1. **INTRODUCTION:**

As the internet is growing fast and large group of people have access to the internet, people started looking through internet instead of visiting places directly and started taking help of the internet to look for places to eat.

* 1. **Purpose:**

Johnny On The Spot is the perfect help for people in need who want to look for places to eat in a unknown place. Johnny creates a sense of relief when you are in a strange place and all you want is a good place to eat! All you got to do is open the page and click on search nearby restaurants and Eureka! You get the restaurant within seconds!

The proposed Online Restaurant Search System is designed keeping both customer and restaurant in the mind so that customers can look for restaurants online and place order with the help of interacting menu in the future. It is an Online Restaurant Search System, a comprehensive Restaurant Application which allows customers to look for places online as well as helps restaurants to manage their menus and their new dishes. The system employs the latest, state-of-the-art technology and operates under a local network, combined with external Intranet and Internet networks.

* 1. **Scope:**

Due to the great increase in the awareness of internet and the technologies associated with it, several opportunities are coming up on the web. So many businesses and companies now venture into their business with ease because of the internet. One of such business that the internet introduced is an online restaurant search system.

* 1. **Application Design:**

Online Restaurant Search is a platform/website through which any user can search the best restaurant registered at that location and the Review/Rating option help the user to select the best one. The best part of this website is that any restaurant can update their “Offer of the day” which helps for the attraction of user.

* 1. **Overview:**

Online Restaurant Search named “Johnny on the Spot” is a webpage through which any Restaurant can register itself for free and update their Menu and Offers regularly so that any Guest user can search the Restaurant at his desired location and select one with the help of Review & Rating. But if the user wants to rate any restaurant or write some review, first he will have to register.

1. **OVERALL DESCRIPTION:**
   1. **Product Perspective:**

The main perspective of this website is to reduce the effort of user for finding the best restaurant at the desired location. There are lots of modules given to the user to make their search easy and effective. The user just need to enter his desired location and a list of restaurants will come and then he can choose the best restaurant according to recent reviews & ratings.

* The home page of this web page provides an avenue where customers will be able to gather more and reliable information about the various restaurants in the searched area.
* A restaurant registration page which would make the restaurant involve directly in the processing of the system smoothly without any confusion.
* Restaurants could easily update their offers on the daily basis or weekly basis according to their choice and preferences.
* A social feature which would allow the user to rate and review the restaurants which would in turn help other customers to choose a good restaurant according to the rating.
* A user registration page which would help in decreasing the duplicity since the review and ratings could only be done once a user register himself or herself.
* The updating of the menu and various offers would be strictly done by the restaurant administrator itself, no outsider could change those information.
  1. **Software Interface:**

This project Online Restaurant Search will run on windows and any other operating system. It can be accessed as webpage when it will be deployed on the server.

* **Operating System:** Windows (any version), Linux, Mac and other OS also.
* **Web Development Tool:** Macromedia Dreamweaver 8.
* **Browser:** Google chrome, Mozilla Firefox, Internet Explorer and others.
* **Web Server:** This software is being designed to run on **Wamp** server.
* **Database:** The system will access MYSQL database developed by Microsoft for providing the backend support for various feature included later like tutorial series.

|  |  |
| --- | --- |
| **Operating System** | Windows, Linux, Mac etc. |
| **Front End** | PHP, HTML, CSS, JAVAScript |
| **Back End** | MySQL |
| **Documentation** | Microsoft word |
| **Presentation** | Microsoft PowerPoint |

* 1. **Hardware Interface:**
* The project will work when **wamp** server is installed in the system.
* The hardware interface of the system is handled by the windows operating system. No hardware dependent code is written by the development team.

**Technology Used:**

* **PHP:** PHP is used for server side scripting and to establish the connection between webpage and Database.
* **HTML:** It is used to develop Web pages in the project and giving eye catching look to this site.
* **MySQL:** It is used to store the various data and image files of this Webpage.

**Performance Requirements:**

* **Client Hardware:** The performance of system does not depend on the client machine.
  1. **Product Function:**

The main purpose of Online Restaurant Search (named Johnny on the Spot) is to input the desired location and search for the registered restaurants available at that location as output. Now the user can visit the restaurant’s page as per their rating and further contact them through their contact details.

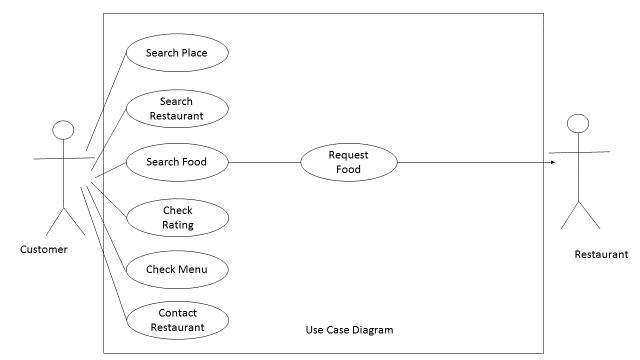
The key function of this project is:

* Input the desired location and shows the list of all registered restaurants at that location.
* Restaurants can update their Offer of the Day which help to be eye catching or attractive for users.
* Rating & Review option gives freedom to user to show their opinion about the Restaurant.
* Query & Feedback option is very beneficial so that anyone can ask any question and admin will clear their doubts.
  1. **User Characteristics:**
* **Educational level:** Users should be comfortable with the English language.
* **Internet:** User should have proper internet connection.
* **Experience:** Users should have prior information regarding the present scenario of online restaurant services.
* **Skills:** Users should have basic knowledge and should be comfortable using general purpose applications on computers.
* **Trust:** User should be ready to trust the details given on our website.
  1. **Constraints:**

Due to absence of gateway, the payment mode cannot be done online. It is something which will be done as a future scope of our project. Online orders as well as cash on delivery is not available on our website, we will be working on this as future scope.

* 1. **Architecture Design:**
  2. **Use Case Model Description**

The Use Case Diagram is shown below –



**Class Diagram**:

In [software engineering, a](https://en.wikipedia.org/wiki/Software_engineering) class diagram in the [Unified Modelling Language (](https://en.wikipedia.org/wiki/Unified_Modeling_Language)UML) is a type of static structure diagram that describes the structure of a system by showing the system's [classes,](https://en.wikipedia.org/wiki/Class_(computer_science)) their attributes, operations (or methods), and the relationships among objects.

The class diagram is the main building block of [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) modelling. It is used both for general [conceptual modelling](https://en.wikipedia.org/wiki/Conceptual_model) of the systematics of the application, and for detailed modelling translating the models into [programming code. Class](https://en.wikipedia.org/wiki/Programming_code) diagrams can also be used for [data modelling.](https://en.wikipedia.org/wiki/Data_modeling) The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

In the diagram, classes are represented with boxes that contain three compartments:

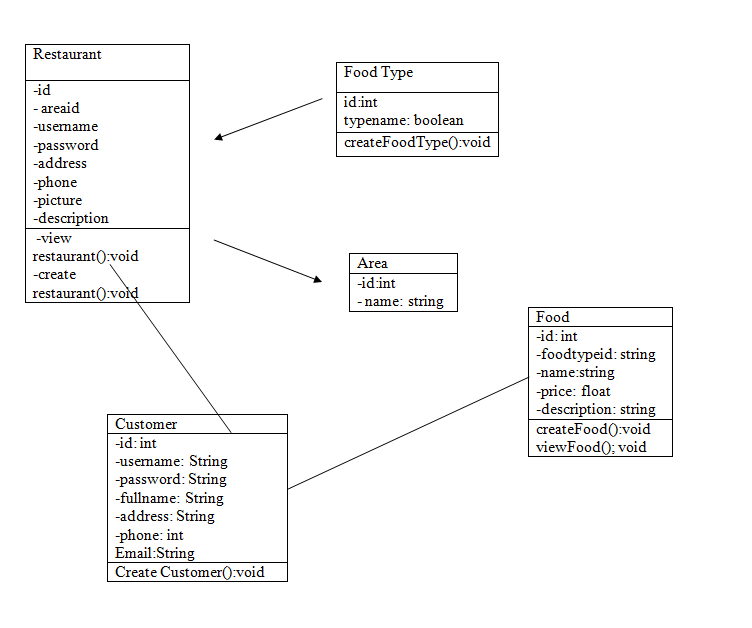
The top compartment contains the name of the class. It is printed in bold and centred, and the first letter is capitalized.

The middle compartment contains the attributes of the class. They are left-aligned and the first letter is lowercase.

The bottom compartment contains the operations the class can execute. They are also left- aligned and the first letter is lowercase.

In the design of a system, a number of classes are identified and grouped together in a class diagram that helps to determine the static relations between them. With detailed modelling, the classes of the conceptual design are often split into a number of subclasses.

In order to further describe the behaviour of systems, these class diagrams can be complemented by a [state diagram or](https://en.wikipedia.org/wiki/State_diagram) [UML state](https://en.wikipedia.org/wiki/UML_state_machine)



Class Diagram Explanation:

|  |
| --- |
| **Restaurant** |
| <Attributes>  private int id;  private int areaid;  private String username;  private String password;  private String phone;  private String address;  private String picture;  private String description;  ; |
| <Operations>  public void createRestaurant()  public void viewRestaurant () |

### Food

|  |
| --- |
| **Food** |
| <Attributes>  private int id;  private int foodtypeid;  private String name;  private float price;  private String picture;  private String description; |
| <Operations>  public void createFood()  public void viewFood () |

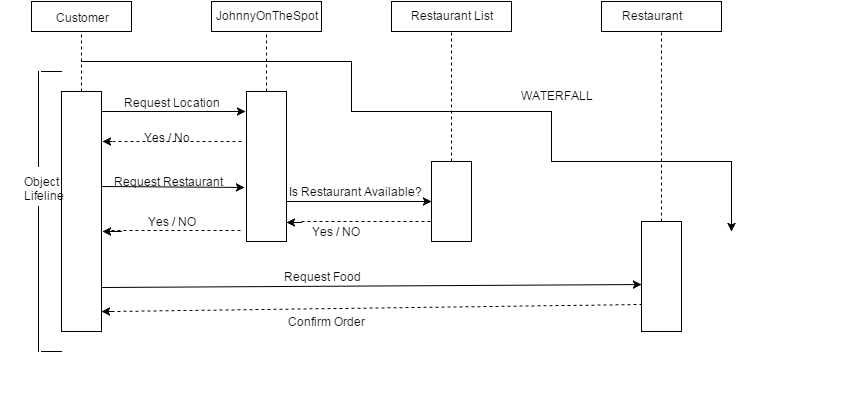
### Customer

|  |
| --- |
| **Customer** |
| <Attributes>  private int id;  private String username;  private String password;  private String fullname;  private String address;  private String email;  private String phone; |
| <Operations>  public void createCustomer() |

**2.9 Sequence Diagram**

A Sequence diagram is an [interaction diagram](https://en.wikipedia.org/wiki/Interaction_diagram) that shows how processes operate with one another and in what order. It is a construct of a [Message Sequence Chart](https://en.wikipedia.org/wiki/Message_Sequence_Chart). A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines (*lifelines*), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.



**2.10 Database Design**

Database design is the process of producing a detailed [data model](https://en.wikipedia.org/wiki/Data_model) of a [database](https://en.wikipedia.org/wiki/Database). This [data model](https://en.wikipedia.org/wiki/Data_model) contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a [data definition language](https://en.wikipedia.org/wiki/Data_definition_language), which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity.

The term database design can be used to describe many different parts of the design of an overall [database system](https://en.wikipedia.org/wiki/Database_system). Principally, and most correctly, it can be thought of as the logical design of the base data structures used to store the data. In the [relational model](https://en.wikipedia.org/wiki/Relational_model) these are the [tables](https://en.wikipedia.org/wiki/Database_table) and [views](https://en.wikipedia.org/wiki/Database_view). In an [object database](https://en.wikipedia.org/wiki/Object_database) the entities and relationships map directly to object classes and named relationships. However, the term database design could also be used to apply to the overall process of designing, not just the base data structures, but also the forms and queries used as part of the overall database application within the [database management system](https://en.wikipedia.org/wiki/Database_management_system) (DBMS).

The process of doing database design generally consists of a number of steps which will be carried out by the database designer. Usually, the designer must:

* Determine the data to be stored in the database.
* Determine the relationships between the different data elements.
* Superimpose a logical structure upon the data on the basis of these relationships

**2.10.1 E-R Diagram**

An entity–relationship model (ER model) is a [data model](https://en.wikipedia.org/wiki/Data_modeling) for describing the data or information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a [database](https://en.wikipedia.org/wiki/Database) such as a [relational database](https://en.wikipedia.org/wiki/Relational_database). The main components of ER models are [entities](https://en.wikipedia.org/wiki/Entities) (things) and the relationships that can exist among them.

An entity–relationship model is the result of using a systematic process to describe and define a subject area of business data. It does not define business process; only visualize business data. The data is represented as components (entities) that are linked with each other by relationships that express the dependencies and requirements between them, such as: one building may be divided into zero or more apartments, but one apartment can only be located in one building. Entities may have various properties (attributes) that characterize them. Diagrams created to represent these entities, attributes, and relationships graphically are called entity–relationship diagrams.

An ER model is typically implemented as a [database](https://en.wikipedia.org/wiki/Database). In the case of a [relational database](https://en.wikipedia.org/wiki/Relational_database), which stores data in tables, every row of each table represents one instance of an entity. Some data fields in these tables point to indexes in other tables; such pointers are the physical implementation of the relationships.

The [three schema approach](https://en.wikipedia.org/wiki/Three_schema_approach) to [software engineering](https://en.wikipedia.org/wiki/Software_engineering) uses three levels of ER models that may be developed.

[**Conceptual data model**](https://en.wikipedia.org/wiki/Conceptual_data_model)

This is the highest level ER model in that it contains the least granular detail but establishes the overall scope of what is to be included within the model set. The conceptual ER model normally defines master reference data entities that are commonly used by the organization. Developing an enterprise-wide conceptual ER model is useful to support documenting the [data architecture](https://en.wikipedia.org/wiki/Data_architecture) for an organization.

A conceptual ER model may be used as the foundation for one or more logical data models (see below). The purpose of the conceptual ER model is then to establish structural [metadata](https://en.wikipedia.org/wiki/Metadata) commonality for the [master data](https://en.wikipedia.org/wiki/Master_data) entities between the set of logical ER models. The conceptual data model may be used to form commonality relationships between ER models as a basis for data model integration.

[**Logical data model**](https://en.wikipedia.org/wiki/Logical_data_model)

A logical ER model does not require a conceptual ER model, especially if the scope of the logical ER model includes only the development of a distinct information system. The logical ER model contains more detail than the conceptual ER model. In addition to master data entities, operational and transactional data entities are now defined. The details of each data entity are developed and the relationships between these data entities are established. The logical ER model is however developed independent of technology into which it can be implemented.

[**Physical data model**](https://en.wikipedia.org/wiki/Physical_data_model)

One or more physical ER models may be developed from each logical ER model. The physical ER model is normally developed to be instantiated as a database. Therefore, each physical ER model must contain enough detail to produce a database and each physical ER model is technology dependent since each [database management system](https://en.wikipedia.org/wiki/Database_management_system) is somewhat different.

The physical model is normally instantiated in the structural metadata of a database management system as relational database objects such as [database tables](https://en.wikipedia.org/wiki/Database_table), [database indexes](https://en.wikipedia.org/wiki/Database_index) such as [unique key](https://en.wikipedia.org/wiki/Unique_key) indexes, and database constraints such as a [foreign key constraint](https://en.wikipedia.org/wiki/Foreign_key_constraint) or a commonality constraint. The ER model is also normally used to design modifications to the relational database objects and to maintain the structural metadata of the database.

The first stage of [information system](https://en.wikipedia.org/wiki/Information_system) design uses these models during the [requirements analysis](https://en.wikipedia.org/wiki/Requirements_analysis) to describe information needs or the type of [information](https://en.wikipedia.org/wiki/Information) that is to be stored in a [database](https://en.wikipedia.org/wiki/Database). The [data modeling](https://en.wikipedia.org/wiki/Data_modeling) technique can be used to describe any [ontology](https://en.wikipedia.org/wiki/Ontology_(computer_science)) (i.e. an overview and classifications of used terms and their relationships) for a certain [area of interest](https://en.wikipedia.org/wiki/Universe_of_discourse). In the case of the design of an information system that is based on a database, the [conceptual data model](https://en.wikipedia.org/wiki/Conceptual_data_model) is, at a later stage (usually called logical design), mapped to a[logical data model](https://en.wikipedia.org/wiki/Logical_data_model), such as the [relational model](https://en.wikipedia.org/wiki/Relational_model); this in turn is mapped to a physical model during physical design. Note that sometimes, both of these phases are referred to as "physical design."

**2.10.2 Data Flow Diagram**

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an [information system](https://en.wikipedia.org/wiki/Information_system), modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs can also be used for the [visualization](https://en.wikipedia.org/wiki/Data_visualization) of [data processing](https://en.wikipedia.org/wiki/Data_processing) (structured design).

A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel (which is shown on a [flowchart](https://en.wikipedia.org/wiki/Flowchart)).

**Level 0 DFD:**

**Level 1 DFD:**

**Level 2 DFD:**

**2.11 Assumptions and Dependencies**

**4.0 TESTING AND MAINTAINENCE**

**4.1 Introduction:**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies and/or a finished product. It is the process of exercising with the intent of ensuring that the

Software system meets its requirements and user expectation and does not fail in an unacceptable manner. There are various types of test. Each test type address a specific testing requirement.

**4.2 TYPES OF TESTING:**

**4.2.1 UNIT TESTING:**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that the programs inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software unit of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit test performs basic test at component level and test a specific business process, application, and/ or system configuration. Unit tests ensures that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**4.2.2 INTEGRATION TESTING:**

Integration test is designed to test integrated software components to determine if they actually runs as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration test demonstrate that although the components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**4.2.3 FUNCTIONAL TESTING:**

Functional tests provide systematic demonstrations that function tested are available as specified by business and technical requirements, system documentation, and user manuals.

Functional testing is centred on the following items:

Valid Input : Identifies classes of valid input must be accepted.

Invalid input : Identified function must be rejected.

Functions : Identified functions must be exercised.

Output : Identified classes of application output must be exercised.

System : Interfacing system must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or Special test cases. In addition, systematic coverage pertaining to identify Business process flows; Data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**4.2.4 SYSTEM TESTING:**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**4.2.5 WHITE BOX TESTING:**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least it’s Purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**4.2.6 BLACK BOX TESTING:**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works

**4.3 Unit testing**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach Field testing will be performed manually and functional tests will be written in detail.

Test objectives

 All entries must work properly.

 Pages must be activated from the identified link.

 The input screen, messages and responses must not be delayed.

Features to be tested

 Verify that the entries are of the correct format.

 No duplicate entries should be allowed.

**4.4 Integration testing**

Software integration testing is the incremental integration testing of two or more integrated Software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one steps up – software applications at the company level – interact without error.

**4.5 Acceptance testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.