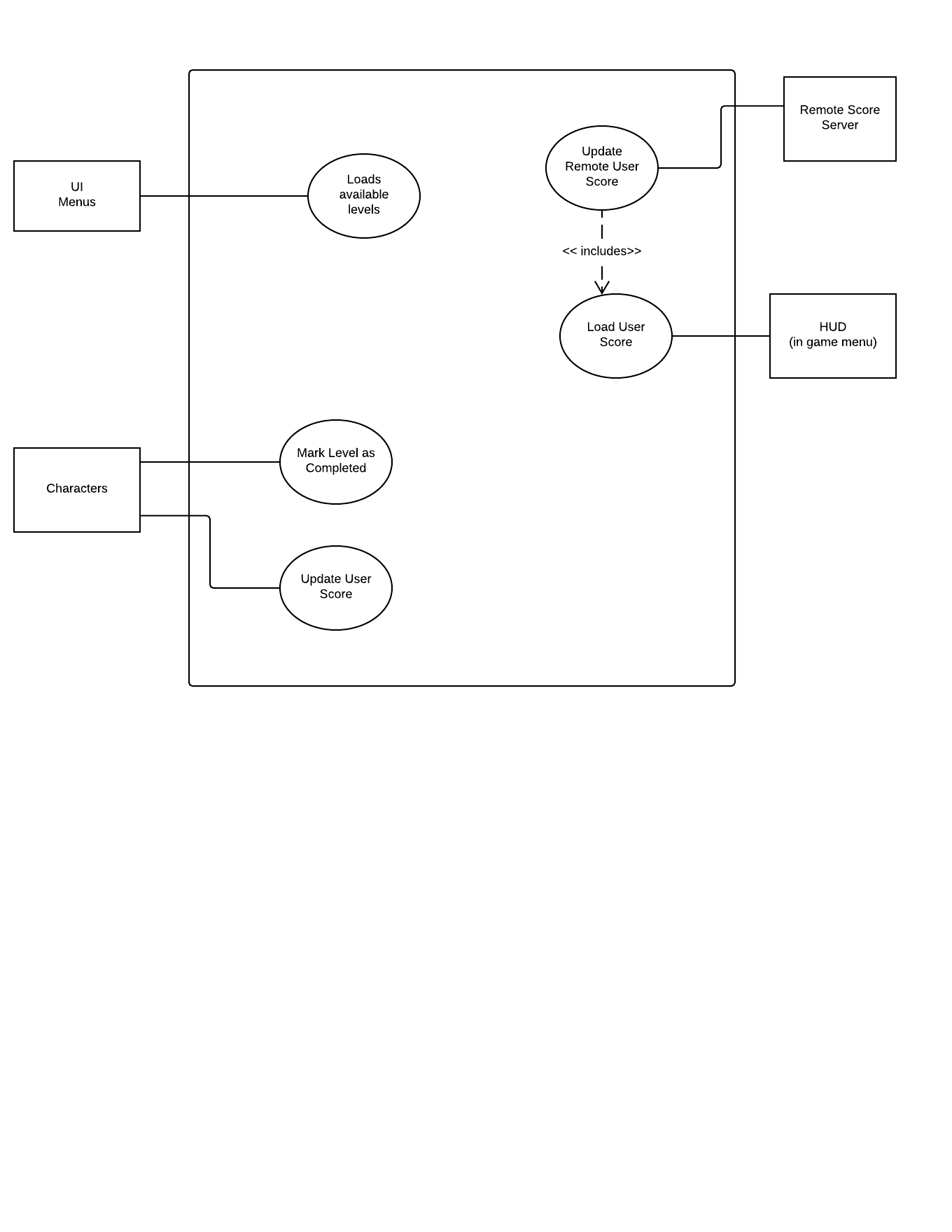
Name:\_Zane Durkin\_\_ Mark \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/50

## Brief introduction \_\_/3

My feature is the saving of user profile data such as which levels the player has unlocked, what the player’s highest score is, as well as sending the high scores to the remote score server.

## Use case diagram with scenario \_\_/14

### Use Case Diagrams



### Scenarios

**Name:** Load Available levels

**Summary:** The levels that a user has unlocked are returned.

**Actors:** User Interface.

**Preconditions:** Main Menu must be showing.

**Basic sequence:**

**Step 1:** User profile is created from Main Menu and saved as a profile.

**Step 2:** User profile is selected from Main Menu.

**Step 3:** Load what levels the user has unlocked to.

**Exceptions:**

**Step 1:** Available levels are requested before profile selected, return only first level.

**Post conditions:** Available levels are displayed.

**Priority:** 1\*

**ID:** C01

**Name:** Update User Score

**Summary:** The user’s score is updated.

**Actors:** Character, HUD.

**Preconditions:** Game must have been initialized.

**Basic sequence:**

**Step 1:** Character Tells how many points should be added / removed.

**Step 2:** Total points are written to disk.

**Step 3:** Total points are sent to HUD.

**Exceptions:**

**Step 1:** Character removes more points than are available, return 0.

**Post conditions:** Data is written to disk.

**Priority:** 2\*

**ID:** C02

**Name:** Load User Score

**Summary:** return User Score.

**Actors:** HUD.

**Preconditions:** Game must have been initialized.

**Basic sequence:**

**Step 1:** Total points are returned.

**Exceptions:**

**Step 1:** Character removes more points than are available, return 0.

**Post conditions:** Data is sent to HUD.

**Priority:** 2\*

**ID:** C03

**Name:** Level marked as Completed

**Summary:** The users completes a level and can now access the next level.

**Actors:** Character.

**Preconditions:** previous level must have been completed.

**Basic sequence:**

**Step 1:** Completed levels returned by Character.

**Exceptions:**

**Step 1:** Level 1 is not marked as available, return level 1 is available.

**Post conditions:** Data is written to disk.

**Priority:** 1\*

**ID:** C04

**Name:** Update Remote Score server

**Summary:** The user’s score is sent to the remote score server.

**Actors:** Remote Score Server.

**Preconditions:** User profile must have been created.

**Basic sequence:**

**Step 1:** Load User score.

**Step 2:** Send updated scores to Score server.

**Exceptions:**

**Step 1:** Score is requested even if player has never played, return 0.

**Post conditions:** Send updated scores to Score Server.

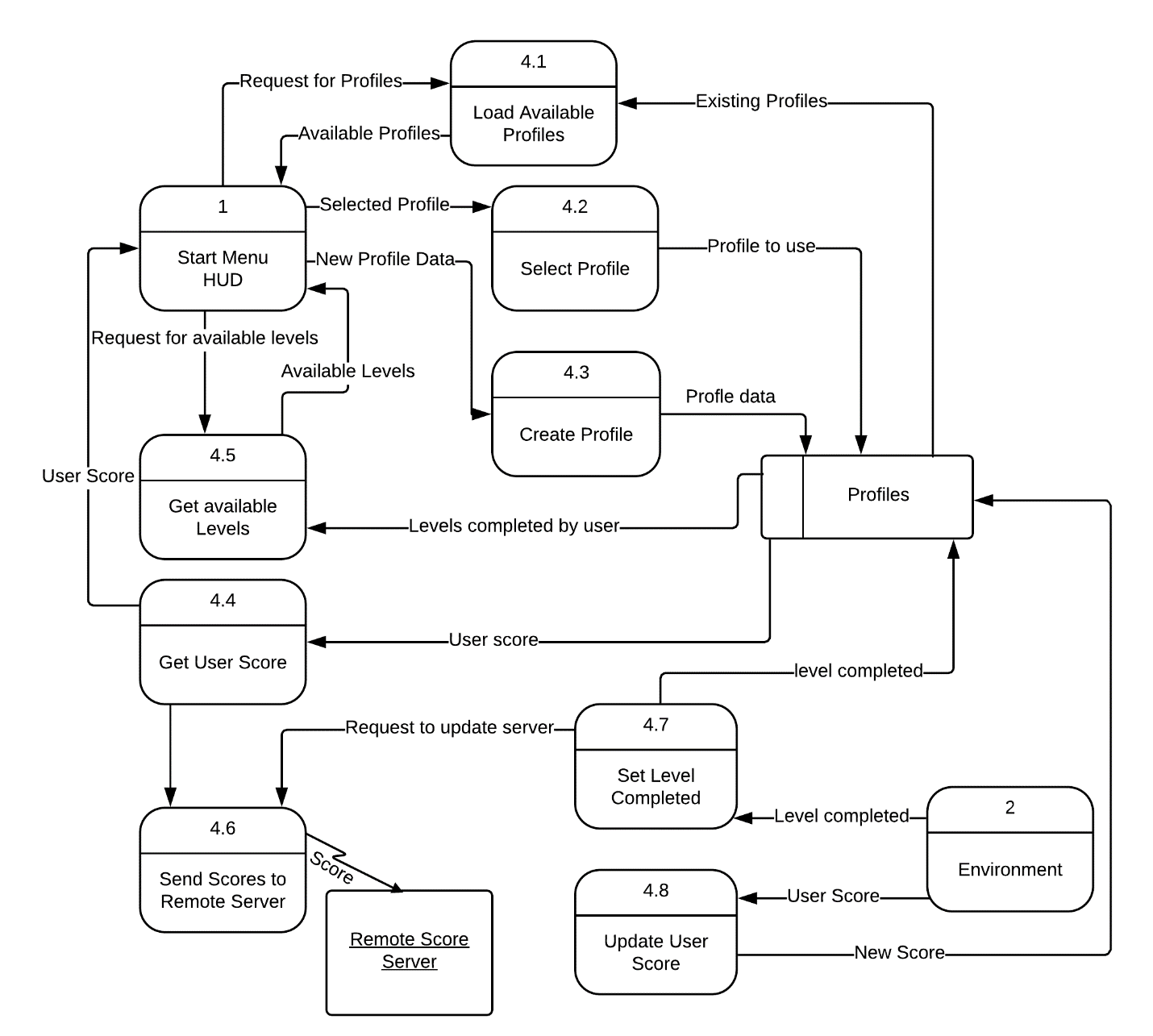
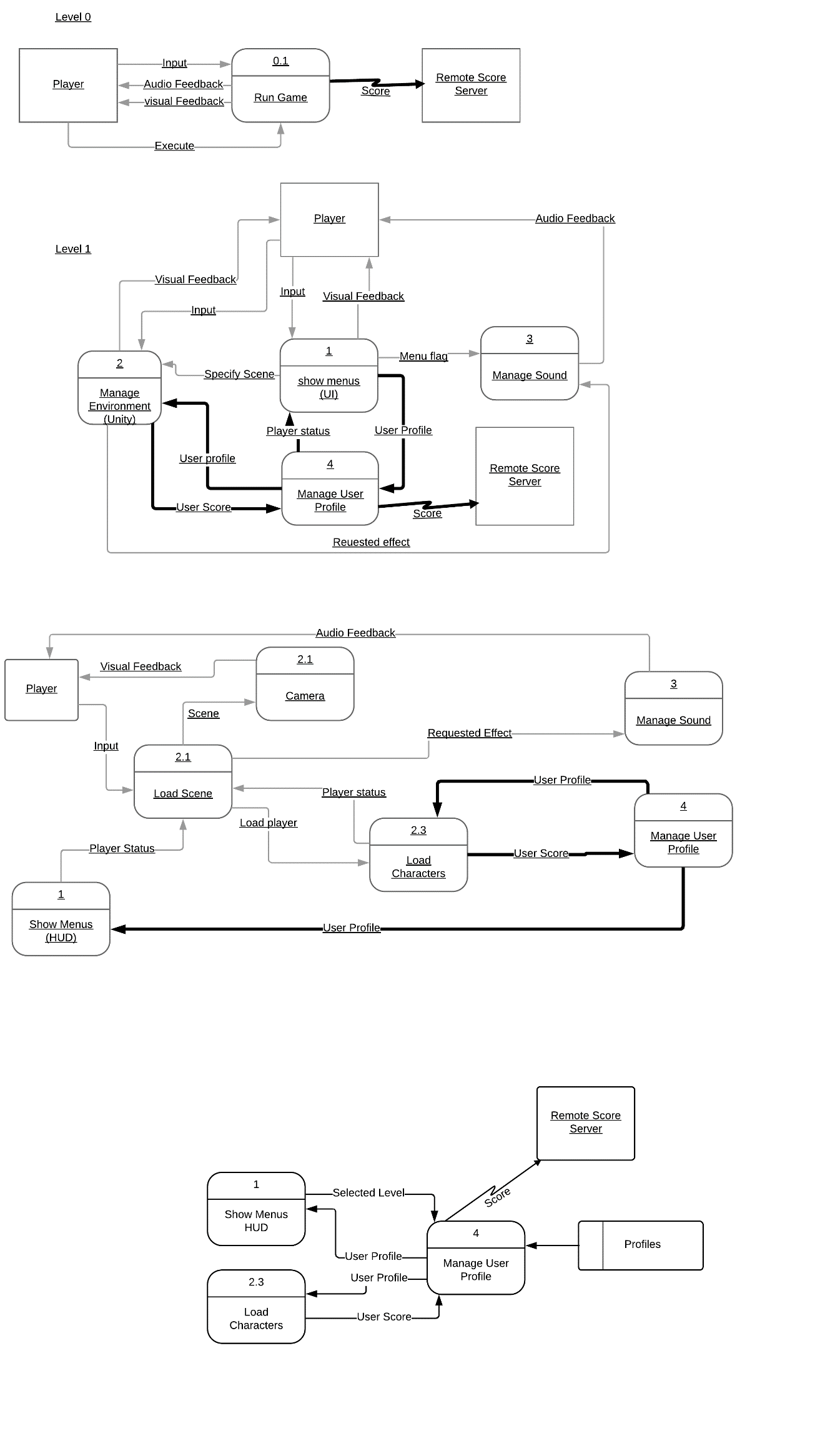
**Priority:** 3\*

**ID:** C05

\*The priorities are 1 = must have, 2 = essential, 3 = nice to have.

## Data Flow diagram(s) from Level 0D to process description for your feature \_\_\_\_\_\_/14

### Data Flow Diagrams



### Process Descriptions

Select User Profile\*:

CREATE User Profile IF Doesn’t exist.

Select User Profile IF not selected.

Return Available Levels IF User profile selected.

Return User score IF User Profile selected.

Update User Score\*:

Receive Score to add to User’s total.   
Save updated score to User’s profile.

Mark level as completed\*:

Receive which level the user has completed.   
Save level as completed on User’s profile.

Send User’s total score to remote score server.

## Acceptance Tests \_\_\_\_\_\_\_\_/9

**Create User Profile**

Create 3 User Profiles

* Each profile has level 1 available
* Each Profile has a score of 0 to start
* Each Profile has no tokens collected to start

Add points to a New User Profile

* Verify user starts with 0 points
* Add 10 points to user
* Verify user has 10 total points
* Add 20 points to user
* Verify user has 30 total points

Add completed level to New User Profile

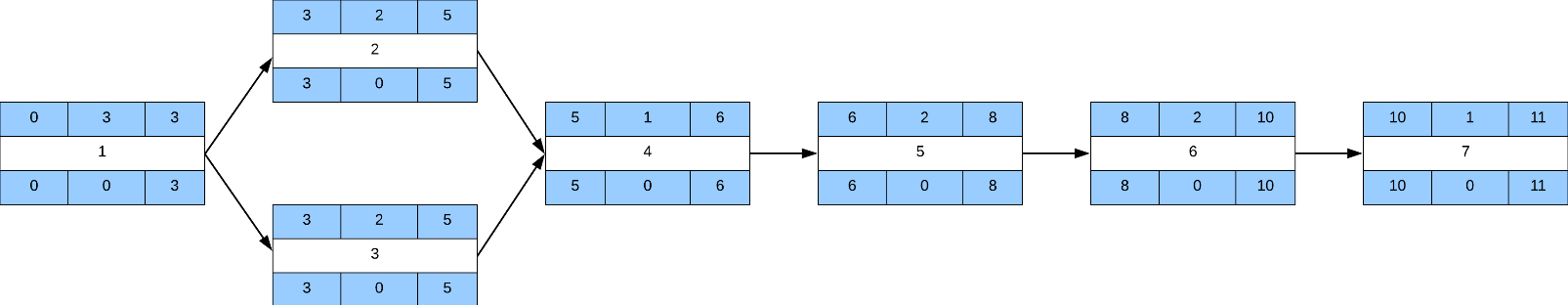
* Verify user starts with only level 1 available
* Mark level 1 as completed
* Verify User can now access Level 2
* Verify User can only access Level 1 and 2

## Timeline \_\_\_\_\_\_\_\_\_/10

### Work items

|  |  |  |
| --- | --- | --- |
| Task | Duration (PWks) | Predecessor Task(s) |
| 1. Requirements Collection | 3 | - |
| 2. Function Definitions | 2 | 1 |
| 3. Database Construction | 2 | 1 |
| 4. User Documentation | 1 | 2, 3 |
| 5. Programming | 2 | 4 |
| 6. Testing | 2 | 5 |
| 7. Installation | 1 | 6 |

### Pert diagram



### Gantt timeline

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 |  |  |  | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 |  |  |  | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  | 3 |  |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  | 4 |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 5 |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 6 |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |