THE ICDE SYSTEM – BOMBAY BRASSERIE

Deliverable II

Pranav Jha[1], Parth Dineshkumar Patel[2], Hephzibah Pocharam[3] and Jayapriya Muthuramasamy[4]
[1,2,3,4]*Department of Electrical and Computer Engineering*
Concordia University, Montreal, Canada
[1]jha_k.pranav@live.com, [2]parth.mangukiya12@gmail.com,
[3]hepsissb9726@gmail.com, [4]jayapriya054@gmail.com

***Abstract*—The ICDE (Information and dissemination) system for the restaurant ”Bombay Brasserie” is designed using web services, allowing users/customers to access the restaurant’s website online via any device, including mobile phones, laptops, tablets, and iPads. The implementation of this system involves three stages: data capturing, storage, and assessment. Through these web services, users can conveniently order food online, make table reservations, and modify or change their bookings as needed. If there is a waitlist, users will receive notifications as soon as a seat becomes available. To access these features, users can either sign up as a customer or log in as a guest on the restaurant’s webpage. Furthermore, to enhance the quality of services offered, users will have the opportunity to provide ratings and feedback on various aspects such as the food, dining experience, service provided by the waitstaff, and overall management of the restaurant. This feedback can be collected via email or by following a link sent to their cell phones. By implementing these web services, the ”Bombay Brasserie” restaurant aims to provide a user-friendly platform for customers to conveniently access and interact with the restaurant’s services. The inclusion of online ordering, reservation management, and feedback mechanisms aims to enhance the overall dining experience and improve the quality of services provided.**

***Keywords*—ICDE, Python, Front-end, Back-end, Database, Server**

PART 1

I. AGILE PROCESS

A. Team setup

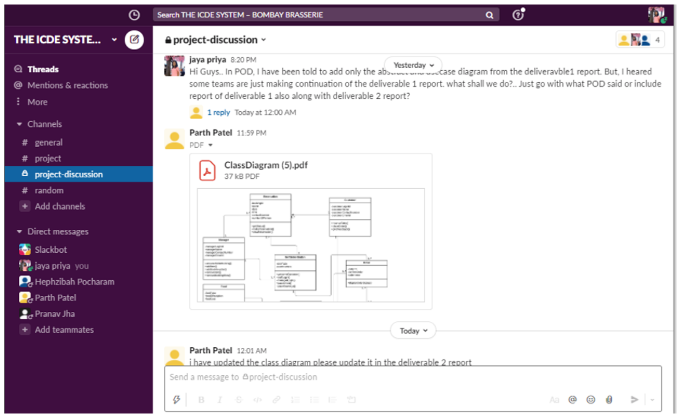
The team follows the Agile methodology for creating THE ”ICDE SYSTEM – BOMBAY BRASSERIE” website, which specifies the incremental approach model. The project scrum team members play different roles namely Scrum master, product owner and scrum team members.

Pranav Jha acts as scrum master. Scrum master helps to maintain the focus of the scrum team by conducting sprints. Monitoring the progress of each sprint and remove the factors that affect the development of the product. Providing report/ feedback on the current sprint and deals with the sprint backlogs.

Role of product owner is played by Parth Dinesh Kumar. Primary function is to know the exact need of the customers and defining it to the scrum team members to develop product accordingly. Most of the time customer defines the need, at times anticipation of client need is performed by product owner. Focus on the current product backlogs to get cleared in the upcoming sprint. Evaluate and track the progress of the product each sprint.

Jayapriya Muthuramasamy, Hephzibah Pocharam act as sprint development team members. Key role is to deliver the work in each sprint as planned based on product owner’s requirement and also be able to smartly convert product backlog to the practical case.

The current role might be interchanged in upcoming sprints based on convenience of the group. *Trello* is used for project task management. Effective team communication is done using *Slack* communication tool. Below is the snap of *Slack* communication of the scrum team. We use this for our group conversation regarding the project.



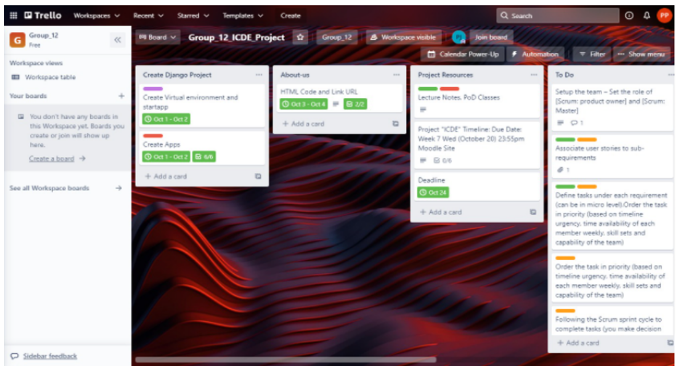
jaya priya 8:20 PM
Hi Guys.. In POD, I have been told to add only the abs[illegible]case diagram from the deliveravble1 report. But, I heared some teams are just making continuation of the deliverable 1 report. what shall we do?.. Just go with what POD said or include report of deliverable 1 also along with deliverable 2 report?
1 reply Today at 12:00 AM

Parth Patel 11:59 PM
PDF
ClassDiagram (5).pdf
37 kB PDF

Parth Patel 12:01 AM
i have updated the class diagram please update it in the deliverable 2 report

Send a message to project-discussion

Fig. 1. Slack group communication snap



You don't have any boards in this Workspace yet. Boards you create or join will show up here.
Create a board

Create Django Project
- Create Virtual environment and startapp — Oct 1 - Oct 2
- Create Apps — Oct 1 - Oct 2 — 6/6

About-us
- HTML Code and Link URL — Oct 3 - Oct 4 — 2/2

Project Resources
- Lecture Notes, PoD Classes
- Project "ICDE" Timeline: Due Date: Week 7 Wed (October 20) 23:55pm Moodle Site — 0/6
- Deadline — Oct 24

To Do
- Setup the team – Set the role of [Scrum: product owner] and [Scrum: Master] — 1
- Associate user stories to sub-requirements — 1
- Define tasks under each requirement (can be in micro level).Order the task in priority (based on timeline urgency, time availability of each member weekly, skill sets and capability of the team)
- Order the task in priority (based on timeline urgency, time availability of each member weekly, skill sets and capability of the team)
- Following the Scrum sprint cycle to complete tasks (you make decision

Fig. 2. Trello scrum sprint board snap

The Fig. 2 is a snap from our work space in Trello. This shows the tasks completed or in progress. The cards also shows the notes with specific label on it for our to do list. Once a certain task is completed, we change the label for that task such as ’in progress’ to ’complete’. Fig. 3 shows the task backlog for this deliverable 2. We will work on these tasks and will try to complete for the next deliverable.

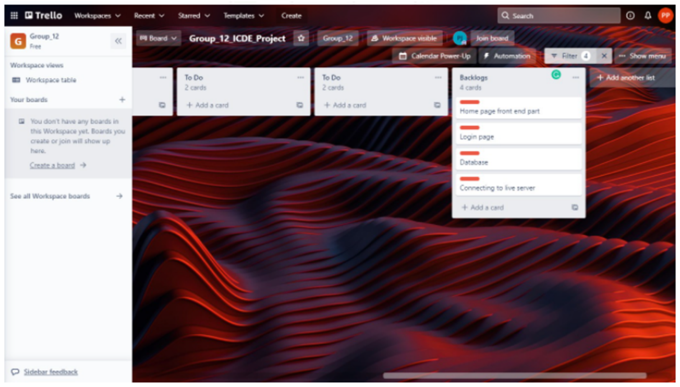


Fig. 3. Trello scrum backlog board snap

B. *Associating user stories to sub requirements:*

User Requirements

- 1. The user can see the menu of the restaurant and other details associated to it.
  - 1.1 The user would be able to contact the restaurant for any issue related to the services offered.
    - **Task**: ’Contact Us!’ app in the web page.
    - **Task Status**: Completed
  - 1.2 The user can see the menu list either on the home page or inside the ’Meals’ app on the web page.
    - **Task**: Add ’Meals’ app in the web page.
    - **Task Status**: Completed
- 2. The user would be able to register themselves.
  - 2.1 The user would be able continue as an unregistered user or guest.
  - 2.2 The user would be able to register at any point while searching for the order to be placed.
    - **Task**: Add ’User Registration and Login’.
    - **Task Status**: In progress.
- 3. The customer would be able to reserve a table for a specific time, or order meals or other food items available.
  - 3.1 The customer would be able to get assistance while processing for a specif service through call.
    - **Task**: Add ’Reserve A Table’ app in the web page.
    - **Task Status**: Completed
- 4. The customer would be able to modify, cancel their bookings or add a new booking.
  - 4.1 The customer would be able to modify or cancel the booking until the specified time for an existing booking.
    - **Task**: Add ’Order Now’ app in the web page.
    - **Task Status**: In progress.
  - 4.2 The customer would be able to add a new booking only before four hours from his existed booking.
  - 4.3 The customer would be able to get the full refund only if he cancels his booking(s) four hour in advance.
- 5. The customer would be able to write send a message to the restaurant through ’Contact Us!’ app inside the web page.

System Requirements

- 1. System shall display the web application with all the apps associated to it.
  - 1.1 The system shall be platform independent.
  - 1.2 System shall withstand loads at times.
    - **Task**: Add ’Scalability’ for the web page.
    - **Task Status**: In progress.
- 2. The system shall allow the user for registration based on his access level.
  - 2.1 The system shall allow a user to modify or change his information.
    - **Task**: Add ’User Registration and Login’.
    - **Task Status**: In Progress.
- 3. System shall save the data entered for registration and checks for duplicate data inside the system.
  - 3.1 The system shall encrypt the data for safety and privacy.
    - **Task**: Add ’Database’.
    - **Task Status**: Completed.
- 4. System shall display available tables, time-slots and the food menu to the customers.
- 5. System shall display the order details to the chefs and waiters/waitresses linked to the customers in the order it receives the requests.
- 6. System shall allow to add, change or modify certain data by a user with particular access level such as a manager.
  - 6.1 The system shall allow a user with particular access level such as managers to limit or update the number of people inside the restaurant at times.
    - **Task**: Add ’Order Management’.
    - **Task Status**: In Progress.
- 7. The system shall allow a user with particular access level, such as managers or staffs, to accept or reject an order request if the restaurant has reached it’s capacity for the requested time of the order.
  - **Task**: Add ’Accept/Reject’ control.
  - **Task Status**: In Progress.

- 8. The system shall allow a customer to pay using different methods of payment.
  - **Task**: Add 'Payment' option.
  - **Task Status**: In Progress.

C. Task under each requirement:

The tasks under specific requirements have been added in the previous section II-B. It also shows the status of a task whether it is 'in progress' or 'completed'. Further, the detailed user stories associated with these tasks is shown below with these use case description Tables I, II, III, IV, V and VI.

1) Use Case Descriptions: Users can see the details of the restaurant on it's web page such as the cuisine menu and their specialities, it's location and contact of the restaurant etc. When a user clicks on the sign up button, he/she is directed to the sign up page of the restaurant and can register themselves. According to the data provided by the user, the system shows them the options available. The details will be saved in the database server of the system in the back-end.

For all use cases, at any step, if a user would be able to cancel the use case linked to user input. This will end the use case and the system will not save any data collected from that use case. Also, For all use cases linked with logged in user, the current login session will be updated during the use case which will subsequently provide the navigation paths through the use case.

Login User:

In order to place orders, reserve a table or do other transactions, a user must login so that that the system can determine his access level.

TABLE I.

Use Case	**Login User**
Summary:	This use case allows a user to place orders, access personalized or restricted information based on his registered access level.
Basic Flow:	1. The use case starts when a user indicates to login on the restaurant website. 2. The system requests the username and password. 3. The user provides username and password to the system. 4. The system verifies the username and password against all registered users. 5. The system starts a login session and displays a welcome message based on the user's preferences.
Alternative Flows:	Step 4: if username is invalid, the use case goes back to step 2. Step 4: if the password is invalid the system requests that the user re-enter the password. When the user enters another password the use case continues with step 4 using the original username and new password.
Preconditions:	The user is registered.
Postconditions:	The user can now access data and carry out functions based on his registered access level.
Business Rules:	Some information are restricted to users with a particular access level.

Register User:

In order to get personalized or restricted information, place orders or do other specialized transactions a new user must register a username and password.

TABLE II.

Use Case	**Register User**
Summary:	The user must register a username and password in order to place orders, access personalized or restricted information based on his registered access level.
Basic Flow:	1. The use case starts when a user wants to register on the restaurant website. 2. The system requests a username and password. 3. The user enters a username and password. 4. The system checks that the username does not duplicate any existing registered usernames. 5. The system requests for address, phone number and email address with some fields star (*) marked. Items marked by (*) are mandatory fields. 6. The user enters the information. 7. The system determines the user's location and access level and stores all user information. 8. The system executes use case Register Preferences. 9. The system starts a login session and displays a welcome message based on the user's preferences.
Alternative Flows:	Step 4: If the username duplicates an existing username, a message is displayed and the use case repeats step 2. Step 5: If the user does not enter a required field, the system displays a message and the use case goes back to step 4.
Preconditions:	The user is not registered.
Postconditions:	The user can now access data and carry out functions based on his registered access level.
Business Rules:	Some information are restricted to users with a particular access level.

TABLE III.

Use Case	**Web App Display** Scenario: Customer visits the restaurant web page.
Summary:	This use case allows an unregistered user to see the web app or place an order without being registered in the system.
Basic Flow:	1. The use case starts when a user wants to visit the restaurant web app. 2. The user indicates to see the app list on the home page of the web app such as meals, reserve a table, contact us and so on. 3. The user wants to see the menu list, reserve a table for a specific time or wants to use any other service offered by the restaurant. 4. The system will take user to his indicated app inside the webpage. 5. The intended app will display the menu details for the meal app, a form for reserve a table app and so on.
Alternative Flows:	Step 4: If the user stays on the web app inactive for over a minute or so, the system displays a message ”Need Help!” and the system will show a pop up table asking user to enter the query.
Preconditions:	The user is not registered.
Postconditions:	The user can register themselves or continue as a guest.
Business Rules:	For placing an order or book a table as an unregistered user, the user must provide his contact details at chekout app.

TABLE IV.

Use Case	**Register Preferences**
Summary:	This use case allows a registered user to enter his preferences or to modify an existing order information in the cart.
Basic Flow:	1. The use case starts when a user indicates that he wants to add or modify the existing order preferences in the system. 2. The system displays the order that the user wants to modify. 3. The user modifies the order. 4. The system stores any change done by the user.
Alternative	none
Preconditions:	The user is logged in.
Postconditions:	The system can customize a welcome message based on the user’s updated information in the system.
Business Rules:	Some information can only be modified by a user with particular access level.

TABLE V.

Use Case	**Place an Order (Customer)** Scenario: Customer places an order in the system.
Summary:	This use case allows a registered customer to place an order.
Basic Flow:	1. The use case starts when a customer indicates that he wants to place an order. 2. The system displays the order information that the customer wants to place. 3. The customer may add or modify his order. 4. The system stores any change done by the customer. 5. The system displays the methods of payment available to this customer. 6. The customer selects a payment method. 7. The system completes the payment by executing use case ’Charge Customer’ based on the selected method of payment. 8. The system stores the order and shipping information. 9. The system displays a message for successful order completion and sends a receipt to the customer.
Alternative	Step 10: If the selected payment method could not be validated, the use case goes back to step 6.
Preconditions:	The user is logged in and searched for the order to be placed.
Postconditions:	The order has been placed.
Business Rules:	The customer must pay in full at the time of placing order.

TABLE VI.

Use Case	**Charge Customer**
Summary:	This use case charges the customer for the order currently being placed to the selected method of payment.
Basic Flow:	1. The use case starts when a customer selects ”Credit Card” as a payment method, while in ’Place An Order’ use case. 2. The system requests the credit card number, type and expiration date. 3. The user enters the information. 4. The system checks the validity of the card for the amount to be charged and completes the transaction. 5. The system stores the payment details and displays a success message.
Alternative	If the credit card cannot be validated, the system displays a failure message and use case ends.
Preconditions:	The system is executing ’Place An Order’ use case.
Postconditions:	The customer has been charged for the order.
Business Rules:	Credit cards accepted are Visa and MasterCard only.

D. Scrum sprint cycle:

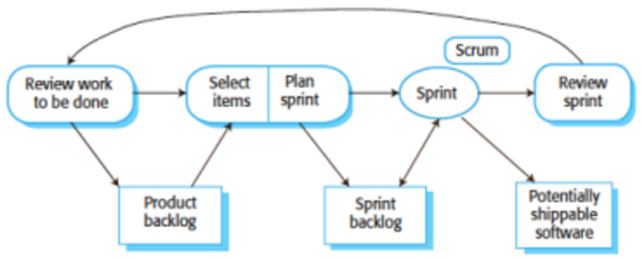


Fig. 4. Scrum sprint cycle [1]

Referring to the Fig. 4 All user requirements are divided into individual tasks. Significant tasks are assigned with high priority status and the timeline for completion of the same is fixed beforehand when a specific task is assigned to among the scrum team. A proper sprint plan is made to track the progress of the designated tasks. During sprint, the scrum master (Pranav Jha) checks for the tasks that were planned to complete and provides the feedback. Any improvisation of the task or the backlogs is considered for the next sprints along with the next sprints tasks and then the same process repeats. Currently, team is working on the main user requirement namely 1,2 and 3. Remaining user requirements such as login page creation and adding item to the cart, etc are planned to complete in the upcoming sprints. The product backlog is tracker and cleared by product owner Parth Dinesh Kumar by maintaining a good communication with the development team members Hephzibah Pocharam and Jayapriya Muthuramasamy.

II. System Model and Design

A. Use case diagram

1) Overall level use case diagram: The Fig. 5 below shows the use case diagram of the restaurant We have 4 users in this use case. Guests can see the menu and details of restaurant. They can also register in order to book the table. Waiter/waitress and chefs can register themselves and check the orders received. Manager can edit menu and also update the available time slots.

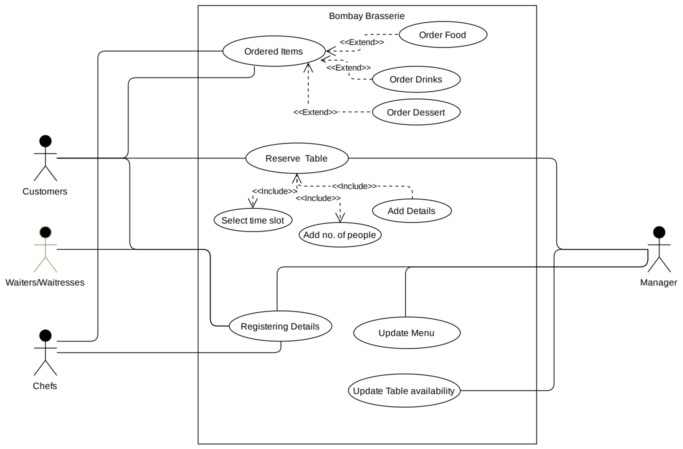


Fig. 5. Use case diagram

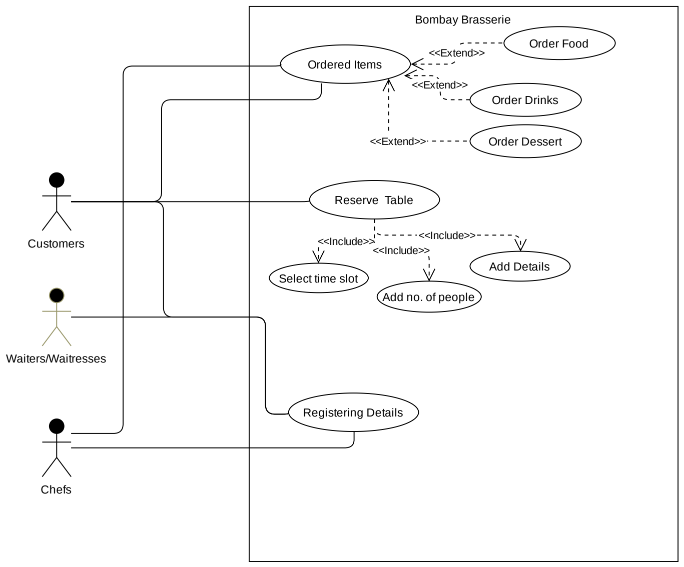


Fig. 6. User requirement use case diagram

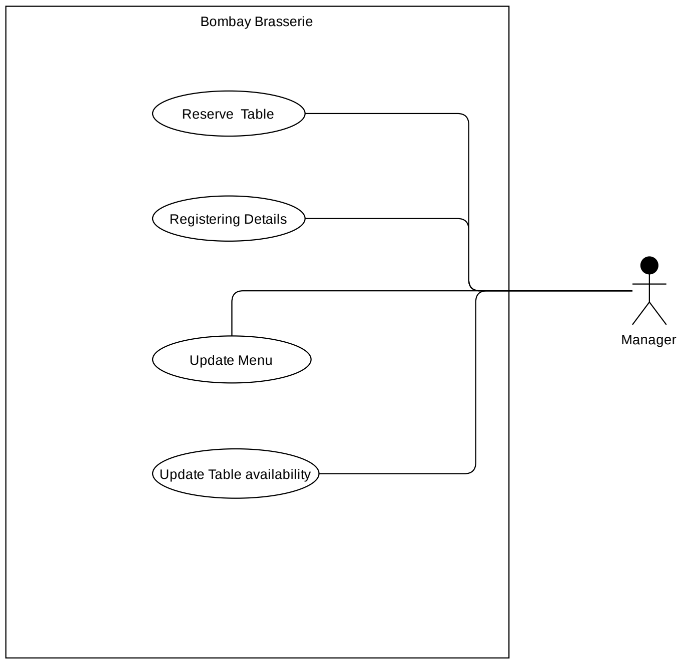


Fig. 7. System requirement use case diagram

2) System/user requirement use case diagram:

B. Architecture

There are number of steps that need to be fulfilled when someone places an order on the restaurant website. These are mentioned in the following subsections.

Web Server: The list of contents such as the ready to go kind of foods, are stored in a memory on back-end servers and can be easily find by the chef and send out instantaneously to the customers upon receiving a request or ordered through the front-end web application. Also, there are memory cache available for the web server which stores information about some food or snacks items and can help in preparing and serving them straightaway.

Database: There are different kinds of data which need to be stored securely in a database server. We are using MySQL to store data. These data take some time to fetch from the database server to the web application. These are well sorted and can be find using some steps in the database server when requested. The steps involved in fetching a data from a back-end application of the database server to the front-end are

1) Connecting to the database
2) Creating a query
3) Sending the query
4) Wait for the response
5) Processing the error
6) Handling of data

After all these steps, the chef has the proper data in hand and can respond to the customer.

Data Formatting: The data coming from the database needs to be formatted. We are using JSON to format our data and store it on the web server. This can be useful when someone requests the same information again.

The aforementioned steps show the processes for our application from requests in the front-end to responses from the back-end.

1) High level architecture diagram: Major components of the system are illustrated in the high level architecture diagram shown below in Fig. 8. The overall work flow of data capture, data storage, data access and its related blocks are shown.

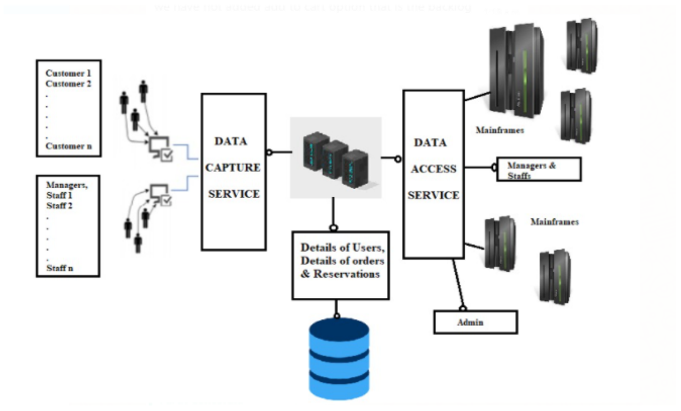


Fig. 8. High Level Architecture Diagram for the ICDE SYSTEM – BOMBAY BRASSERIE

2) System level architecture diagram: The relationship between between the user interface in the front-end and the back-end application is clearly explained in the system level architecture diagram as shown below in Fig. 9. Database, web service and web interface in the system level architecture diagram depict that the ICDE system of Bombay Brasserie is a web based application.

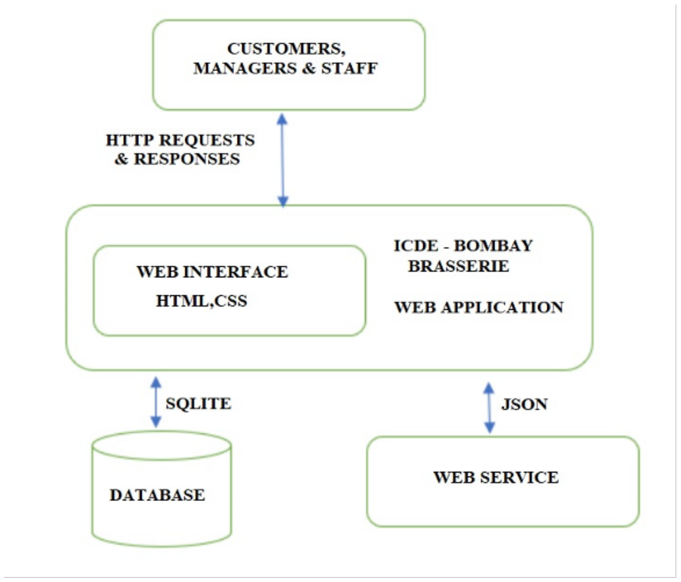


Fig. 9. System Level Architecture Diagram for the ICDE SYSTEM – BOMBAY BRASSERIE

C. Task implementation with modeling and design

1) Class diagram: Class diagram is made for the complete THE ICDE SYSTEM – BOMBAY BRASSERIE. Attributes and its function/methods are clearly explained in the class diagram shown above. Data type is not mentioned as the language used is python. It is shown that the food drink and dessert are the derived class of the parent class 'Menu'. Similarly 'staff' is the parent class which is derived into chef and waitress. The relation 1 to 1..* in the diagram represents that the relation is one to many relation (such as one customer can reserve many tables).- represents the access modifier is private (such as: customer login Id), '+' represents that the access modifier is public [such as: placeorder()].

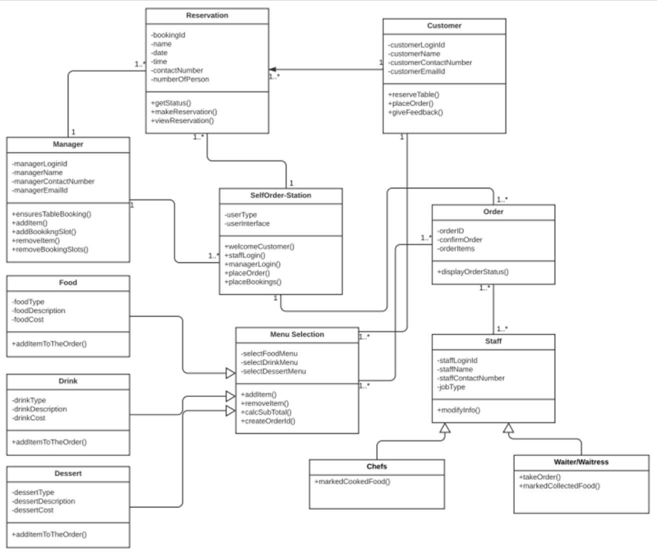


Fig. 10. Class Diagram for the ICDE SYSTEM – BOMBAY BRASSERIE

2) Activity diagram: Activity diagram in Fig. 11 is made for a single task of 'selecting the menu' as clearly shown in the above picture. One the customer logins in using username and password (considered as the sprint backlog), it will take the customer to the Home Page where menu option will be available. Once the menu option is selected, Customer will be displayed with three options namely select food, select drink and select dessert. According to the option selected by the customer, the details of the same will be shown. For example, when food option is selected, The details option will show the food description, price and the no of people, etc,.

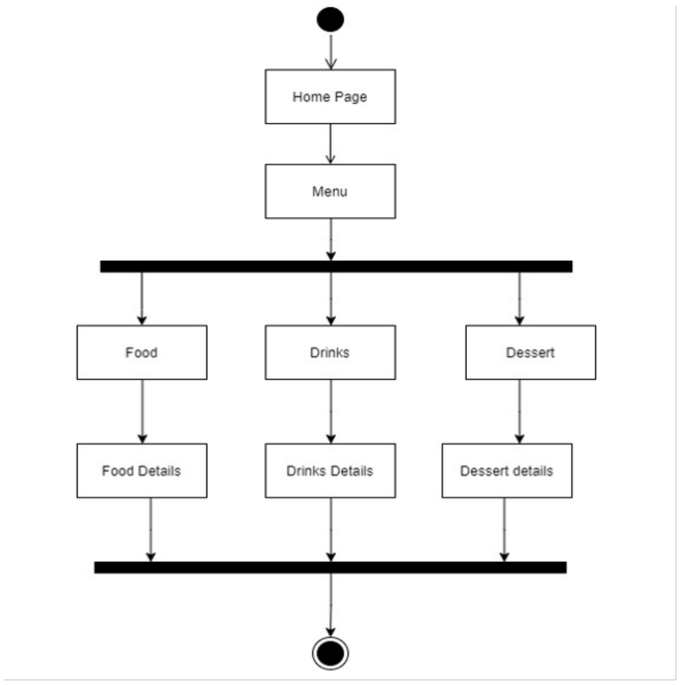


Fig. 11. Activity diagram for user requirements

Customer activity diagram:

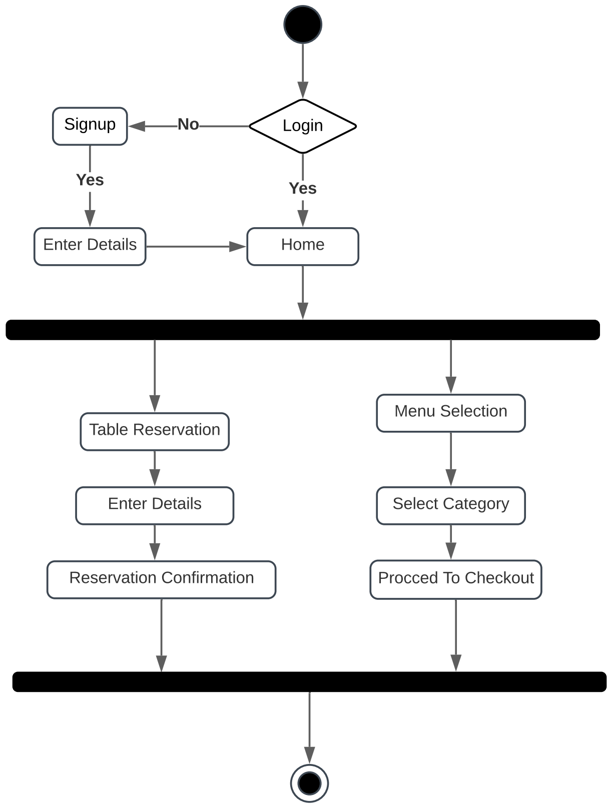


Fig. 12. Customer activity diagram

Manager activity diagram:

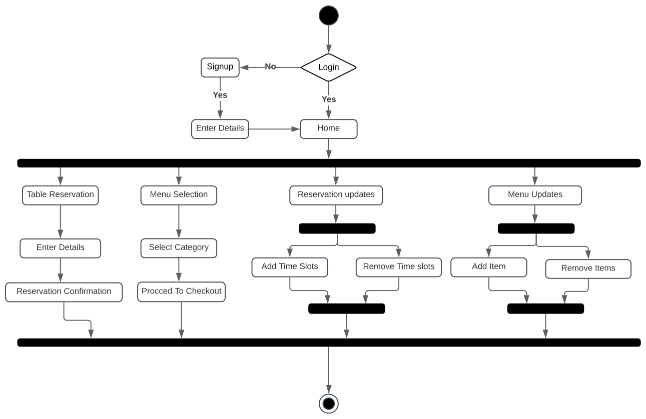


Fig. 13. Manager activity diagram

3) Sequence Diagram: Sequence Diagram as shown in Fig. 14 is nothing but the interaction diagram that helps to provide the way the tasks are being done with the help of good interaction between the objects. The order in which the interaction takes place is considered to be significant. Here, the sequence diagram is again made for the same task for which the activity diagram is done. Once the customer click on menu option in the interface, the feedback "selected menu option" will be provided and then it takes the customer to the page where drink food and dessert option are available. Once the user/customer selects the drink option on the interface, the same command will be sent to database where details of the drink (such as drink content, price, etc) is available. As a feedback, user will be displayed with the drink details. Same procedure goes with food and dessert.

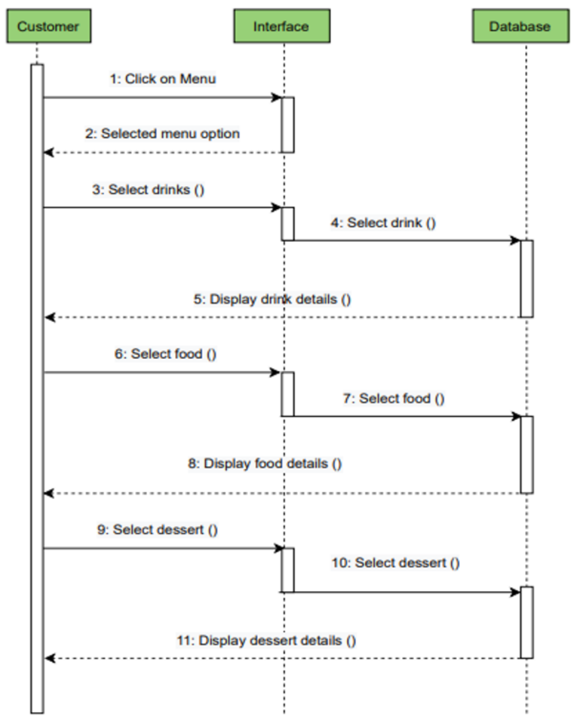


Fig. 14. Sequence diagram for user requirements

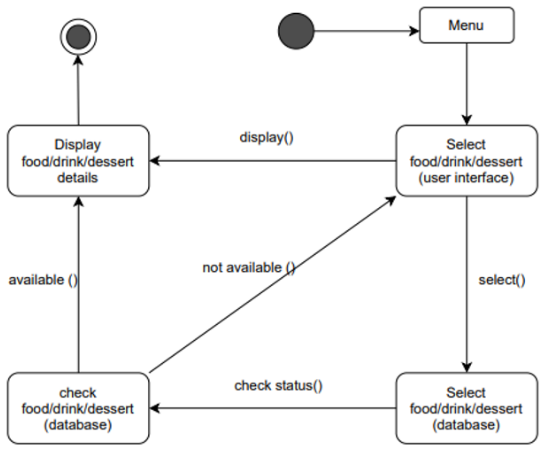

Menu
display()
Display food/drink/dessert details
Select food/drink/dessert (user interface)
available ()
not available ()
select()
check status()
check food/drink/dessert (database)
Select food/drink/dessert (database)

Fig. 15. Statement diagram for user requirements

4) Statement diagram: In Fig. 15, the statement diagram is shown for the user requirements for the web ICDE application. The statement diagram basically Statements provide a structured description of the parent Use Case. When a user wants to select an order, he has to go through some steps. We ca understand this using an example in this context. As shown in Fig. 15, a user can select the menu from the 'Meals' app of the web page. Then, he will see the menu list where all food item can be shown categorically. The user will select a category based on his preference and would continue further. The system will check for the selected item for it's availability in the database. If the selected item is not available, the user will go back to the menu list page and select another category or another item inside chosen category and proceed further which will again go through the same steps. In this way, the user would be able to order his choice of item based on the availability of the item in the database. This meal selection use case is just one among other use cases which is shown by the statement diagram.

PART 2

III. NEW TASK

A. *User Login Creation:*

This task has been chosen by our team and we are working on it.

1) Class diagram:: The class diagram in Fig. 10 remains the same also for this task.

2) Sequence diagram:: Fig. 16 shows the sequence diagram for this task.

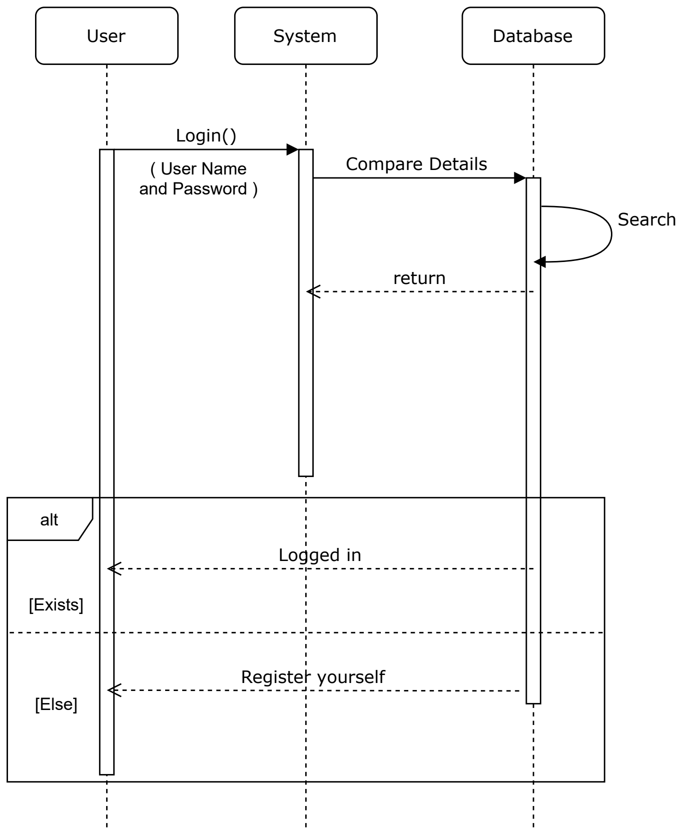


Fig. 16. Sequence diagram for user requirements

3) Statement diagram:: Fig. 17 shows the statement diagrams for this task.

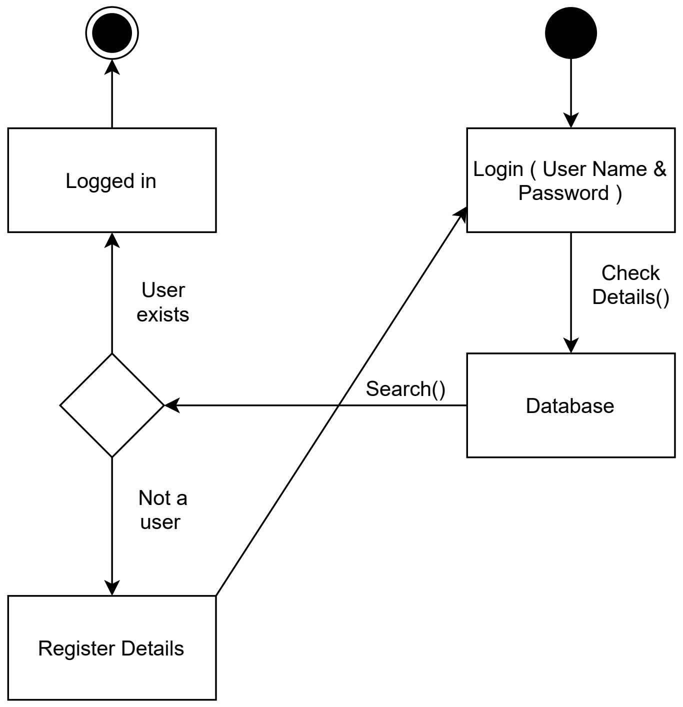


Fig. 17. Statement diagram for user requirements

Backlog: The activity diagram is still in the development process.

B. Evaluating the progress:

We have divided the project development for our ICDE app in three phases as mentioned below:

1) **Initial Phase:** In this phase, we decided to go through a rough sketch of the whole system, prepared a list of required software products, such as Python installation, code editor which is Visual studio, overleaf for document preparation services etc., and installed them on our local system. Further, we used *Lucidchart* and *Draw.io* to draw our figures.
2) **Implementation Phase:** In the next phase, we started building our ICDE app using Python and Django based web framework. We have built our front-end and working to build our back-end servers and also, trying to add other required services, user authentication and payment methods in our web page. This wil be completed in next one to two weeks period.
3) **Deployment Phase:** In this phase, we will deploy our fully functional web ICDE app to the live server. This will be done in coming weeks when we will finish our implementation phase.

References

[1] I. Sommerville, "Software engineering. 10th," in *Book Software Engineering. 10th, Series Software Engineering*. Addison-Wesley, 2015.