**lorelines.com**

# **2. Project Management**

## 2.1 Change Management Procedure

In the event that there is a request to modify the project, the Change Administration Team (CAT) will be informed of the change via the Change Request Form found in Appendix A. The CAT currently includes all members of the project: Justin, Evan, Seth, and Isaac. Upon receiving a change request, the CAT will weigh the impact of the change on the project and either accept or decline the change. All requests will be compiled and saved in a text document on the project’s GitHub.

## 2.2 Software Delivery, Installation, and Acceptance Criteria

Lorelines will be completed and accessible via lorelines.com at the end of the 2020 Winter term. All documentation will be accessible from the project’s GitHub. Acceptance criteria will be based on the completeness and operability of the aforementioned items as determined by Todd Breedlove.

## 2.3 Documentation and Online Help

As stated in Section 2.2, documentation will be accessible through the project’s GitHub. Online help will be accessible through the completed Lorelines website.

## 2.4 Project Risks

Risks for this project include learning new technologies, balancing schedules for team meetings, and managing personal time to achieve project milestones.

## 2.5 Customer Responsibilities

The customer is expected to have access to the internet and an internet browser.

## 2.6 Status Reporting

Status reporting will be submitted weekly to Todd Breedlove. The report will include:

* Word completed during the past week
* Work to be completed during the next week
* Issues found during the previous week’s work
  + Potential way to fix the issues found

**4. Product Requirements**

* 1. **Functional**

1. Terminology
   1. *Custom entity:* a user-created or pre-built ‘class’ in which ideas, concepts, and user creations are stored and organized.
   2. *Entity instance:* a custom entity with its fields populated with unique characteristic and identifiers
   3. *Entity schema:* the layout of *fields* in a *custom entity.*
   4. *Entity name*: the collection name for all instance of this entity.
   5. *Entity color*: the color that instances of this entity will appear as on the *timeline*. The color will default to a random, bright color for users that do not wish to maintain a *timeline*.
   6. *Instance Name:* the keyword that this instance is referred to by the timeline.
   7. *Field:* Text-Field, Text-Box, Check-Box, Entity-List and/or Image
   8. *Node:* GUI objects that represent events and branching paths on the timeline
   9. *Link:* points on the left and right of *nodes* that represent ‘previous’ and ‘following’ events, visual *tethers* attach to these links.
   10. *Tether:* visual lines that connects nodes in a chronological sequence.
   11. *Instance link:* colored segments of text in *nodes* and *entity instances* that reference an *entity instance* and provide a link to view that instance.
   12. *World:* The complete collection of *custom entities*, *entity instances*, and *timeline*, essentially a save file.
2. Custom Entities
   1. Ideas are stored in instances of user-created classes and pre-built classes called *custom entities*
   2. Examples of pre-built *custom entities* include: Character, City and Event. These are entities that users are likely to use and demonstrate to users how to properly construct their own *custom entities*. Pre-built entities are an optional extra when creating a new *world*.
   3. *Custom Entity* Creator GUI
      1. Allows user to drag and drop *fields* onto an *entity schema*
      2. *Custom entities* can be saved, edited, deleted, and duplicated
      3. When creating a new *custom entity*, the user is required to define two variables: *entity name* and *entity color.* When creating an *entity instance* only the *instance name* must be defined.
      4. Users will be prompted by a warning when attempting to delete an entity that has one or more *entity instances*
   4. How *entity instances* are referenced on the timelineand in *entity instances*
      1. When it is determined that the user has typed an *instance name* into a *node* or an *instance field* the *instance name* will be colored the *entity color*, creating an *instance link*
      2. When the user clicks on an *instance link* the referenced instance is shown.
3. Timeline
   1. An endless grid on which *nodes* and *tethers* can be placed to construct a complete story.
   2. User can export timeline as an image file (possibly a premium feature).
4. Hierarchical Directory
   1. A visual way to view custom entities and entity instances without use of the timeline. Visually similar to windows file explorer.
5. User Accounts
   1. Creating user accounts
      1. User will be asked to provide the following information when creating their account:
         1. First name
         2. Last name
         3. Username
         4. Email address
         5. Birthday (For analytics)
         6. Country (For analytics)
         7. Password
            1. Encrypted password will be stored in database
6. Creating a new *world*
   1. Users can create as many *worlds* as they like (likely a premium feature)
7. Stretch Goals
   1. Map Maker
      1. User can export map as an image file (likely a premium feature).
   2. Color Themes
      1. Light theme
      2. Dark theme
   3. Ads and revenue streams
      1. Optional one-time purchase for an ad-free experience and premium features

**4.2 Performance**

**4.3 Reliability**

**4.4 Data Description**

**4.5 Security and Safety**

**4.6 Constraints**