Dan’s Frappuccino Paradise

# Project Overview

This project aims to build a system for running a Frappuccino Café.

There are three roles a person can have while using the system: customer, barista, or manager. Customers can place orders and manage account balances. Baristas can log hours, place orders for customers, and view and edit pending order progress. Managers can pay, hire and fire employees, buy inventory, edit the menu (including prices) and manage the company’s finances. Baristas can do everything that customers can do, and managers can do everything that baristas can do.

# Team Organization

Project Manager: Gabe Tonks (for this phase - subject to change)

Developers: Justin Roylance, Spencer Hall, Logan Nielsen

# 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, Architectural, UI, and DB Design |
| 3 | Phase 3 - Implementation, and Unit Testing |
| 4 | Phase 4 - 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

All communication shall take place over discord.

This document and any other Word documents shall be shared and edited on Google Drive, then uploaded to GitHub upon completion.

All code shall be pushed to the shared GitHub repository.

UML diagrams shall be created using draw.io

# Risk Analysis

* Login
  + Likelihood: Low
  + Severity: Medium
  + Consequences: Unfavorable customer experience accessing account information, making and saving purchases.
  + Workaround: Design manual method of placing orders without providing account information.
* UI
  + Likelihood: Low
  + Severity : High
  + Consequences: Poor user experience with the application. Application likely to be unsuccessful as a whole.
  + Work-Around: None
* Payment System
  + Likelihood: Low
  + Severity: Very High
  + Consequences: Difficulty paying employees and buying inventory. Difficulty receiving payment for coffee.
  + Work-Around: None. Store loses value if unable to make transactions.
* Hosting
  + Likelihood: Low
  + Severity: Medium
  + Consequences: Unable to input or receive information from the application
  + Work-Around: Use an established, tested hosting service.

# Configuration Management

See the README.md in the Git repository.