Schedule is subject to change

SWE 3313 Group 2 Schedule

Intro

This is our group's schedule, which will clearly convey what our group will have done by certain deadlines, and how long it takes to complete specific portions of the project. The schedule document contains a Work Breakdown Structure (WBS), milestones for deliverables, and a Gantt chart. The WBS will help identify and organize the tasks that need to be done to complete the project. Milestones will help the project remain on track for completion before the client's proposed due date. The Gantt chart will help visualize the projects timeline, so the team is on the same page.

Work Breakdown Structure

Deliverables:

1. Project Plan Part 1
    - Scope
        - Critical thinking
        - Planning
        - Collaboration
        - Writing
    - Organization
        - Writing
2. Project Plan Part 2
    - Schedule
        - Critical thinking
        - Collaboration
        - Writing
    - Test Plan
        - Critical thinking
        - Collaboration
        - Writing

Paper prototype and requirements can be worked on at the same time

---

3. Paper Prototype
    - Creating an accurate model using MS paint or other art software.
    - Collaboration

4. Requirements part 1, 2, and 3
    - Planning
    - Collaboration
    - Writing
    - Creating Diagrams

---

5. Concept Design
    - Planning
    - Collaboration
    - Writing

6. Technical Design
    - Planning
    - Collaboration
    - Writing
    - Creating Diagrams
    - Programming

Time and Effort Required:

1. Project Plan Part 1 **Est Time: 40min-1hr|20min**
    - Write out their plan on a document that includes the scope, team organization, and data management. **Est Time: 30min-1hr**
    - Members need to figure out the scope of their project and organize all of the team's roles. **Est Time: 10min-20min**
    - Members need to upload a resume.

2. Project Plan Part 2 **Est Time: 1hr|40min- 2hr|50min**
    - Collaborate on a schedule that all members can agree on. Est Time: 10min-20min
    - Create a Gantt Chart **Est Time: 30min-1hr**
    - Write a document that contains the Work Breakdown Structure, milestones, and Gantt chart. **Est Time: 1hr-1hr|30min**

3. Paper Prototype and Requirements **Est Time: 2hr|55min-4hr|30min**
    - Creating a rough visual prototype of our project. **Est Time: 15min-30min**
    - Written Requirements
        - Creating a document that will help track written portions of our project and describing each. **Est Time: 20min-30min**
    - Software Requirements
        - Creating a document that will contain the paper prototype and describe our software requirements. **Est Time: 20min-30min**
    - Creating Use-Case diagrams and flow of events. **Est Time: 30min-1hr**
    - Creating ER diagrams. **Est Time: 20min-30min**
    - Creating State Transition diagrams. **Est Time: 20min-30min**
    - Creating a Class Specification Document. **Est Time: 30min-1hr**

4. Concept Design **Est Time: 1hr|10min-2hr**
    - Creating screen mockups. **Est Time: 20min-30min**
    - Creating a High-Level UML Class Diagram. **Est Time: 20min-30min**
    - Designing a conceptual system. **Est Time: 30min-1hr**

5. Technical Design **Est Time: 20hr-39hr**
    - Creating and designing the Low-Level Class Diagrams. **Est Time: 1hr-1hr|30min**
    - Programming the software. **Est Time: 15hr-30hr**
    - Testing the software. **Est Time: 3hr-6hr**
    - Creating the Project Notebook. **Est Time: 1hr-1hr|30min**

Group 2 (Allen, Exley, Farah, Goff, Kershaw, Lysaght)
March 2, 2020
Project Plan, Part II

Milestones

Project Plan Part 1 – *Includes project scope, team organization, team resumes, and data management plan*

Project Plan Part 2 – *This document. Includes work breakdown structure, list of milestones, Gantt chart, and system technical description*

Requirements Part 1 – *Includes written requirements and software mockup*

Requirements Part 2 – *Includes use case diagrams, use case flow of events document, and class diagrams*

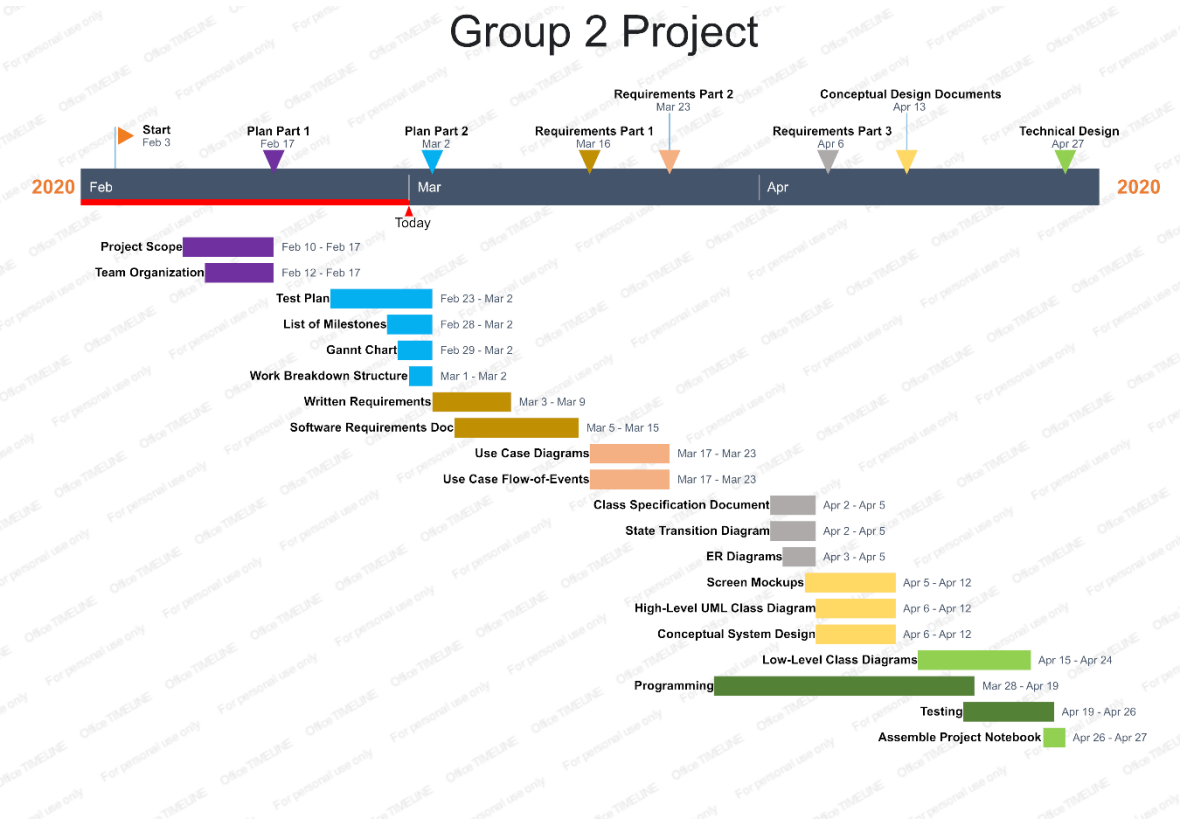
Requirements Part 3 – *Includes entity relationship diagrams, class specification document, and state transition diagram*

Concept Design Documents – *Includes screen layouts with requirement traceability*

Technical Design – *Includes class diagrams*

Group 2 (Allen, Exley, Farah, Goff, Kershaw, Lysaght)
March 2, 2020
Project Plan, Part II

Gantt Chart



Testing Plan

This test plan covers the different aspects of a game of baker's dozen solitaire, how those software pieces interact, and how each must be tested individually and as a whole to meet the functional requirements of our client Kennesaw State University via Dr. Chao Mei.

Test Policy…… 1

Test Strategy and Project Test Plan…… 2

Components Test Plan…… 2

Test Policy

The overall test policy for the project should focus on the design requirements laid out in the SWE_3313_Term_Project_Spring2020 document along with additional requirements laid out by the development team. These are as follows

Backend

- A standard deck of 52 cards (no Jokers)
- Randomized into 13 piles (tableaus)
- Four piles must have kings as the bottom item in the stack
- Four foundations that each accept cards of one suit and in order from lowest to highest
- Ability to reset game to new random start state
    - Empty tableaus and foundations, then refill randomly
- Only the top card of each tableau can be moved
- Cards can only be placed on cards that are one value higher
- Cards may be moved from the foundations back into the tableaus.
- Win condition of all foundations filled and the tableaus being empty
- Timer that resets on game start/reset
- Move counter that resets on game start/reset and isn't affected by the undo move feature.

Frontend

- Desktop game application
- GUI with graphics for the cards, tableaus, foundations, reset button
- Dynamic timer graphic
- Dynamic move counter graphic

Each of these features will be tested for functionality to the development teams, client's, and end user's desires and requirements. Test results that do not meet requirements will be fixed by the development team and results that call in to question the aspects of requirements will be relayed to the client for review and any requirement changes will be passed back to the development team.

Test Strategy and Project Test Plan

Testing will occur at each milestone. All features completed upon the reaching of the milestone will be tested to show they are functional and designed with both the client's intent and efficiency in mind. Tests scripts will identify and indicate fixes for any defects and be able to retest components or systems post fixes.

All features and aspects of the game will be examined and determined to be in either the early stages of implementation, a working and testable state, or the final product integrated with the rest of the project's requirements. In testing, project members will confer with the other members of the team in order to reach a consensus on a feature's progress. Based on that progress the team will decide on what actions should be taken, whether it's more testing, a change in design, or altering the method the feature is using to implement. During final system testing the product will also be tested for user acceptance.

Components Test Plan

The Deck of cards

- Test should increment through the data structure the cards are stored in and display the cards value, suit, and color to confirm the existence of all 52 unique cards
  - The four kings can be removed and checked in a separate data structure making a check of 4 cards and a check of 48

Randomizing the Deck

- The cards data structure should be randomized then incremented through and display the cards value, suit, and color to confirm all cards are present and their order. Then the data structure should be randomized and analyzed again. Repeat this process a number of times and record results
  - The four kings if separate should be tested in the same way

Preparing the Deck for the Game

- Test the ability to empty tableaus
  - This test script should empty the tableau data structures and increment through each of them to confirm they are empty
- Test the ability to randomly insert the four kings into the 13 tableau data structures.
  - Increment through each of the tableaus to confirm the placement of the kings, then reset and repeat until satisfied each of the kings is being randomly and uniquely placed into one of the 13 tableaus
- Test the ability to randomly insert the remaining cards into the tableaus
  - This should be done after testing the four kings and each instance of this test should start from an end of case of the king test. Confirm the cards are being put into the tableaus and accounts for the already placed kings. The test should

increment through all 13 tableaus showing that each has four cards and four have kings as their lowest item

Card Movement

- Cards should be put into the tableau in a not random order to test movement interaction
  - cards can be placed on a card of any suit that is one value higher
  - nothing can be placed into an empty tableau
- Scripts must be written that call individual tableaus and attempt to move card objects to other tableaus
- The movement of cards to the foundations must also be tested
  - Only one suit per foundation, stacks from lowest to highest card value, with ace acting as lowest.
- Movement from foundation piles back out to the tableaus is permitted.
- If a user-friendly interface has been built client/user testing can begin to be done here. This could help cover any input conditions that test scripts fail to.

Rule Implementation

- The goal is to move all the cards to the foundation piles.
- Baker's Dozen uses one deck, and there are no redeals.
- An empty spot may not be filled.
- The top card of a tableau or foundation may be moved.
- The foundation piles must be built up by rank and by suit, starting from Ace and ending with King.
- The tableau stacks are built down only by rank (number), meaning the cards can only be stacked on a card that is one card higher.
- All kings must be moved to the bottom of their respective tableau before user play begins.
- Conditions and tests will be put in place to guarantee users follow the rules.

Additional Components

- Extra features will be explored depending on the state of testing and progression of the core components. Testing for the additional components will follow the same format as core component testing.
- The extra features would improve the quality of life of the user. Features such as undo move, a timer, or a move counter.

Group 2 (Allen, Exley, Farah, Goff, Kershaw, Lysaght)
March 2, 2020
Project Plan, Part II

Technical Description

This software is a desktop game application created by students. The application allows any user to play a game of Baker's Dozen solitaire. It utilizes a GUI to display the game that will be played by a user and is created for ease of use by users through the UI of the software.

The software accepts various inputs such as clicking and dragging to allow interaction between the user and the interface of the game. All criteria for winning the game of solitaire must be met, and only then will the player be rewarded with a victory screen. The game runs on any Windows PC as a standalone application and will look identical on any display.