Name_Michael Moldenhauer___ Mark ____________________/50

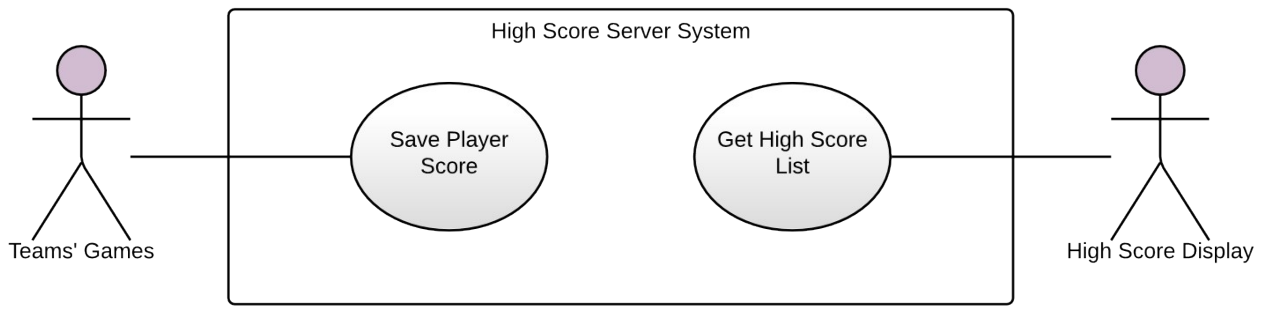
[**Instructions**: Remove everything that is not a heading below and fill in with your own diagrams, etc.]

1. Brief introduction __/3

This document sets out the usage and design requirements for a high-score server to be used at the gaming session at the end of this CS 383 class. The high-score server is a program designed to tabulate and maintain a constantly-updating list of the highest scores earned by players in the games played during the session at any given moment in time, which will be displayed on the presentation screen along with the games themselves. Such a server requires input from client games in the form of scores, players' names, and the name of the game in question, and it will display a table of scores organized by which game they come from, in descending order of size, perhaps snazzed up with graphics to add eye appeal.

2. Use case diagram with scenario __14

Use Case Diagrams



Scenarios

Name: Save Player Score
Summary: A game stores a score for one of its players.
Actors: Teams' Games.
Preconditions: Server has been initialized.
Basic sequence:

Step 1: Score receipt client accepts game name, user initials, and score
Step 2: Score receipt client passes this data on to the server.
Step 3: The server stores these items in its internal database, overwriting other scores for the same name and initials, if necessary.

Exceptions:

- **Step 1:** There was no game and player initials specified.
- **Step 2:** There was no connection to the server or no viable server running.

Post conditions: The given data has been entered into the database successfully.
Priority: 1
ID: HSS01

Name: Get High Score List
Summary: Request a list of the highest scores (say the top 5) scored in the database.
Actors: High Score Display
Preconditions: Server has been initialized.
Basic sequence:

- **Step 1:** Display client polls server for high score list.
- **Step 2:** High score server sends list to display client.
- **Step 3:** Display client presents it to the screen.

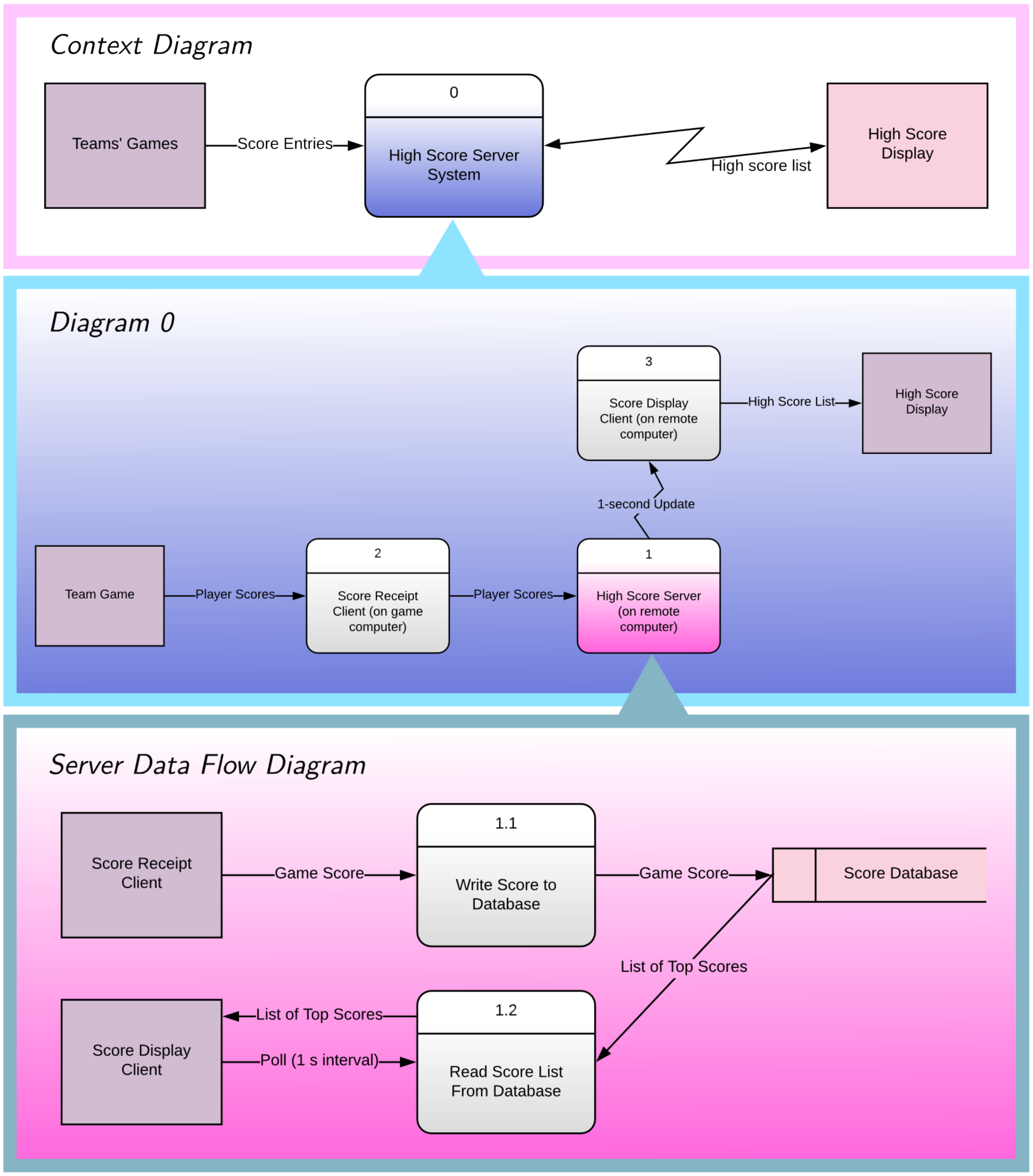
Exceptions:

- **Step 1:** The server was not running or could not be connected to.

Post conditions: The database is unchanged.
Priority: 1
ID: HSS02
*The priorities are 1 = must have, 2 = essential, 3 = nice to have.

3. Data Flow diagram(s) from Level 0 to process description for your feature ______14

Data Flow Diagrams



Process Descriptions

```
Write Score to Database:
    Receive score from client
    Receive game name from client
    Receive initials from client
    IF game name and initials are not empty strings
        Write the score, game, and initials to the database file
        in its proper format
    ELSE
        Exceptional case – report error
    END IF

Read Score List From Database (Polling):
    Receive destination address in poll
    Validate destination address
    Receive number of scores, N, to get
    IF destination address is invalid
        Exceptional case – report error
    END IF
    Read high score database file into memory
    Sort scores from highest to lowest
    Send N highest-sorted scores to destination
```

4. Acceptance Tests ________9

Server database fill test

Insert 100 random entries into the high score database through the server interface with random, non-empty game names and initials and random score numbers from 0 to the maximum

The output file should contain all the stored information in the proper format and no errors should have been raised

High score retrieval

First, try to poll server with empty database. After the first test, poll the server once for the high score information and check if it matches the input that was sent.

Live polling

Try the live-update high-score display with a continuous feed of random dummy input.

Error (exceptional) cases

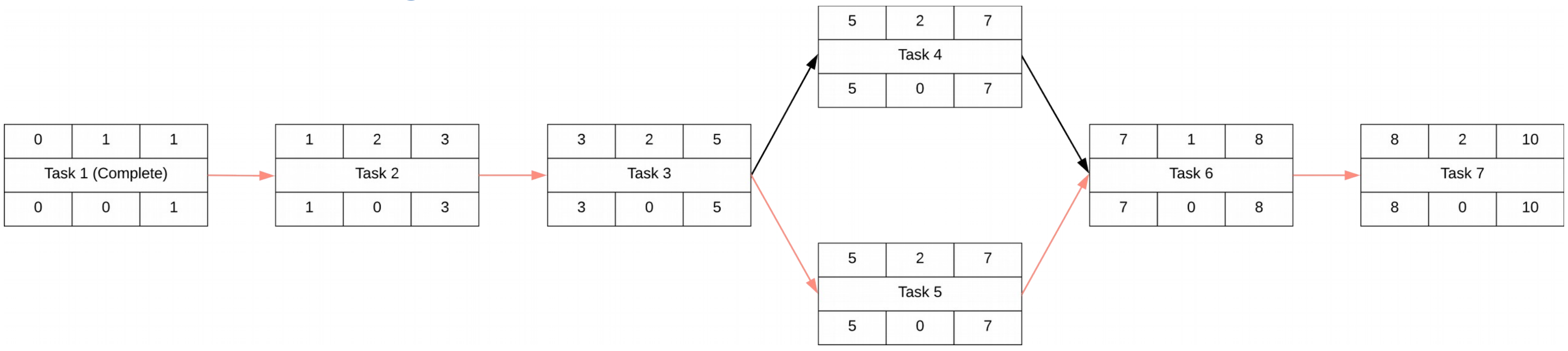
Try to send one of each: an entry with empty game name but non-empty initials, empty initials but non-empty game name, and both game name and initials empty, to see if an exception (error) raises properly. Also, try clients with server disconnected / not functioning or if/when server cuts out in the middle of a run to test for graceful failure

5. Timeline ________/10

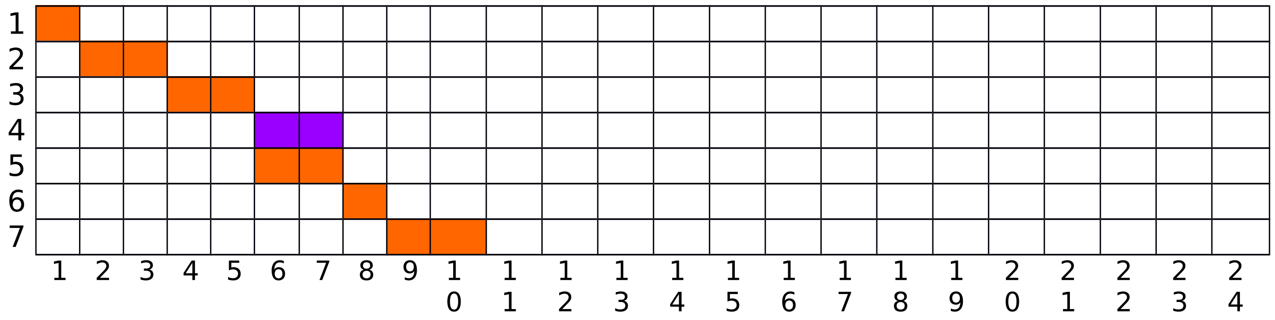
Work items

Task	Duration (PWks)	Predecessor Task(s)
1. Presentation of SA	1 (already done)	-
2. Initial Prototype	2	1
3. Prospection for and Study of Reusable Component	2	2
4. Construction of Clients	2	3
5. Construction of Server	2	3
6. Testing and Debugging	1	4, 5
7. Game Integration	2	6

Pert diagram



Gantt timeline



1
2
3
4
5
6
7
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24