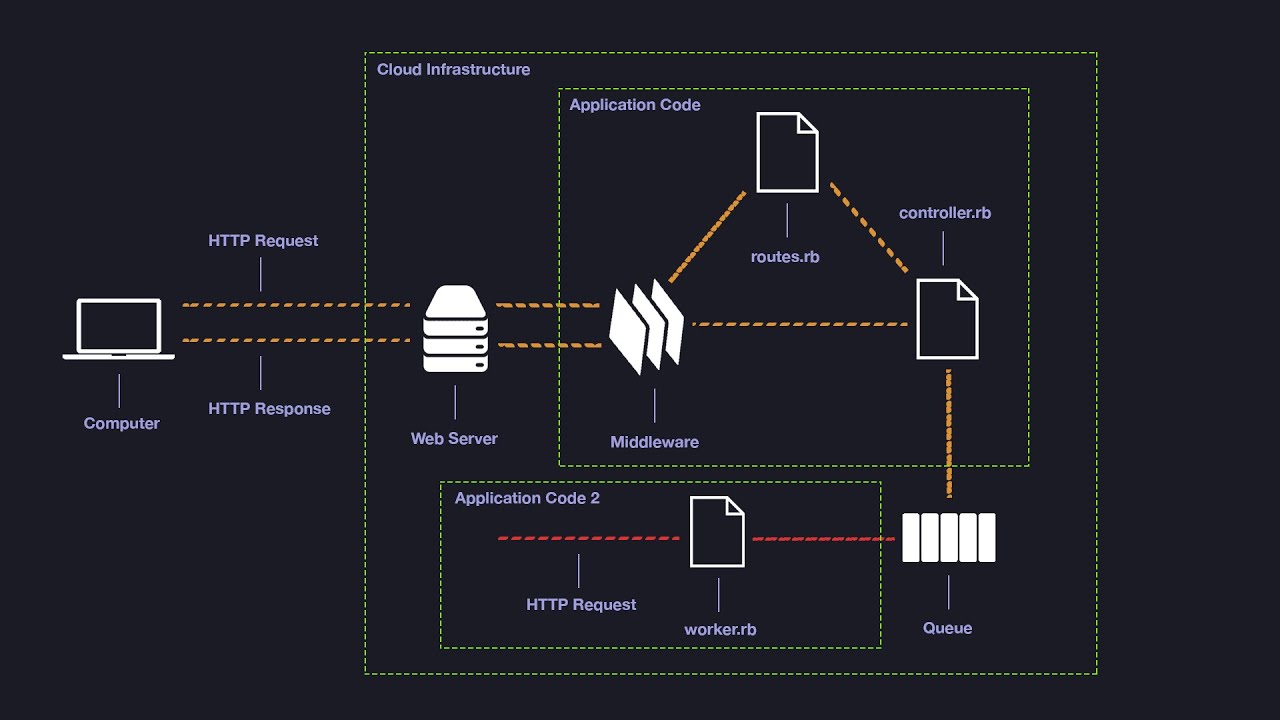
**Figure 2:** Database Schema

The database currently consists of seven tables to meet the Sprint 1 goals: User, Pet, Post, Comment, Restaurant, Park, and Event. A user can add a pet profile to be displayed with their user profile. A user can also make a forum post or comment on another forum post. They can also view information about approved restaurants, parks, and events in their area.

**9.1.3 Non Functional Requirement / Specification:**

**9.1.3.1 Base System:**

The project is based on a ruby on rails server. We are using Ruby 2.5.1 and Rails 5.2.2 for the most up to date features, with the exception of sqlite3 in that the latest 1.4.0 version is broken, therefore, 1.3.6 needed to be in use.



***Figure 2. Rails Architecture***

**9.1.3.2 Source Code Structure:**

Our source code file structure is as follows:

lunasList:.

│ .gitignore

│ .ruby-version

│ config.ru

│ Gemfile

│ Gemfile.lock

│ package.json

│ Rakefile

│ README.md

│ tree.txt

├─app

│ ├─assets

│ │ ├─config

│ │ │ manifest.js

│ │ ├─images

│ │ │ .keep

│ │ │ pawprint.jpg

│ │ │ pawprint2.jpg

│ │ ├─javascripts

│ │ │ │ application.js

│ │ │ │ cable.js

│ │ │ │ events.coffee

│ │ │ │ parks.coffee

│ │ │ │ restaurants.coffee

│ │ │ │ stores.coffee

│ │ │ │ vets.coffee

│ │ │ └─channels

│ │ │ .keep

│ │ └─stylesheets

│ │ application.scss

│ │ events.scss

│ │ parks.scss

│ │ restaurants.scss

│ │ stores.scss

│ │ vets.scss

│ ├─channels

│ │ └─application\_cable

│ │ channel.rb

│ │ connection.rb

│ ├─controllers

│ │ │ application\_controller.rb

│ │ │ events\_controller.rb

│ │ │ pages\_controller.rb

│ │ │ parks\_controller.rb

│ │ │ pets\_controller.rb

│ │ │ restaurants\_controller.rb

│ │ │ stores\_controller.rb

│ │ │ vets\_controller.rb

│ │ └─concerns

│ │ .keep

│ ├─helpers

│ │ application\_helper.rb

│ │ events\_helper.rb

│ │ parks\_helper.rb

│ │ restaurants\_helper.rb

│ │ stores\_helper.rb

│ │ vets\_helper.rb

│ ├─jobs

│ │ application\_job.rb

│ ├─mailers

│ │ application\_mailer.rb

│ ├─models

│ │ │ application\_record.rb

│ │ │ event.rb

│ │ │ park.rb

│ │ │ pet.rb

│ │ │ restaurant.rb

│ │ │ store.rb

│ │ │ user.rb

│ │ │ vet.rb

│ │ └─concerns

│ │ .keep

│ └─views

│ ├─events

│ │ edit.html.erb

│ │ index.html.erb

│ │ new.html.erb

│ │ show.html.erb

│ │ \_form.html.erb

│ ├─layouts

│ │ application.html.erb

│ │ application.html.erb\_old

│ │ mailer.html.erb

│ │ mailer.text.erb

│ ├─pages

│ │ home.html.erb

│ ├─parks

│ │ edit.html.erb

│ │ index.html.erb

│ │ new.html.erb

│ │ \_form.html.erb

│ ├─restaurants

│ │ edit.html.erb

│ │ index.html.erb

│ │ new.html.erb

│ │ \_form.html.erb

│ ├─stores

│ │ edit.html.erb

│ │ index.html.erb

│ │ new.html.erb

│ │ show.html.erb

│ ├─users

│ │ ├─confirmations

│ │ │ new.html.erb

│ │ │

│ │ ├─mailer

│ │ │ confirmation\_instructions.html.erb

│ │ │ email\_changed.html.erb

│ │ │ password\_change.html.erb

│ │ │ reset\_password\_instructions.html.erb

│ │ │ unlock\_instructions.html.erb

│ │ ├─passwords

│ │ │ edit.html.erb

│ │ │ new.html.erb

│ │ ├─registrations

│ │ │ edit.html.erb

│ │ │ new.html.erb

│ │ ├─sessions

│ │ │ new.html.erb

│ │ ├─shared

│ │ │ \_error\_messages.html.erb

│ │ │ \_links.html.erb

│ │ └─unlocks

│ │ new.html.erb

│ └─vets

│ edit.html.erb

│ index.html.erb

│ new.html.erb

│ \_form.html.erb

├─bin

│ bundle

│ rails

│ rake

│ setup

│ spring

│ update

│ yarn

├─config

│ │ application.rb

│ │ boot.rb

│ │ cable.yml

│ │ credentials.yml.enc

│ │ database.yml

│ │ environment.rb

│ │ puma.rb

│ │ routes.rb

│ │ spring.rb

│ │ storage.yml

│ ├─environments

│ │ development.rb

│ │ production.rb

│ │ test.rb

│ ├─initializers

│ │ application\_controller\_renderer.rb

│ │ assets.rb

│ │ backtrace\_silencers.rb

│ │ content\_security\_policy.rb

│ │ cookies\_serializer.rb

│ │ devise.rb

│ │ filter\_parameter\_logging.rb

│ │ inflections.rb

│ │ mime\_types.rb

│ │ wrap\_parameters.rb

│ └─locales

│ devise.en.yml

│ en.yml

├─db

│ schema.rb

│ seeds.rb

├─lib

│ ├─assets

│ │ .keep

│ └─tasks

│ .keep

│ auto\_annotate\_models.rake

├─log

│ .keep

├─public

│ 404.html

│ 422.html

│ 500.html

│ apple-touch-icon-precomposed.png

│ apple-touch-icon.png

│ favicon.ico

│ robots.txt

├─storage

│ .keep

├─test

│ │ application\_system\_test\_case.rb

│ │ test\_helper.rb

│ ├─controllers

│ │ .keep

│ │ events\_controller\_test.rb

│ │ parks\_controller\_test.rb

│ │ restaurants\_controller\_test.rb

│ │ stores\_controller\_test.rb

│ │ vets\_controller\_test.rb

│ ├─fixtures

│ │ │ .keep

│ │ │ events.yml

│ │ │ parks.yml

│ │ │ pets.yml

│ │ │ restaurants.yml

│ │ │ stores.yml

│ │ │ users.yml

│ │ │ vets.yml

│ │ └─files

│ │ .keep

│ ├─helpers

│ │ .keep

│ ├─integration

│ │ .keep

│ ├─mailers

│ │ .keep

│ ├─models

│ │ .keep

│ │ event\_test.rb

│ │ park\_test.rb

│ │ pet\_test.rb

│ │ restaurant\_test.rb

│ │ store\_test.rb

│ │ user\_test.rb

│ │ vet\_test.rb

│ └─system

│ .keep

├─tmp

│ .keep

└─vendor

.keep

**9.1.3.3 Installation Procedure:**

For our system, there are two options for installation. The first option is to install a pre-configured virtual machine in which the user needs to simply clone the Github repository and navigate to the lunasList folder. From inside, the commands ‘bundle install’, ‘rake DB:migrate’, ‘rake DB:SETUP’, and ‘rails s -b 0.0.0.0’ can be run to have a local version running, reachable by typing ‘localhost:3000’ into a browser navigation bar.

The other option is to download a Debian based variant of Linux, our suggested version being Mint. The commands that follow may vary a little from distribution to distribution and are designed for use Linux Mint 19. After installation the following commands should be run in a terminal to ensure all necessary items needed are installed.

sudo apt-get update

sudo apt-get install ruby-full

sudo gem update

sudo gem update --system

sudo apt-get install ruby-dev zlib1g-dev liblzma-dev

sudo gem install rails

sudo apt-get install libsqlite3-dev

sudo apt-get install nodejs

Once these commands and or variations needed for the Linux installation have been run, the first installation variant instruction can be continued at cloning the Github repository. Video installation instructions are also available at:

Linux Setup - https://youtu.be/gkEwZndLZc0

Ruby on Rails Setup - https://youtu.be/Oh8GrPInDMQ

**9.1.4 Other specific requirements:**

In order to have this project work an active and stable internet connection is needed. For final deployment, a dedicated and port-forwarded server will be the ideal location to host the web app.

Administrators will be required to be put in place as data submitted by users must be reviewed and approved to ensure accuracy.

Ruby 2.5.1

Rails 5.2.2

Any modern web browser can access the website.

**9.2 Development planning results:**

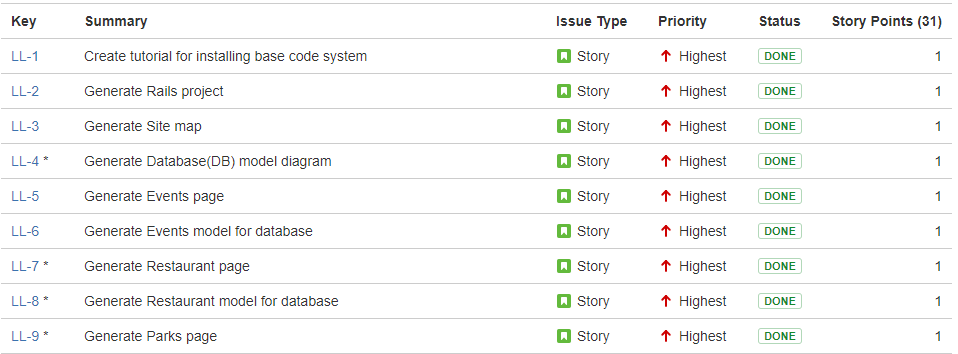
**9.2.1 Sprint planning summary:**

The project began with the team thinking on what core items a website using ruby on rails needed as well as what a user might like to see and what was necessary to be implemented via reading the customer’s initial problem statement, yielding our backlog. We then proceeded to select what user stories we felt were necessary to get the project on its feet. We followed this up with corresponding with the customer to ensure that our selection fell in line with what the customer wanted to see as well in each sprint and ensure proper prioritization. We are reliant on the method of contacting the customer to have final say on what goes in to each sprint.

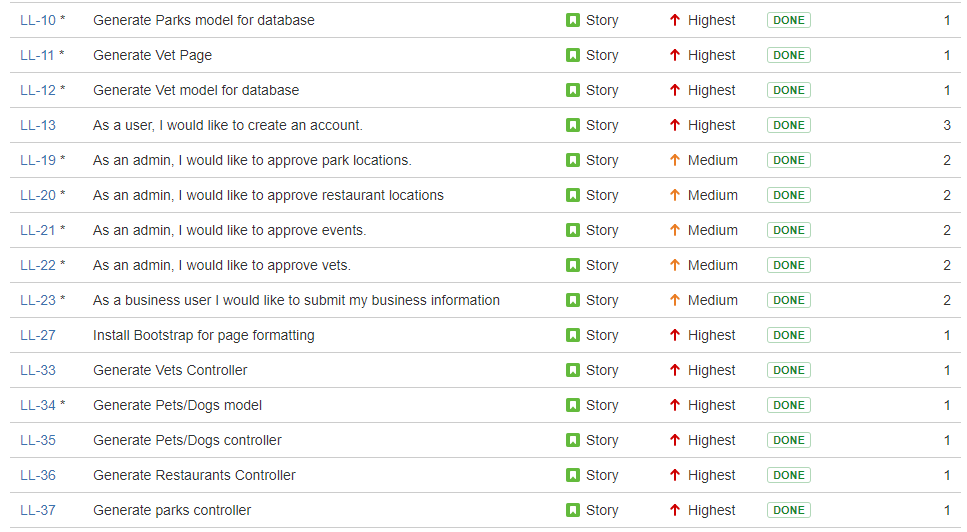
Thus far, our method seems to be yielding positive results in that during the first sprint, all goals have been met to our satisfaction.

**9.2.2 Roadmap from Sprint 1 to Final Sprint:**

**Sprint 1:** The focus of Sprint 1 was to set up the backbone of the project. The idea was to make sure that we had a working system that met the most important of the customer needs. We needed to have the databases for dog friendly locations and events in place, as this is the major theme of the site. We also needed for users to have the ability to add information as being that more places become open to the concept of allowing our canine friends to join, they may not immediately make that information available, thus community aid in gaining this information is needed. These items are also the core of what we needed to proceed further in the project. Below is the list of tasks implemented. We did complete all tasks for this sprint.



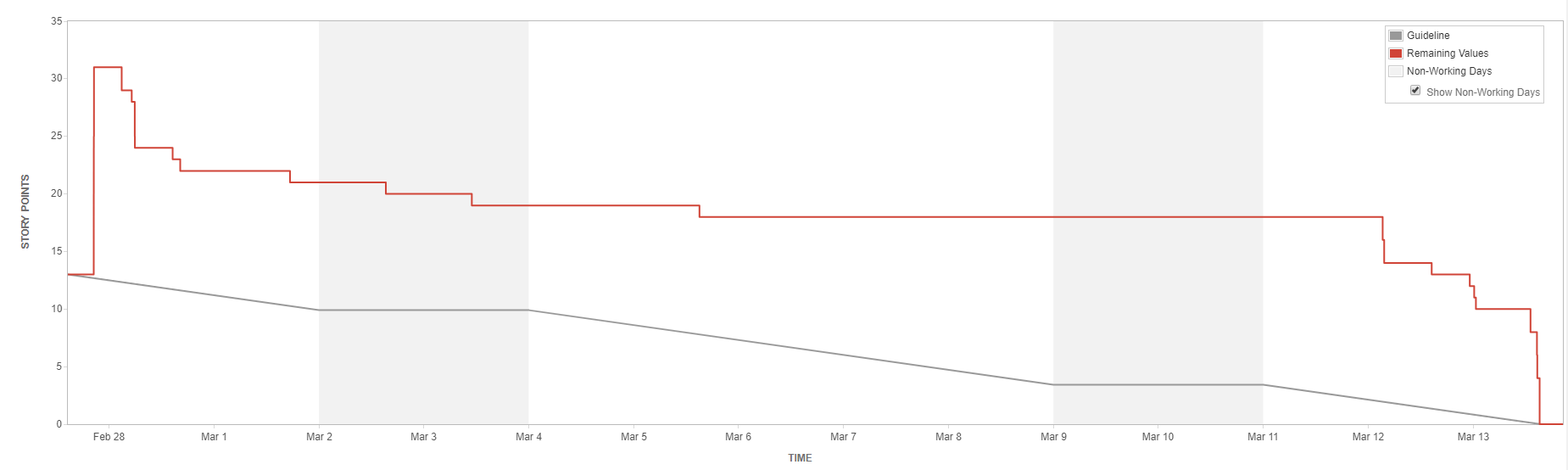
***Figure 3. Sprint 1 Issues***

******

***Figure 4. Sprint 1 Issues Continued***

**9.2.3 Sprint Burn Down Chart History:**

**Sprint 1:**

****

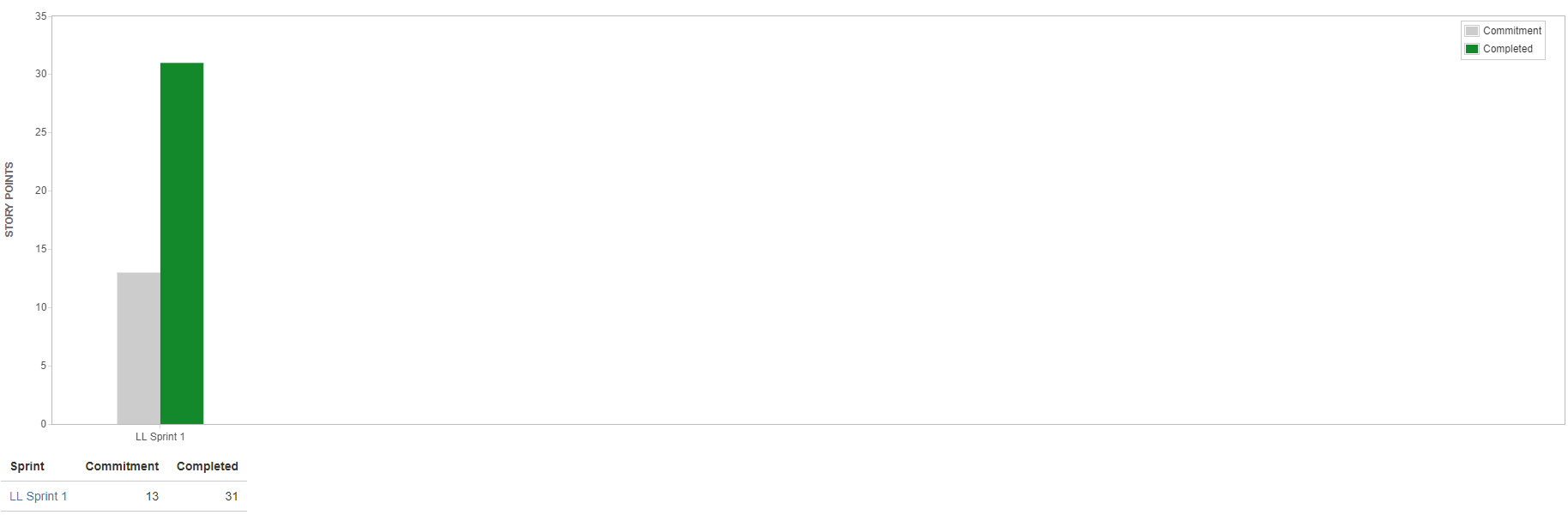
***Figure 5. Sprint 1 Burndown Chart***

**9.2.3.1 Sprint 1 Burndown Chart Description:**

As one can see from the chart, our sprint goals were met. It will be noticed that there is a long period of inactivity, however, that is due to a JIRA software outage during which issues could not be moved from in progress to complete. Near the end, it can also be seen that a fair amount of work crunching went in to play to complete the remaining tasks. It should also be noted that at the start of the sprint, several key stories were added in after the start of the sprint, hence the large rise of remaining values relative to the guideline.

**9.2.4 Sprint velocity and hours:**

**9.2.4.1 Sprint 1 Velocity Chart:**

****

***Figure 6: Sprint 1 Velocity Chart***

We can see from the velocity chart that we well exceeded the Sprint 1 commitment. This does come from the fact that User Stories were added to the sprint in an ineffective manner as we further discussed the needs of the customer for sprint 1. The chart does otherwise show that we were well adapted and handled the necessary changes to ensure customer happiness and sprint completion.

**9.3 Risk Tracking Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk** | **Severity** | **Likelihood** | **Steps to Avoid** | **Outcome** |
| Installation Incompatibility across different systems | HIGH | HIGH | Create a single known working system, and create an iso of said system | PASS |
| Ruby Gem version issues | HIGH | MEDIUM | Test aspects that the gems control to ensure effectiveness | PASS |
| Users are unable to save added data | MEDIUM | LOW | Ensure ruby code is accurate and perform tests | PASS |
| Admins are unable to edit, approve, or delete additions | MEDIUM | LOW | Ensure ruby code is correct and run tests | PASS |

**9.4 Summary of Retrospectives:**

**9.4.1 Retrospectives:**

**Sprint 1:**

Overall Sprint 1 feels to have gone smoothly. We did encounter a major hiccup with the JIRA software, affecting the overall look of the Sprint Burndown Chart, but said issue was quickly resolved upon notification. We could handle to use more tasks as it seems the User Stories were completed rather quickly. We do need to ensure that we can fix some stories relying on others to be completed, however, in looking back at sprint 1 and its theming, it seems an unavoidable issue at this early state. We have not yet heard back from the customer regarding a demonstration and our planned setup for meetings. All team members seem to have been on track and understood well, what needed to be done. We did notice that we had to add User Stories after starting the sprint, something we need to fix.

**9.4.2 Table of Retrospectives:**

|  |  |  |  |
| --- | --- | --- | --- |
| Sprint # | What did we do well and should carry on doing? | What did we do that we should be avoiding in future? | What were the surprises (unusual items) during the sprint? |
| Sprint 0 | We should carry on with picking the relevant topics to be spoken about during each meeting. | Staying on track during discussion and not diverging should be practiced given a couple of times that it has happened, however, even in this, the talking points of the meeting were still met, thus care needs to be taken. | No real surprises as of yet. |
| Sprint 1 | Discussions of where we are on the project continue to be effective as well as relevant issues being brought up and solved in a timely and effective manner. | Have all user stories ready. Sufficient customer contact. Handle more issues. There did exist task dependency. | User stories added after sprint start. Unexpected JIRA outage. Customer responsiveness. |

**9.5 SCRUM Meeting Notes Sample:**

**3/13/2019**

Finalize sections of code and work out any major bugs involved in adding or creating table entries.

Perform testing to ensure that both user and admin can see the correct pages and items relevant to their security status.

Merge pull requests from github and fix any merge issues.

re-perform testing upon completion of merges.

Reach out to the customer and ensure that the desired meeting time can be met.

**10. Conclusions:**

In conclusion, we feel that Sprint 1 went well to what was expected. We obviously have more learning to do and trying other options to solve what did not work well, otherwise we feel that customer expectations have been met as well as exceeding our own.

The theming behind Sprint 1 being environment setup and database configuration, our goal was to set up the base environment, ensure it was stable, and get the core website running. We needed to learn what issues may come from different systems, as well as ensure all basic code problems could be handled.

Thus far, no major requirement changes have been made. Our only change came in to play with desired user story implementation in that we had to go ahead and partially set up the user model in order to ensure that other functions were working.

The lessons learned from Sprint 1 have been simple but highly important. We obviously need to ensure that we are taking on enough work to not be sitting around and that we need to be sure to have all User Stories ready to go before starting the sprint. We also need to ensure that we maintain effective contact with the customer with ample time ahead of sprint close such that we are unsure as to if we will be able to meet for a demonstration in a timely manner. We have learned that our team does work well together and that we should keep up with our effective level of inter-communication. Our meeting scrum meeting agendas have been on point to us completing this sprint as well.

**11. Appendix:**

**11.1 SCRUM meeting:**

**1:**

Ruby or Javascript potential. We will use something everyone is comfortable with.

We should focus on the process and always note what works versus what didn’t. Obviously project completion with a working product is important, but the process is just as important to learn.

Ensure everyone understands the uploaded documents for reports.

Decide on the team name to be unique.

**2:**

Look at the uploaded projects so that we can begin determining what everyone wants to do. Be sure to pick a primary and backup projects that are liked.

Begin learning JIRA and play with the sandbox. Be sure to sign in to JIRA and comment on the checklist for participation. Go through the JIRA tutorial.

Consider what each role is going to require.

**3:**

Make note that Sprint 0 will be coming to an end soon, so be sure to have the JIRA tutorial completed, and play around in the sandbox.

Establish who is responsible for what parts.

Decide on a project by the end of the meeting.

**4:**

**3/11/2019**

This scrum meeting consisted mostly of code bug smashing

Set up remote computer such that users experiencing issues have a known working system to finalize tasks from.

Begin getting ready to merge code, and have a coding freeze at the end of Wednesday.

**5:**

**3/13/2019**

Finalize sections of code and work out any major bugs involved in adding or creating table entries.

Perform testing to ensure that both user and admin can see the correct pages and items relevant to their security status.

Merge pull requests from github and fix any merge issues.

re-perform testing upon completion of merges.

Reach out to the customer and ensure that the desired meeting time can be met.

**11.2 Customer Client Discussions:**

**Email 1:**

Mr. Rivera,

We will be the team working on your project "Luna's List". We have a few questions we would like to ask regarding the project.

* What is your preferred way of communicating (email, phone, skype, etc.)?
* Is the primary vision for the project mobile or web based? (As a group we have very limited mobile development experience so web-based would be preferred)
* Do you have any specific vision for the UI design of the site or will that be left to us to decide?
* Do you have a full problem statement already created or will that need to be generated

Thanks in advance!

**Email 2:**

Noah,

Thanks for reaching out. I am excited about working with you and the team!

My preferred way of communication is all 3.  First email for anything quick. Then skype in order to communicate effectively. Finally for anything urgent phone would be best.

We could absolutely do web based and move to mobile after the MVP.

I will leave it to the group on UI vision.  Some elements that are important will be ease of use and interesting as well as organized and visually appealing.

The full problem statement is here:

Dogs have been interwoven into our lives at a scale never seen before. They have moved outside of the perimeter of our homes and yards and into our roads, stores and entertainment areas.

More and more questions have flooded businesses, parks, and buildings on when, where and how can I bring my dog?

Lunas List will help answer these questions and more by providing a platform that provides communities with Dog friendly spaces.  It will also bring Dog Owners together in different ways.

Noah can I ask for a quick bio or resume of the group who is working on this project. And let me know if you have any questions. Also, is there milestone dates that are already established?

My number is 281.513.1354.

Thank you in advance,

Victor M. Rivera

**Email 3:**

Mr. Rivera,

Thanks for your response. As requested, here is a quick bio of the team members:

Thi Phan: Python, Java, Swift, Ruby on Rails

Wesley Jones: Python, Java, Ruby on Rails

Larz Leonard: Java, Python, Ruby on Rails

Vinh Tran: Java, C/C++, SQL

Noah Hanks: Python, C++, SQL, Ruby on Rails

Since most of us have experience with Ruby on Rails, it would probably be the best platform for us to base the project on. In regards to your question about milestones, the dates are not established yet but there will be a total of 4 sprints. If you have any more questions please feel free to ask.

Noah Hanks

**Email 4:**

Yes I can

On Feb 12, 2019, at 9:56 AM, Noah Wilson Hanks (nhanks) <nhanks@memphis.edu> wrote:

Mr. Rivera,

We would like to set up a skype meeting with you to discuss the project further. Will you be available to meet tomorrow (Feb 12) at 4:00 PM?

Noah Hanks

**Email 5:**

I can.

What will the agenda be?

Víctor

On Feb 18, 2019, at 4:11 PM, Noah Wilson Hanks (nhanks) <nhanks@memphis.edu> wrote:

Mr. Rivera,

I just wanted to confirm that you are still able to meet with us again on Wednesday (Feb 20) at 4:00PM.

Noah Hanks

**Email 6:**

Noah,

I will be en rite to the airport. Please call me on my cell phone 281.513.1354.

Victor

On Feb 18, 2019, at 4:21 PM, Noah Wilson Hanks (nhanks) <nhanks@memphis.edu> wrote:

The agenda for the meeting will be to discuss the current user stories that we already have for sprint one and to ensure that we are on the same page as far as priorities go.

**Email 7:**

Yes let’s reschedule for Friday. Send over what you have and I can also review them in the plane.

Thank you,

Victor

On Feb 20, 2019, at 4:58 PM, Noah Wilson Hanks (nhanks) <nhanks@memphis.edu> wrote:

Mr. Rivera,

It would be best if you were able to view the list of user stories as we discuss them. Would it be possible to reschedule the meeting for Friday (Feb 22) at 2:00PM?

Noah Hanks

**Email 8:**

Noah and Team,

I apologize for the confusion on my end. I thought it was at 4 pm.

We can do email. Or if you want to call me my number is 281.513.1354.

Victor

**Email 9:**

Mr. Rivera,

I never heard back with you on skype so I wanted to confirm over email. Will you be available to meet with us to discuss the results of sprint 1 this Monday (Mar 18) at 4:00PM?

Noah Hanks