Version History Table

Version	Date	Description
v1.0	04.11.2023	-
v1.1	02.12.2023	Special Requirements are updated according to review feedback. System interaction is added in the main scenario. Technology and Data Variations List is removed, as it is not required in UP format

Use Case UC3: Update Reservation (Diner)

Scope: ReserveWell Application

Level: user goal

Primary Actor: Diner

Stakeholders and Interests:

- Restaurant Manager: Wants accurate and efficient reservation management and optimized table allocation. He/ She needs the ability to check real-time availability and streamlined reservation management processes.
- Waitstaff: Wants to provide efficient and high-quality service to customers. They need the ability to arrange tables' physical availability according to reservations updates.
- Diner: Wants flexibility to modify a reservation based on changing plans or preferences.
- Development Team: Wants to accurately account for reservation changes to the restaurant using correct format and protocol. Need to ensure system's stability, scalability, security, and adherence to best practices.

Preconditions:

- The diner has a stable internet connection.
- The diner has already made a successful reservation.
- The time of the update/cancel operation is before the reservation time.

Success Guarantee (or Postconditions):

- Reservation is updated or canceled.
- The reservation system database is updated with the changes.
- The changes are reflected in the user interface of the application.

ReserveWell	
Use Case UC3: Update Reservation (Diner)	Date: <02/12/23>

Main Success Scenario (or Basic Flow):

1. System displays home page.
2. Diner accesses their existing reservations by clicking 'My Reservations' button.
3. System displays personal reservations page.
4. Diner selects the reservation they wish to update.
5. System enables editing.
6. Diner modifies the reservation details, such as date, time, party size, or special requests.
7. System shows the change info and asks for confirmation.
8. Diner confirms the changes to be updated in the system.
9. System updates the reservation and notifies the restaurant and sends a confirmation to the diner.

Extensions (or Alternative Flows):

*a. At any time, the diner needs to abandon the process,

1. Diner quits the page.
2. System asks to discard changes, review changes, save changes or cancel quitting,
   2a. Diner selects "discard changes".
      1. System reconstructs the prior state.
   2b. Diner reviews the changes.
      1. Diner chooses to discard changes.
         System reconstructs prior state.
      2. Diner chooses to save changes.
         System updates related data in real-time.
      3. Diner chooses to cancel quitting and continues where he/she left.

   2c. Diner saves the changes.
      1. System updates related data in real-time.

*b. At any time, System fails:

To support recovery and correct updates, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Diner restarts the system, logs in, and requests recovery of prior state.
2. System reconstructs the prior state.
   2a. System detects anomalies preventing recovery:
      1. System signals error to the diner, records the error, and enters a clean state.
      2. Records are automatically sent to support executives for a review.
      3. System displays home page.

4a. Diner selects an invalid reservation.

1. System informs the diner that the chosen reservation is not valid.
2. Diner returns to personal reservations page.

6a Diner tries to update the reservation with invalid information.

1. System informs the diner that the modifications are not valid (e.g., exceeding capacity).
2. Diner revises the changes.

Special Requirements:

- Invalid information or reservations should be detectable.

Frequency of Occurrence: Could be nearly continuous.

Open Issues: -