**Bon Applícation Use Cases**

***Use Case 1: Catering Manager - Handle Meal Swipes***

The Catering Manager receives a list of students’ names and their LU IDs, denoting which students will be attending a catered event. The Catering Manager will enter the student information into the system and the number of meal swipes to deduct from each student. The System will then make those deductions from the students’ meal plans.

***Use Case 2: All Staff - Clock In and Out***

At the beginning of their assigned shift, the Employee logs in at a terminal with their identification, and indicates that they are beginning their shift. The SuD records this time as the start time of their shift. The Employee also logs the start and end time of their breaks, if any, in the same fashion. At the end of their shift, the Employee logs into the terminal and indicates that they are ending their shift. The SuD records the end time, and the number of hours worked.

***Use Case 3: Student Employee Manager - Schedule Shifts***

*Main Success Scenario:*

At the beginning of the year, Student Employees fill out a schedule form with the SuD indicating the work shifts they are available to take. If everything is hunky-dory, the Student Employee Manager lets the SuD create the schedule. The SuD distributes shifts to Student Employees in an optimal way. The Student Employee Manager then checks the output schedule to ensure hunky-doriness was preserved. They then let the SuD distribute the schedule to the Student Employees.

*Alternate Scenarios:*

*Before schedule creation:*

If no Student Employee takes a shift, the Student Employee Manager may communicate to the Student Employees that the shift is still open and ask whether anyone can take it, or may instead decide to leave it open as a sub position that a Student Employee might claim in the future.

If some Student Employee’s weekly availability is less than 10 hours (the required number of weekly hours for student employment), the System notifies the Student Employee Manager, who must negotiate with the Student Employee before the SuD distributes shifts.

*After schedule creation:*

If the System was unable to give any shifts to a Student Employee, the Student Employee Manager must renegotiate the Student Employee’s availability and let the SuD redistribute shifts.

If some shifts are not to the Student Employee Manager’s liking for whatever reason, the Student Employee Manager may manually edit shift assignments. The SuD will alert the Student Employee Manager if some Student Employee was not assigned at least 10 hours of work per week.

|  |  |  |
| --- | --- | --- |
| **USE CASE 3** | **Reserve Kitchen Time** | |
| **Goal in Context** | The Catering Manager, in preparation for an upcoming catering event, needs to reserve the Andrew Commons kitchen space for cooking. | |
| **Scope** | Bon Application | |
| **Level** | Primary Task | |
| **Preconditions** | The Catering Manager knows the number of hours of kitchen time needed to prepare the catering food. | |
| **Success End Condition** | The kitchen is reserved for the needed number of hours before the event. | |
| **Failed End Condition** | The kitchen was unable to be reserved. | |
| **Primary Actors** | Catering Manager | |
| **Trigger** | Catering event is scheduled. Catering Manager indicates to the SuD (via GUI button, punch cards, telepathy, etc.) that they want to reserve the kitchen. | |
| **DESCRIPTION** | **Step** | **Action** |
|  | 1 | Catering Manager approves catering request. |
|  | 2 | Catering Manager checks inventory (see Use Case CHECK INVENTORY) to see if the necessary supplies are on-hand. |
|  | 3 | SuD calculates how many hours are needed to prepare the food given the menu. |
|  | 4 | Catering Manager enters the date of the catering event, and the number of hours they need to reserve. |
|  | 5 | The SuD reserves the kitchen at that time for the number of hours needed. |
|  | 6 | The SuD updates the kitchen calendar |
|  | 7 | The SuD notifies all relevant employees of the schedule change |
| **EXTENSIONS** | **Step** | **Branching Action** |
|  | 2a | Not all necessary supplies are available :  2a1. Catering Manager orders supplies (See Use Case ORDER SUPPLIES) |
|  | 5a | SuD cannot reserve the kitchen:  5a1. SuD indicates to Catering Manager that the time can’t be reserved.  5a2. SuD displays calendar of reservations to Catering Manager.  5a3. Catering Manager negotiates new schedule with the Kitchen Manager. |

**Use Case: 4 Track Inventory**

--------------------------------------------------

CHARACTERISTIC INFORMATION

Goal in Context: Update the database to accurately reflect changes in inventory contents.

Scope: Bon Application

Level: Primary function

Preconditions: Kitchen Manager has a dif list of inventory from this day.

Success End Condition: The inventory database accurately reflects the contents of the inventory.

Failed End Condition: The inventory database is inaccurate.

Primary Actor: Kitchen Manager (duty includes keeping inventory DB up to date.)

Trigger: Inventory must be updated at the end of each meal.

----------------------------------------

MAIN SUCCESS SCENARIO

1. Kitchen Manager prepares list of added and removed items.

2. Kitchen Manager logs into terminal and brings up inventory dialog.

3. Kitchen Manager enters the list.

4. SuD updates the inventory.

----------------------

EXTENSIONS

2a. Wrong/forgotten username and password:

2a1. Kitchen Manager inputs pre-designated e-mail

2a2. Actual username and password sent via e-mail

4a. Trying to remove more items than accounted for.

4a1. Kitchen Manager must manually count the inventory and enter the true number.

4a2. Kitchen Manager then chastises whoever is responsible for the inconsistency (or keeps the problem quietly under wraps if they are the one responsible).

4b. Trying to add back more/less items than were originally there. (items flagged for borrowing)

4b1. The Kitchen Manager must manually resolve whichever scenario (see sub-variations) is extant.

4b2. The Kitchen Manager manually enters the true count of the item.

4b3. Kitchen Manager then chastises whoever is responsible for the inconsistency (or keeps the problem quietly under wraps if they are the one responsible).

4c. Caveat: The System will not detect if the updated sum is greater than true number, but can detect if the number is less IF the resulting sum is less than 0, or if the number of items returned doesn’t match the number of items borrowed.

4d. If the SuD detects that some supplies are lower than the needed amount (if one is enforced), it notifies the Kitchen Manager.

--------------------

SUB-VARIATIONS

4a1. Two possibilities of what went wrong.

1. The number the Kitchen Manager tried to update with was wrong.
2. The inventory’s original number was wrong.

4b1. Several possibilities of what went wrong:

1. Wrong number counted at moment of borrowing.
2. Less items returned than borrowed.
3. More items returned than borrowed (sneaky alien kitchenware).
4. Incorrect count of returned items.

----------------------

RELATED INFORMATION

Priority: 8/10

Frequency: after-each-meal/day

Channel to primary actor: Computer terminal

Secondary Actors: Inventory DB

**Plan of Attack:**

**Intro**

**Domain Description: Sanfer**

**Actor-Goal List: Jordan**

**Use-Case Diagram: Liza**

**Use Cases**

**Brief 1: Sanfer**

**Brief use case 2: Liza**

**Casual use case: Jordan**

**Table use case:**

1. **Everything up to steps: Sanfer**
2. **Steps: Liza**
3. **Extensions: Jordan**

**Numbered use case:**

1. **Characteristic Information: Sanfer**
2. **Main Success Scenario / Extensions (and sub-variations) 1 through 4a: Liza**
3. **Remainder of Extensions (and sub-variations): Jordan**
4. **Related Information: Sanfer**