CineList.com

Project Overview

This project aims to build an app that helps people streamline sharing of and minimize duplicate purchases of movies.

Have you ever purchased a movie online then discovered that a family member or close friend just brought the same thing? To reduce redundant purchases, it would be nice to have a simple app (web-based) that allows you, family members, and close friends with whom you often watch movies to be able to browse each other lists of titles. The app would allow you to add, update, and delete the titles in your own list, allow others to see your list, invite them to share their lists, and seamlessly browse or search all lists to which you have access. To maintain confidentiality and security, the app will authenticate users and allow them to manage access rights they have granted to others.

The envisioned app could be expanded or repurposed to handle other kinds of things besides movies. For example, it could be specialized to manage music, audiobooks, tools, camping equipment, board games, etc.

Team Organization

Our team roles will shift each phase as shown below. Due to the small size of our team, we expect that many of our responsibilities will overlap. We will all participate in regular development of the app. All code written during any given Phase must be reviewed and tested by the Project Manager. If the code was written by the Project Manager, one of the remaining two teammates must review and test the code before release. We expect any criticism to be given and received in a kind, objective, and respectful way.

Phase 1
Project Manager – Garrison Fehlauer
Scrum Master – Gary Johnson
Scrum Team Member – Hannah Davies

Phase 2
Project Manager – Hannah Davies
Scrum Master – Garrison Fehlauer
Scrum Team Member – Gary Johnson

Phase 3
Project Manager – Gary Johnson
Scrum Master – Hannah Davies
Scrum Team Member – Garrison Fehlauer

Phase 4
Project Manager – Garrison Fehlauer
Scrum Master – Gary Johnson
Scrum Team Member – Hannah Davies

Phase 5
Project Manager – Hannah Davies
Scrum Master – Garrison Fehlauer
Scrum Team Member – Gary Johnson

Software Development Process

The development will be broken up into five phases. Each phase will be a little like a Sprint in an Agile method and a little like an iteration in a Spiral process. Specifically, each phase will be like a Sprint, in that work to be done will be organized into small tasks, placed into a "backlog", and prioritized. Then, using on time-box scheduling, the team will decide which tasks the phase (Sprint) will address. The team will use a Scrum Board to keep track of tasks in the backlog, those that will be part of the current Sprint, those in progress, and those that are done.

Each phase will also be a little like an iteration in a Spiral process, in that each phase will include some risk analysis and that any development activity (requirements capture, analysis, design, implementation, etc.) can be done during any phase. Early phases will focus on understanding (requirements capture and analysis) and subsequent phases will focus on design and implementation. Each phase will include a retrospective.

Phase	Iteration
1.	Phase 1 - Requirements Capture
2.	Phase 2 - Analysis
3	Phase 3 - Architectural, UI, and DB Design
4	Phase 4 - Detailed Design, Implementation, and Unit Testing
5	Phase 5 - More Implementation and Testing

We will use Unified Modeling Language (UML) to document user goals, structural concepts, component interactions, and behaviors.

Communication policies, procedures, and tools

We will be using Slack as our primary means of communication. Slack provides an instantaneous text-based messaging service as well as various other tools to assist in communication.

We will conduct a daily scrum meeting every day at 3:20pm. In this meeting, we will discuss what has been accomplished since the prior meeting, and we will discuss what will be worked on next. There will be no formal communication procedures, aside from common sense. This means that we will limit communication to within standard business hours, with the exception of an emergency.

Risk Analysis

Person-related risk:

At any time, there is the risk that one or more team members will be incapable of working on the project for some amount of time. This may be the result of other schoolwork, a medical emergency, or some sort of family-related issue. These issues will be addressed as they occur at the discretion of the group as a whole. Extended periods of time without a team member will result in a deprecation of quality for the project.

Software-related risk:

Security risks: Because the software will store some personal information about the end-users, it is paramount that their information be secure from any sort of leak or malicious attack.

Legality risks: This software proposes to catalog and manage the borrowing of copyrighted material, and as such it may be at risk for a lawsuit from the copyright holders in question.

Lack or loss of user interest: This risk could be a result of any of the above risks. If a user feels their information is not secure or that the service may be risky, they will likely lose interest in the application. In addition, a lack of interest could simply be a result of lackluster software to begin with.

All of the above software-related risks could result in the immediate failure of the project. As such, steps must be taken to limit the likelihood of each occurring.

Configuration Management

See the README.md in the Git repository.