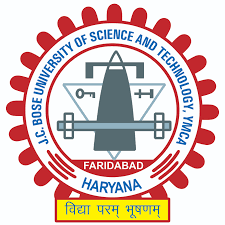
**INDUSTRIAL TRAINING REPORT**

## ON

**FOOD RUNNER APP**

## HELD AT

**Dreamer Infotech**



submitted in partial fulfilment of the requirement for the award of the degree

**Bachelor of Technology**

(Computer Science and Engineering Department)

Under the Supervision of

SUBMITTED TO: SUBMITTED BY:

Ms. Hema Rathaun Aditya Kumar

Roll no: 18011004006

**Aravali College of Engineering and Management**

R-Zone Jasana-Tigaon Road, Greater Faridabad, Haryana 121006

(2018-2022)



## CANDIDATE DECLARATION

I hereby, certify that the work which is being presented in the report entitled, “**FOOD RUNNER APP**” , in the partial fulfillment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND

ENGINEERING respectively, submitted in the COMPUTER SCIENCE department, Aravali College of Engineering and Management, affiliated to J.C BOSE UNIVERSITY OF SCIENCE AND TECHNOLOGY Faridabad, Haryana (India), is an authentic record of my own work carried out during session 2021-21,under the supervision of  **Mr SP Awasthi** Assistant Professor, CSE department, Aravali College of Engineering and Management.

The matter embodied in this project report has not been submitted by me for the award of any other degree or diploma.

**Dated: 30-06-2022 Aditya Kumar**

## ACKNOWLEDGEMENT

I would like to take this opportunity to express my profound sense of gratitude and respect to all those who helped me throughout the duration of this project. I acknowledge the efforts of those who have contributed significantly to my project. I express my heartiest thanks to all who selflessly assisted me in my expedition of carrying out this project.

I feel privileged to offer our sincere thanks and deep sense of gratitude to **Ms Sakshi kumar (HOD, CSE)** for expressing his confidence in me by assisting and giving me liberal encouragement as well as moral support not only during this project, but also throughout my studies at the college.

Furthermore, we are grateful for the co-operation and valuable suggestions rendered **Ms. Hema Rathaun** (Project Guide) aunder whose supervision and inspiring guidance, this project was embarked upon, planned and executed. Their sincere suggestions helped me greatly in bringing out this work at its present shape.

At last, but not the least the constant source of inspiration from my parents, faculty members of ACEM, CSE dept. as well as the library and friends that made me put my hard work and dedication, with the help of which I have come all this way.

### Dated: 30-06-2022 Aditya kumar

## TABLE OF CONTENT

### Introduction 4-6

* 1. Introduction to the project
  2. Purpose of the project
  3. Problems in the existing system
  4. Solution to these problems

### Feasibility Study 7-9

* 1. Technical Feasibility
  2. Operational Feasibility
  3. Economic Feasibility

### System Analysis 10-15

* 1. Study of the system
  2. Proposed system
  3. Input and Output
  4. Process models used with justifications.

### Software Requirement Specifications 16-35

* 1. Functional Requirements
  2. Performance Requirement
  3. Hardware Requirements
  4. Software Requirements
     1. Features of first software to be used
     2. Features of second software to be used

### System design 36-44

* 1. Introduction
  2. E-R Diagram
  3. Data Flow Diagram
  4. Data Dictionary

### Output Screens 45-55

1. **System Testing and Implementation 56-61**
   1. Introduction of testing
   2. Testing Strategies
   3. Implementation

### System Security 62-66

* 1. Introduction
  2. Security in Software

### Conclusion 67

1. **Future Scope 68**

**11 Bibliography 69**

1. **Brief Profile 70**

## COMPANY PROFILE

**Dremer Infotech** established by Mr. Vivek Kumar & Mr. Nitish (Director) with an idea to build an organization which adds value to student, with the mission Company have a commitment to provide high quality training in the field of Information Technology. The primary objective of the company has been to strive towards achieving total satisfaction. Dreamer infotech is one of the largest solution providers IT companies in the all region of India and a fast growing group, Secure IT Solutions is offering training & technical solution for web development & networking, AI and ML, security and data leak, IT management, and related services and gaining a global reputation by providing the best in the class facility for its customers and delighting them by serving through unmatched support and service backup. The management of the company is dedicated to deliver a "lasting value" to the student. Dreamer Infotech is consistently working towards building up the necessary expertise and skills in handling the ever-changing IT Industry.

## Introduction

### Introduction to the project

This Food Runner App helps to order your Food online and delivery at your doorsteps .

Through this App user can order their required Food and also Check health related Nutrition from their favourite restaurants.

A food delivery app that provides food delivery at your door in very less time and with the best packaging. Providing food from every famous food place near you. Order food with the best user experience.

An [Empathy](https://www.interaction-design.org/literature/topics/empathy) map will help you understand your user’s needs while you develop a deeper understanding of the persons you are designing for. An Empathy Map is just one tool that can help you empathize and synthesize your observations from the research phase, and draw out unexpected insights about your user’s needs.

He business of buying goods and services and having them delivered to your home or office has been developing for some years, and it has accelerated its expansion during times of crises and epidemics. Most families have become accustomed to receiving ready-to-eat meals delivered to their homes, and in the coming year, we will increasingly order goods at home and prepare them ourselves. It will be ineffective to waste time shopping. This article will give you a brief overview of the necessary features of [food delivery app.](https://ideausher.com/apps-development/food-delivery-application/)

This application will firstly review your location and then only you can order your food. It is because it selects near by restaurants so that the food will fresh as much as delivery agent can do.

This application also provides a login account for Exciting offers given by the App admin .Using this account a person can order food with more less price than market price .

### Purpose of the project

The main purpose of this app is too combine all the restaurants and the Local seller to the local Consumer. The main purpose of this app is too combine all the

Local food vender who have very good chef but didn’t get as much exposer so that this app can do multitasking.

By connecting all food provider to all consumer.

Food-delivery apps allow customers to order from a nearby restaurant at their convenience. The customers can get their order delivered, they can pick it up themselves or they can dine in. The restaurants receive the order on the restaurant app and prepare the meal.

Ordering process is easy. An app is fast, easy and comfortable to use. ...

Exposure to new customers.

Online ordering is convenient.

More business opportunities.

Stay ahead of the competition.

Greater reach.

Better customer data

### Problems in the existing system

At same time so many orders come

Because of this the Delivery Partner won’t deliver food on time.

Sometimes some restaurants which is highly in demand didn’t make the food on time.

On Cash on delivery there is a problem of loose money.

### Solution to these problems

Better Management System Required so that everything will go smooth.

Manage with restaurant so that they prepare food with priority.

Try to make payment online.

## Feasibility Study

**Definition :-** Feasibility study is the initial design stage of any project, which brings together the elements of knowledge that indicate if a project is possible or not.

As the name implies, a feasibility analysis is used to determine the viability of an idea, such as ensuring a project is legally and [technically feasible](https://en.wikipedia.org/wiki/Feasibility_study) as well as economically justifiable.

## Five Areas of Project Feasibility

### 2.1 Technical Feasibility

* + - This assessment focuses on the technical resources available to the organization. It helps organizations determine whether the technical resources meet capacity and whether the technical team is capable of converting the ideas into working systems. Technical feasibility also involves evaluation of the hardware, software, and other technical requirements of the proposed system. As an exaggerated example, an organization wouldn’t want to try to put Star Trek’s transporters in their building—currently, this project is not technically feasible.

### 2.2 Economic Feasibility

* + - This assessment typically involves a cost/ benefits analysis of the project, helping organizations determine the viability, cost, and benefits associated with a project before financial resources are allocated. It also serves as an independent project assessment and enhances project credibility— helping decision-makers determine the positive economic benefits to the organization that the proposed project will prove.

### 2.3 Legal Feasibility

This assessment investigates whether any aspect of the proposed project conflicts with legal requirements like zoning laws, data protection acts or social media laws. Let’s say an organization wants to construct a new office building in a specific location. A feasibility study might reveal the organization’s ideal location isn’t zoned for that type of business. That organization has just saved considerable time and effort by learning that their project was not feasible right from the beginning.

### 2.4 Operational Feasibility

This assessment involves undertaking a study to analyze and determine whether—and how well—the organization’s needs can be met by completing the project. Operational feasibility studies also examine how a project plan satisfies the requirements identified in the requirements.

### 2.5 Scheduling Feasibility

This assessment is the most important for project success; after all, a project will fail if not completed on time. In scheduling feasibility, an organization estimates how much time the project will take to complete.

* + - When these areas have all been examined, the feasibility analysis helps identify any constraints the proposed project may face, including:
    - Internal Project Constraints: Technical, Technology, Budget, Resource, etc.
    - Internal Corporate Constraints: Financial, Marketing, Export, etc.
    - **External Constraints: Logistics, Environment, Laws, and Regulations, etc.**

## 3. System Analysis

### Study of the System

The requirement analysis is done in order to understand the problem the software system is to be solved . The emphasis in requirement analysis is on identifying what is need for the system not how will achieve its goal . The goal of requirement activities is to document requirement in a software requirement specification . There are two major phase in this phase:

* + 1. Problem understood or analysis
    2. Requirement specification .

In problem analysis the aim is to understand the problem and its context , the requirement of the system to be develop . Once , problem analysis is done and requirements are understood .

Requirements must be specified in requirement specification document . The requirement document must be specifying all function and performance of requirement.

Requirements Analysis is the process of defining the expectations of the users for an application that is to be built or modified. Requirements analysis involves all the tasks that are conducted to identify the needs of different stakeholders. Therefore requirements analysis means to analyse , document, validate and manage software or system requirements.

### Proposed System

The requirements analysis process involves the following steps:

### Eliciting requirements

The process of gathering requirements by communicating with the customers is known as eliciting requirements.

### Analyzing requirements

This step helps to determine the quality of the requirements. It involves identifying whether the requirements are unclear, incomplete, ambiguous, and contradictory. These issues resolved before moving to the next step.

### Requirements modelling

In Requirements modelling, the requirements are usually documented in different formats such as use cases, user stories, natural-language documents, or process specification.

### Review and Retrospective

This step is conducted to reflect on the previous iterations of [requirements gathering](https://reqtest.com/requirements-blog/requirements-gathering-in-agile-2/) in a bid to make improvements in the process going forward.

### 3.3 Input and Output

Input is something put into a system or expended in its operation to achieve output or a result. The information entered into a computer system, examples include: typed text, mouse clicks, etc. Output is the information produced by a system or process from a specific input. Within the context of systems theory, the inputs are what are put into a system and the outputs are the results obtained after running an entire process or just a small part of a process. Because the outputs can be the results of an individual unit of a larger process, outputs of one part of a process can be the inputs to another part of the process. Output includes the visual, auditory, or tactile perceptions provided by the computer after processing the provided information. Examples include: text, images, sound, or video displayed on a monitor or through speaker as well as text or Braille from printers or embossers.

Input-output analysis is “a technique used in economics for tracing resources and products within an economy. The system of producers and consumers is divided into different branches, which are defined in terms of the resources they require as inputs and what they produce as outputs. The quantities of input and output for a given time period, usually expressed in monetary terms, are entered into an input-output matrix within which one can analyze what happens within and across various sectors of an economy where growth and decline takes place and what effects various subsidies may have

### 3.4 Process models used with justifications

The process which we used is waterfall model which is very convenient and best way to execute our requirement.

Waterfall model is an example of a Sequential model. In this model, the software development activity is divided into different phases and each phase consists of a series of tasks and has different objectives.

The ***Waterfall Model*** was the first Process Model to be introduced. It is very simple to understand and use. In a *Waterfall* model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. *The waterfall* model is the earliest [***SDLC***](http://toolsqa.com/software-testing/software-development-life-cycle/) approach that was used for software development.

In “***The Waterfall****”* approach, the whole process of software development is divided into separate phases. The outcome of one phase acts as the input for the next phase sequentially. This means that any phase in the development process begins only if the previous phase is complete. The waterfall model is a sequential design process in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of **Conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation, and Maintenance.**

**This is how the process will Executed**.

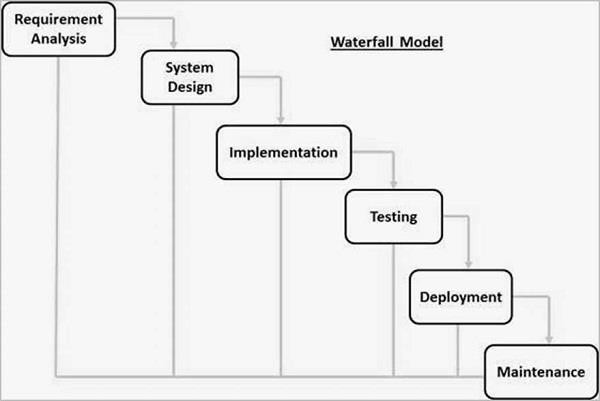


Fig 1.1. Waterfall Model

## Sequential Phases in the Waterfall Model

### Requirements:

The first phase involves understanding what needs to design and what is its function, purpose, etc. Here, the specifications of the input and output or the final product are studied and marked*.*

### System Design*:*

The requirement specifications from the first phase are studied in this phase and system

design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. The software code to be written in the next stage is created now*.*

### Implementation:

With inputs from system design, the system is first developed in small programs called units, which are integrated into the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.

### Integration and Testing:

All the units developed in the implementation phase are integrated into a system after testing of each unit. The software designed, needs to go through constant software testing to find out if there are any flaws or errors. Testing is done so that the client does not face any problem during the installation of the software.

### Maintenance:

This step occurs after installation, and involves making modifications to the system or an individual component to alter attributes or improve performance. These modifications arise either due to change requests initiated by the customer, or defects uncovered during live use of the system. The client is provided with regular maintenance and support for the developed software.

### Advantages of the Waterfall Model

* The advantage of waterfall development is that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.
* The waterfall model progresses through easily understandable and explainable phases and thus it is easy to use.
* It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
* In this model, phases are processed and completed one at a time and they do not overlap. The waterfall model works well for smaller projects where requirements are very well understood*.*

### Disadvantages of Waterfall Model

* It is difficult to estimate time and cost for each phase of the development process.
* Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought-out in the concept stage.
* Not a good model for complex and object-oriented projects.
* Not suitable for the projects where requirements are at a moderate to high risk of changing.

## Software Requirement Specifications

A **software requirements specification** (SRS) is a detailed description of a software system to be developed with its functional and non-functional requirements. The SRS is developed based the agreement between customer and contractors. It may include the use cases of how user is going to interact with software system. The software requirement specification document consistent of all necessary requirements required for project development. To develop the software system we should have clear understanding of Software system. To achieve this we need to continuous communication with customers to gather all requirements.

A good SRS defines the how Software System will interact with all internal modules, hardware, communication with other programs and human user interactions with wide range of real life scenarios. Using the *Software requirements specification* (SRS) document on QA lead, managers creates test plan. It is very important that testers must be cleared with every detail specified in this document in order to avoid faults in test cases and its expected results.

### Functional Requirements

* + 1. **Correctness of SRS should be checked.**

Since the whole testing phase is dependent on SRS, it is very important to check its correctness. There are some standards with which we can compare and verify.

### Ambiguity should be avoided.

Sometimes in SRS, some words have more than one meaning and this might confused testers making it difficult to get the exact reference. It is advisable to check for such ambiguous words and make the meaning clear for better understanding.

### Requirements should be complete.

When tester writes test cases, what exactly is required from the application, is the first thing which needs to be clear. For e.g. if application needs to send the specific data of some specific size then it should be clearly mentioned in SRS that how much data and what is the size limit to send.

### Consistent requirements

The SRS should be consistent within itself and consistent to its reference documents. If you call an input “Start and Stop” in one place, don’t call it “Start/Stop” in another. This sets the standard and should be followed throughout the testing phase.

### Verification of expected result:

SRS should not have statements like “Work as expected”, it should be clearly stated that what is expected since different testers would have different thinking aspects and may draw different results from this statement.

### Testing environment:

some applications need specific conditions to test and also a particular environment for accurate result. SRS should have clear documentation on what type of environment is needed to set up.

### Pre-conditions defined clearly:

one of the most important part of test cases is pre-conditions. If they are not met properly then actual result will always be different expected result. Verify that in SRS, all the pre- conditions are mentioned clearly.

### Security and Performance criteria:

Security is priority when a software is tested especially when it is built in such a way that it contains some crucial information when leaked can cause harm to business. Tester should check that all the security related requirements are properly defined and are clear to him. Also, when we talk about performance of a software, it plays a very important role in business so all the requirements related to performance must

be clear to the tester and he must also know when and how much stress or load testing should be done to test the performance.

### Assumption should be avoided:

Sometimes when requirement is not cleared to tester, he tends to make some assumptions related to it, which is not a right way to do testing as assumptions could go wrong and hence, test results may vary. It is better to avoid assumptions and ask clients about all the “missing requirements” to have a better understanding of expected results.

### DATA FLOW DIAGRAM SYMBOLS

Depending on the methodology (Gane and Sarson vs. Yourdon and Coad), DFD symbols vary slightly. However, the basic ideas remain the same. There are four basic elements of a data flow diagram: processes, data stores, external entities, and data flows.

### Process :

Activities and action taken on the data are represented by Circle or Round-edged rectangles.

### Symbol

**Entity**

Entities are source and destination of information data. Entities are represented by a rectangles with their respective names.

### Symbol

**Data Store**

There are two variants of data storage - it can either be represented as a rectangle with absence of both smaller sides or as an open-sided rectangle with only one side missing.

**Symbol**

### ER- DIAGRAM

The ER or (Entity Relational Model) is a high-level conceptual data model diagram. Entity-Relation model is based on the notion of real-world entities and the relationship between them.

ER modeling helps you to analyze data requirements systematically to produce a well-designed database. So, it is considered a best practice to complete ER modeling

Entity relationship diagram displays the relationships of entity set stored in a database. In other words, we can say that ER diagrams help you to explain the logical structure of databases. At first look, an ER diagram looks very similar to the flowchart. However, ER Diagram includes many specialized symbols, and its meanings make this model unique.

### History of ER models

ER diagrams are a visual tool which is helpful to represent the ER model. It was proposed by Peter Chen in 1971 to create a uniform convention which can be used for relational database and network. He aimed to use an ER model as a conceptual modeling approach.

### Facts about ER Diagram Model:

* ER model allows you to draw Database Design
* It is an easy to use graphical tool for modeling data
* Widely used in Database Design
* It is a GUI representation of the logical structure of a Database
* It helps you to identifies the entities which exist in a system and the relationships between those entities

### Why use ER Diagrams?

Here, are prime reasons for using the ER Diagram

* Helps you to define terms related to entity relationship modeling.
* Provide a preview of how all your tables should connect, what fields are going to be on each table.
* Helps to describe entities, attributes, relationships.
* ER diagrams are translatable into relational tables which allows you to build databases quickly.
* ER diagrams can be used by database designers as a blueprint for implementing data in spe- cific software applications
* The database designer gains a better understanding of the information to be contained in the database with the help of ERP diagram
* ERD is allowed you to communicate with the logical structure of the database to users

### Components of the ER Diagram

This model is based on three basic concepts:

* Entities
* Attributes
* Relationships

### ENTITY

* A real-world thing either living or non-living that is easily recognizable and non recogniza- ble. It is anything in the enterprise that is to be represented in our database. It may be a physical thing or simply a fact about the enterprise or an event that happens in the real world.
* An entity can be place, person, object, event or a concept, which stores data in the database. The characteristics of entities are must have an attribute, and a unique key. Every entity is made up of some 'attributes' which represent that entity.

### Relationship

Relationship is nothing but an association among two or more entities.

### Attributes

Entities are represented by means of their properties, called **attributes**. All attributes have values. For example, a student entity may have name, class, and age as attributes.

There exists a domain or range of values that can be assigned to attributes. For example, a student's name cannot be a numeric value. It has to be alphabetic. A student's age cannot be negative, etc.

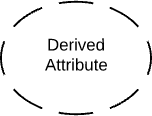
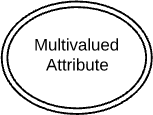
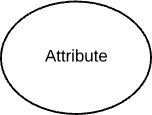
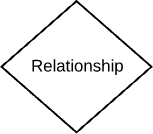
Types of Attributes

* **Simple attribute** − Simple attributes are atomic values, which cannot be divided further. For example, a student's phone number is an atomic value of 10 digits.
* **Composite attribute** − Composite attributes are made of more than one simple attribute. For example, a student's complete name may have first name and last name.
* **Derived attribute** − Derived attributes are the attributes that do not exist in the physical database, but their values are derived from other attributes present in the database. For ex- ample, average salary in a department should not be saved directly in the database, instead it can be derived. For another example, age can be derived from data of birth.
* **Single-value attribute** − Single-value attributes contain single value. For example – Social Security Number.
* **Multi-value attribute** − Multi-value attributes may contain more than one values. For exam- ple, a person can have more than one phone number, email address.

**Conceptual ER diagram symbols**

|  |  |  |
| --- | --- | --- |
| **Entity Symbol** | **Name** | **Description** |
| **ENITITY** | Strong entity | These shapes are independent from other entities, and are often called parent entities, since they will often have weak entities that depend on them. They will also have a primary key, distinguishing each occurrence of the entity. |

|  |  |  |
| --- | --- | --- |
| **WEAK**  **ENITITY** | Weak entity | Weak entities depend on some other entity type. They don't have primary keys, and have no mean- ing in the diagram without their parent entity. |
|  | Relationship | Relationships are associations between or among entities. |
|  | Weak relationship | Weak Relationships are connections between a weak entity and its owner. |
|  | Attribute | Attributes are characteristics of an entity, a many- to-many relationship, or a one-to-one relationship. |
|  | Multivalued attribute | Multivalued attributes are those that are can take on more than one value. |
|  | Derived attribute | Derived attributes are attributes whose value can be calculated from related attribute values. |



### Hardware Requirements

* Minimum 4GB memory (RAM);recommended 6GB or above.
* Minimum 4GB disk space , (500MB for IDE +4 GB for Android SDK and emu- lator system image)
* Minimum 2GHz processor,recommended 2GHz or more Intel Pentium 4 or above.
* Minimum screen resolution of 1280x800 or higher.
* Window PC with at least i3 processor

### Software Requirements

* Android Studio with SDK tools 22.0.5
* Kotlin
* Java runtime environment
* Java SE JDK v6.0
* Database
* Shared Prefrence
  + 1. **Features of first software to be used**

## FRONT END

### INTRODUCTION TO ANDROID:-

**WHAT IS ANDROID?**

Android is basically an operating system for smart phones. But we find now integrated into PADs, touch pads or televisions, even cars (trip computer) or notebooks. The OS was created by the start- up to the same name, which is owned by Google since 2005.

### Why Android is better?

* + - * **Applications**

### Google Applications

Android includes most of the time many Google applications like Gmail , YouTube or Maps .These applications are delivered with the machine most of the time , except in certain cases , such as some phones running android on which the provider has replaced Google applications by its own applications .

### Widgets

With android it is possible to use widgets which are small tools that can most often get information. These widgets are directly visible on the main window.

**Android Market**

This is an online software store to buy applications, Developers who created applications can add them into the store , and these applications can be download by users , they can be both free and paid .

### Multitasking

Android allows multitasking in the sense that multiple applications can run simultaneously. With Task Manager it is possible view all running tasks and to switch from one to another easily.

### SDK

A development kit has been put at disposal of everybody. Accordingly, any developer can create their own applications, or change the android platform. This kit contains a set of libraries, powerful tools for debugging and development, a phone emulator, through documentation

### Features of Android

1. It is open-source.
2. Anyone can customize the Android Platform.
3. There are a lot of mobile applications that can be chosen by the consumer.
4. It provides many interesting features like weather details, opening screen, live RSS (Really Simple Syndication) feeds etc.
5. It provides support for messaging services (SMS and MMS), web browser, storage (SQLite), connectivity (GSM, CDMA, Blue Tooth, Wi-Fi etc.), media, handset layout etc.

### Features :

The following features are provided in the current stable version

* + [Gradle](https://en.wikipedia.org/wiki/Gradle)-based build support
  + Android-specific [refactoring](https://en.wikipedia.org/wiki/Code_refactoring) and quick fixes
  + [Lint](https://en.wikipedia.org/wiki/Lint_(software)) tools to catch performance, usability, version compatibility and other problems
  + [ProGuard](https://en.wikipedia.org/wiki/ProGuard_(software)) integration and app-signing capabilities
  + Template-based wizards to create common Android designs and components
  + A rich [layout editor](https://en.wikipedia.org/wiki/Graphical_user_interface_builder) that allows users to drag-and-drop UI components, option to [preview lay-](https://en.wikipedia.org/wiki/WYSIWYG) [outs](https://en.wikipedia.org/wiki/WYSIWYG) on multiple screen configuration
  + Support for building [Android Wear](https://en.wikipedia.org/wiki/Android_Wear) apps
  + Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine
  + Android Virtual Device (Emulator) to run and debug apps in the Android studio.

Android Studio supports all the same programming languages of [IntelliJ](https://en.wikipedia.org/wiki/IntelliJ) (and [CLion](https://en.wikipedia.org/wiki/CLion))

e.g. [Java,](https://en.wikipedia.org/wiki/Java_(programming_language)) [C++,](https://en.wikipedia.org/wiki/C%2B%2B) and more with extensions, such as [Go](https://en.wikipedia.org/wiki/Go_(programming_language)) and Android Studio 3.0 or later

supports [Kotlin](https://en.wikipedia.org/wiki/Kotlin_(programming_language)) and "all Java 7 language features and a subset of Java 8 language features that vary by platform version." External projects [backport](https://en.wikipedia.org/wiki/Backporting) some Java 9 features. While IntelliJ states that Android Studio supports all released Java versions, and Java 12, it's not clear to what level Android Studio supports Java versions up to Java 12 (the documentation mentions partial Java 8 support). At least some new language features up to Java 12 are usable in Android.

Once an app has been compiled with Android Studio, it can be published on the [Google Play Store](https://en.wikipedia.org/wiki/Google_Play_Store). The application has to be in line with the Google Play Store [developer content policy.](https://play.google.com/about/developer-content-policy/)

The following is a list of Android Studio's major releases

|  |  |
| --- | --- |
| **Version** | **Release date** |
| 1.0 | December 2014 |
| 1.1 | February 2015 |
| 1.2 | April 2015 |
| 1.3 | July 2015 |
| 1.4 | September 2015 |
| 1.5 | November 2015 |
| 2.0 | April 2016 |
| 2.1 | April 2016 |
| 2.2 | September 2016 |
| 2.3 | March 2017 |

|  |  |
| --- | --- |
| **Version** | **Release date** |
| 3.0 | October 2017 |
| 3.1 | March 2018 |
| 3.2 | September 2018 |
| 3.3 | January 2019 |
| 3.4 | April 2019 |
| 3.5 | August 2019 |
| 3.6 | February 2020 |
| 4.0 | May 2020 |
| 4.1 | Oct 2020 |
| 4.2 | May 2021 |
| Arctic Fox (2020.3.1) | July 2021 |
| Bumblebee (2021.1.1) | January 2022 |
| Chipmunk (2021.2.1) | May 2022 |
| Dolphin (2021.3.1) | TBD |

### Features of second software to be used

**Kotlin**

Kotlin is a general purpose, free, open source, statically typed “pragmatic” programming language initially designed for the JVM (Java Virtual Machine) and Android that combines object-oriented and functional programming features. It is focused on interoperability, safety, clarity, and tooling support

Versions of Kotlin targeting JavaScript ES5.1 and native code (using LLVM) for a number of processors are in production as well.

Kotlin originated at JetBrains, the company behind IntelliJ IDEA, in 2010, and has been open source since 2012. The Kotlin team currently has more than 90 full-time members from JetBrains, and the [Kotlin project on GitHub](https://github.com/JetBrains/kotlin) has more than 300 contributors. JetBrains uses Kotlin in many of its products including its flagship IntelliJ IDEA.

Kotlin as a more concise Java language

At first glance, Kotlin looks like a more concise and streamlined version of Java. Consider the screenshot above, where I have [converted a Java code sample](https://try.kotlinlang.org/%23/Kotlin%20Koans/Introduction/Java%20to%20Kotlin%20conversion/Task.kt) (at left) to Kotlin automatically. Notice that the mindless repetition inherent in instantiating Java variables has gone away. The Java idiom

**StringBuilder** sb = **new StringBuilder**();

Becomes in Kotlin

Green IT: The color of money

val sb = **StringBuilder**()

You can see that functions are defined with the fun keyword, and that semicolons are now optional when newlines are present. The val keyword declares a read-only property or local variable. Similarly, the var keyword declares a mutable property or local variable.

Nevertheless, Kotlin is strongly typed. The val and var keywords can be used only when the type can be inferred. Otherwise you need to declare the type. Type inference seems to be improving with each release of Kotlin.

Have a look at the function declaration near the top of both panes. The return type in Java precedes the prototype, but in Kotlin it succeeds the prototype, demarcated with a colon as in Pascal.**]**

It is not completely obvious from this example, but Kotlin has relaxed Java’s requirement that functions be class members. In Kotlin, functions may be declared at top level in a file, locally inside other functions, as a member function inside a class or object, and as an extension function. Extension functions provide the C#-like ability to extend a class with new functionality without having to inherit from the class or use any type of design pattern such as Decorator.

For Groovy fans, Kotlin implements builders; in fact, Kotlin builders can be type checked. Kotlin supports delegated properties, which can be used to implement lazy properties, observable properties, vetoable properties, and mapped properties.

Many asynchronous mechanisms available in other languages can be implemented as libraries using Kotlin coroutines. This includes async/await from C# and ECMAScript, channels and select from Go, and generators/yield from C# and Python.

Functional programming in Kotlin

Allowing top-level functions is just the beginning of the functional programming story for Kotlin. The language also supports [higher-order functions, anonymous functions, lambdas, inline functions,](http://kotlinlang.org/docs/reference/lambdas.html) [closures, tail recursion, and generics.](http://kotlinlang.org/docs/reference/lambdas.html) In other words, Kotlin has all of the features and advantages of a functional language. For example, consider the following functional [Kotlin idioms](http://kotlinlang.org/docs/reference/idioms.html).

Filtering a list in Kotlin

val positives = list.filter { x -> x > 0 }

For an even shorter expression, use it when there is only a single parameter in the lambda function:

val positives = list.filter { it > 0 }

Traversing a map/list of pairs in Kotlin

**for** ((k, v) **in** map) { println(“$k -> $v”) }

k and v can be called anything. Using ranges in Kotlin

**for** (i **in** 1..100) { ... } *// closed range: includes 100*

**for** (i **in** 1 **until** 100) { ... } *// half-open range: does not include 100*

**for** (x **in** 2..10 step 2) { ... }

**for** (x **in** 10 downTo 1) { ... }

**if** (x **in** 1..10) { ... }

The above examples show the for keyword as well as the use of ranges.

Even though Kotlin is a full-fledged functional programming language, it preserves most of the object-oriented nature of Java as an alternative programming style, which is very handy when

converting existing Java code. Kotlin has [classes](http://kotlinlang.org/docs/reference/classes.html) with constructors, along with nested, inner, and anonymous inner classes, and it has [interfaces](http://kotlinlang.org/docs/reference/interfaces.html) like Java 8. Kotlin does *not* have a new keyword. To create a class instance, call the constructor just like a regular function. We saw that in the screenshot above.

Kotlin has single inheritance from a named superclass, and all Kotlin classes have a default superclass Any, which is *not* the same as the Java base class java.lang.Object. Any contains only three predefined member functions: equals(), hashCode(), and toString().

Kotlin classes have to be marked with the open keyword in order to allow other classes to inherit from them; Java classes are kind of the opposite, as they are inheritable *unless* marked with the final keyword. To override a superclass method, the method itself must be marked open, and the subclass method must be marked override. This is all of a piece with Kotlin’s philosophy of making things explicit rather than relying on defaults. In this particular case, I can see where Kotlin’s way of explicitly marking base class members as open for inheritance and derived class members as overrides avoids several kinds of common Java errors. ]

Safety features in Kotlin

Speaking of avoiding common errors, Kotlin was designed to eliminate the danger of null pointer references and streamline the handling of null values. It does this by making a null illegal for standard types, adding nullable types, and implementing shortcut notations to handle tests for null.

For example, a regular variable of type String cannot hold null:

**var** a: **String** = "abc"

a = **null** *// compilation error*

If you need to allow nulls, for example to hold SQL query results, you can declare a nullable type by appending a question mark to the type, e.g. String?.

**var** b: **String**? ="abc" b = **null** *// ok*

The protections go a little further. You can use a non-nullable type with impunity, but you have to test a nullable type for null values before using it.

To avoid the verbose grammar normally needed for null testing, Kotlin introduces a *safe call*, written ?.. For example, b?.length returns b.length if b is not null, and null otherwise. The type of this expression is Int?.

In other words, b?.length is a shortcut for if (b != null) b.length else null. This syntax chains nicely, eliminating quite a lot of prolix logic, especially when an object was populated from a series of database queries, any of which might have failed. For instance, bob?.department?.head?.name would return the name of Bob’s department head if Bob, the department, and the department head are all non-null.

To perform a certain operation only for non-null values, you can use the safe call operator ?. together with let:

val listWithNulls: **List**<**String**?> = listOf("A", **null**) **for** (item **in** listWithNulls) {

item?.**let** { println(it) } *// prints A and ignores null }*

Often you want to return a valid but special value from a nullable expression, usually so that you can save it into a non-nullable type. There’s a special syntax for this called the Elvis operator (I kid you not), written ?:.

val l = b?.length ?: -1

is the equivalent of

val l: **Int** = **if** (b != **null**) b.length **else** -1

In the same vein, Kotlin omits Java’s checked exceptions, which are throwable conditions that *must* be caught. For example, the JDK signature

**Appendable** append(**CharSequence** csq) **throws IOException**;

requires you to catch IOException every time you call an append method:

**try** { log.append(message)

}

**catch** (**IOException** e) {

*// Do something with the exception*

}

The designers of Java thought this was a good idea, and it was a net win for toy programs, as long as the programmers implemented something sensible in the catch clause. All too often in large Java programs, however, you see code in which the mandatory catch clause contains nothing but a comment: //todo: handle this. This doesn’t help anyone, and checked exceptions turned out to be a net loss for large programs.

Kotlin coroutines

[Coroutines](https://kotlinlang.org/docs/reference/coroutines/basics.html) in Kotlin are essentially lightweight threads. You start them with the launch coroutine builder in the context of some CoroutineScope. One of the most useful coroutine scopes is runBlocking{}, which applies to the scope of its code block.

**import** kotlinx.coroutines.\*

fun main() = runBlocking { *// this: CoroutineScope*

launch { *// launch a new coroutine in the scope of runBlocking* delay(1000L) *// non-blocking delay for 1 second* println("World!")

}

println("Hello,")

}

This code produces the following output, with a one-second delay between lines:

Hello, World!

### Kotlin for Android

Up until May 2017, the only officially supported programming languages for Android were Java and C++. [Google announced official support for Kotlin on Android](https://www.youtube.com/watch?v=NqlRg1_bCC4) at Google I/O 2017, and starting with Android Studio 3.0 [Kotlin is built into the Android development toolset.](https://developer.android.com/kotlin/index.html) Kotlin can be added to earlier versions of Android Studio with a plug-in.

Kotlin compiles to the same byte code as Java, interoperates with Java classes in natural ways, and shares its tooling with Java. Because there is no overhead for calling back and forth between Kotlin and Java, adding Kotlin incrementally to an Android app currently in Java makes perfect sense. The few cases where the interoperability between Kotlin and Java code lacks grace, such as Java set-only properties, are rarely encountered and easily fixed.

[Pinterest was the poster child for Android apps written in Kotlin](https://www.youtube.com/watch?v=mDpnc45WwlI) as early as November 2016, and it was mentioned prominently at Google I/O 2017 as part of the Kotlin announcement. In addition, the Kotlin team likes to cite the Evernote, Trello, Square, and Coursera apps for Android.

### Kotlin vs. Java

The question of whether to choose Kotlin or Java for new development has been coming up a lot in the Android community since the Google I/O announcement, although people were already asking the question in February 2016 when Kotlin 1.0 shipped. The short answer is that Kotlin code is safer and more concise than Java code, and that Kotlin and Java files can coexist in Android apps, so that Kotlin is not only useful for new apps, but also for expanding existing Java apps.

The only cogent argument I have seen for choosing Java over Kotlin would be for the case of complete Android development newbies. For them, there might be a barrier to surmount given that, historically, most Android documentation and examples are in Java. On the other hand, converting Java to Kotlin in Android Studio is a simple matter of pasting the Java code into a Kotlin file.

For almost anyone else doing Android development, the advantages of Kotlin are compelling. The typical time quoted for a Java developer to learn Kotlin is a few hours—a small price to pay to eliminate null reference errors, enable extension functions, support functional programming, and add coroutines. The typical rough estimate indicates approximately a 40 percent cut in the number of lines of code from Java to Kotlin.

## System design

### System Design*:*

The requirement specifications from the first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. The software code to be written in the next stage is created now*.*

### Implementation:

With inputs from system design, the system is first developed in small programs called units, which are integrated into the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.

### Integration and Testing:

All the units developed in the implementation phase are integrated into a system after testing of each unit. The software designed, needs to go through constant software testing to find out if there are any flaws or errors. Testing is done so that the client does not face any problem during the installation of the software.

### Maintenance:

This step occurs after installation, and involves making modifications to the system or an individual component to alter attributes or improve performance. These modifications arise either due to change requests initiated by the customer, or defects uncovered during live use of the system. The client is provided with regular maintenance and support for the developed software.

### E-R Diagram

The ER or (Entity Relational Model) is a high-level conceptual data model diagram. Entity-Relation model is based on the notion of real-world entities and the relationship between them.

ER modeling helps you to analyze data requirements systematically to produce a well-designed database. So, it is considered a best practice to complete ER modeling

Entity relationship diagram displays the relationships of entity set stored in a database. In other words, we can say that ER diagrams help you to explain the logical structure of databases. At first look, an ER diagram looks very similar to the flowchart. However, ER Diagram includes many specialized symbols, and its meanings make this model unique.

### History of ER models

ER diagrams are a visual tool which is helpful to represent the ER model. It was proposed by Peter Chen in 1971 to create a uniform convention which can be used for relational database and network. He aimed to use an ER model as a conceptual modeling approach.

### Facts about ER Diagram Model:

* ER model allows you to draw Database Design
* It is an easy to use graphical tool for modeling data
* Widely used in Database Design
* It is a GUI representation of the logical structure of a Database
* It helps you to identifies the entities which exist in a system and the relationships between those entities

### Why use ER Diagrams?

Here, are prime reasons for using the ER Diagram

* Helps you to define terms related to entity relationship modeling.
* Provide a preview of how all your tables should connect, what fields are going to be on each table.
* Helps to describe entities, attributes, relationships.
* ER diagrams are translatable into relational tables which allows you to build databases quickly.
* ER diagrams can be used by database designers as a blueprint for implementing data in spe- cific software applications
* The database designer gains a better understanding of the information to be contained in the database with the help of ERP diagram
* ERD is allowed you to communicate with the logical structure of the database to users

### Components of the ER Diagram

This model is based on three basic concepts:

* Entities
* Attributes
* Relationships

### ENTITY

* A real-world thing either living or non-living that is easily recognizable and non recogniza- ble. It is anything in the enterprise that is to be represented in our database. It may be a physical thing or simply a fact about the enterprise or an event that happens in the real world.
* An entity can be place, person, object, event or a concept, which stores data in the database. The characteristics of entities are must have an attribute, and a unique key. Every entity is made up of some 'attributes' which represent that entity.

### Relationship

Relationship is nothing but an association among two or more entities.

### Attributes

Entities are represented by means of their properties, called **attributes**. All attributes have values. For example, a student entity may have name, class, and age as attributes.

There exists a domain or range of values that can be assigned to attributes. For example, a student's name cannot be a numeric value. It has to be alphabetic. A student's age cannot be negative, etc.

Types of Attributes

* **Simple attribute** − Simple attributes are atomic values, which cannot be divided further. For example, a student's phone number is an atomic value of 10 digits.
* **Composite attribute** − Composite attributes are made of more than one simple attribute. For example, a student's complete name may have first name and last name.
* **Derived attribute** − Derived attributes are the attributes that do not exist in the physical database, but their values are derived from other attributes present in the database. For ex- ample, average salary in a department should not be saved directly in the database, instead it can be derived. For another example, age can be derived from data of birth.
* **Single-value attribute** − Single-value attributes contain single value. For example – Social Security Number.

**Multi-value attribute** − Multi-value attributes may contain more than one values. For example, a person can have more than one phone number, email address, etc

## E R diagram for online food ordering systemER DIAGRAM

Fig 1.2. ER Diagram

* 1. Data Flow Diagram

Data flow (flow, dataflow) **shows the transfer of information (sometimes also material) from one part of the system to another**. The symbol of the flow is the arrow. The flow should have a name that determines what information (or what material) is being moved.

* What is data flow diagram? Definition, advantages, and disadvantages – a tutorial for beginner.
* Rules and symbols for creating DFD.
* Context data flow diagram example (in PDF) with an explanation step by step.
* Level 1 data flow model diagram example (in PDF) with an explanation.
* How to draw DFD online? Best software tools and solutions
* **Processes** – the main activities that are happening within the system boundary. The process can be as simple as collecting customer data and storing it in the company database. Also, it can be a very complicated process such as creating a report containing bank contracts with customers of all bank clones in a region.
* **External entities** – the sources of information coming to or leaving the system. External entities are outside systems such as people (customers, stakeholders, managers), organizations, computers and other systems that send or receive data from our system.
* **Data stores** – places where data is held such as files or repositories. Data stores show information that is not moving.
* **Data flows** – illustrate the movements that data have between the external entities, data stores, and the processes.

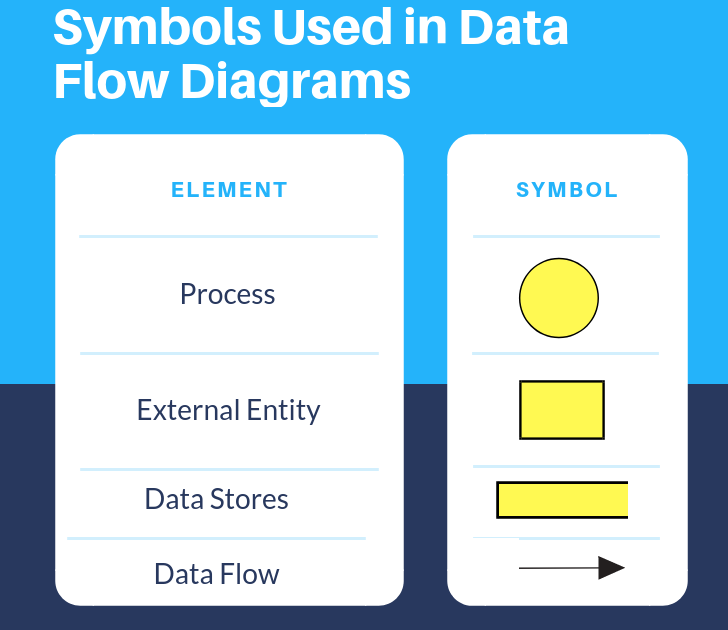


Fig 1.3. Data flow diagram

### 5.3Data Dictionary

A data dictionary in Software Engineering means a file or a set of files that includes a database's metadata (hold records about other objects in the database), like data ownership, relationships of the data to another object, and some other data

A data dictionary is a file or a set of files that includes a database's metadata. The data dictionary hold records about other objects in the database, such as data ownership, data relationships to other objects, and other data. The data dictionary is an essential component of any relational database. Ironically, because of its importance, it is invisible to most database users. Typically, only database administra- tors interact with the data dictionary.

The data dictionary, in general, includes information about the following:

* Name of the data item
* Aliases
* Description/purpose
* Related data items
* Range of values
* Data structure definition/Forms

The **name of the data item** is self-explanatory.

**Aliases** include other names by which this data item is called DEO for Data Entry Operator and DR for Deputy Registrar.

**Description/purpose** is a textual description of what the data item is used for or why it exists.

**Related data items** capture relationships between data items e.g., total\_marks must always equal to internal\_marks plus external\_marks.

**Range of values** records all possible values, e.g. total marks must be positive and between 0 to 100.

**Data structure Forms:** Data flows capture the name of processes that generate or receive the data items. If the data item is primitive, then data structure form captures the physical structures of the data item. If the data is itself a data aggregate, then data structure form capture the composition of the data items in terms of other data items.

**The mathematical operators used within the data dictionary are defined in the table:**

|  |  |
| --- | --- |
| **Notations** | **Meaning** |
| x=a+b | x includes of data elements a and b. |
| x=[a/b] | x includes of either data elements a or b. |
| x=a x | includes of optimal data elements a. |
| x=y[a] | x includes of y or more occurrences of data element a |
| x=[a]z | x includes of z or fewer occurrences of data element a |
| x=y[a]z | x includes of some occurrences of data element a which are between y and z. |

* 1. **Output Screen**

1. **Registration Activity**

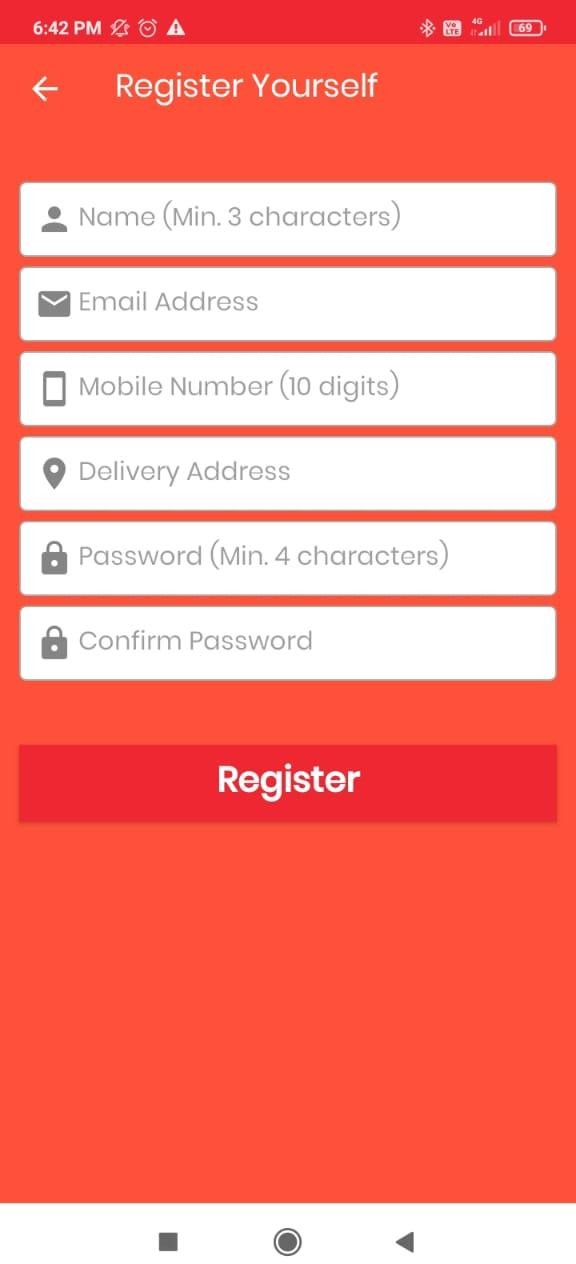
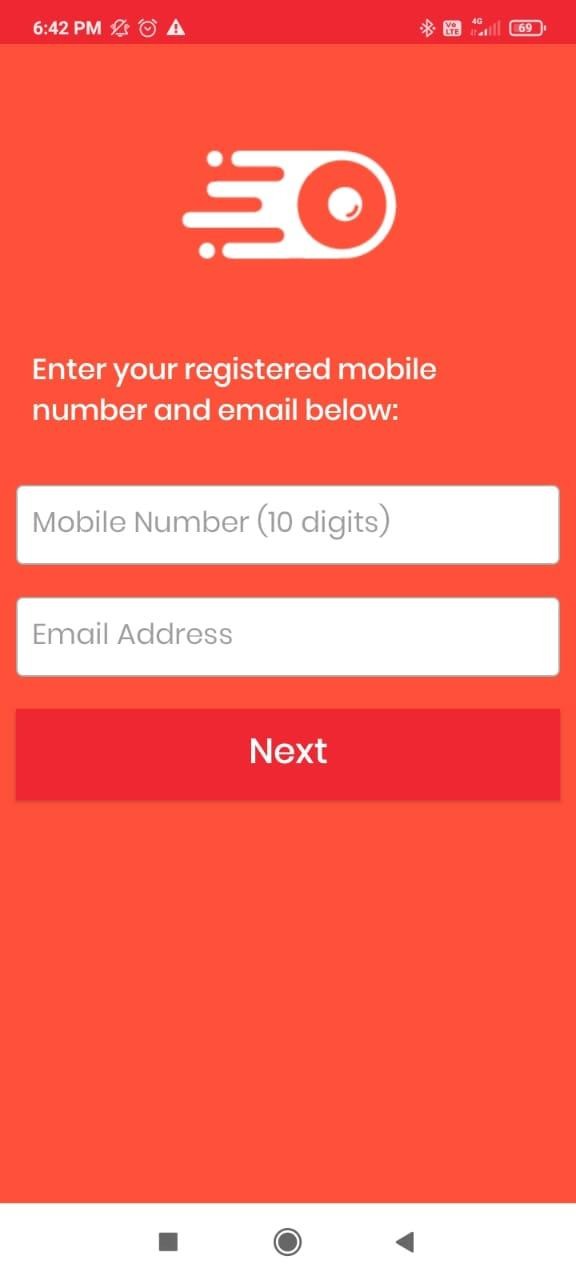
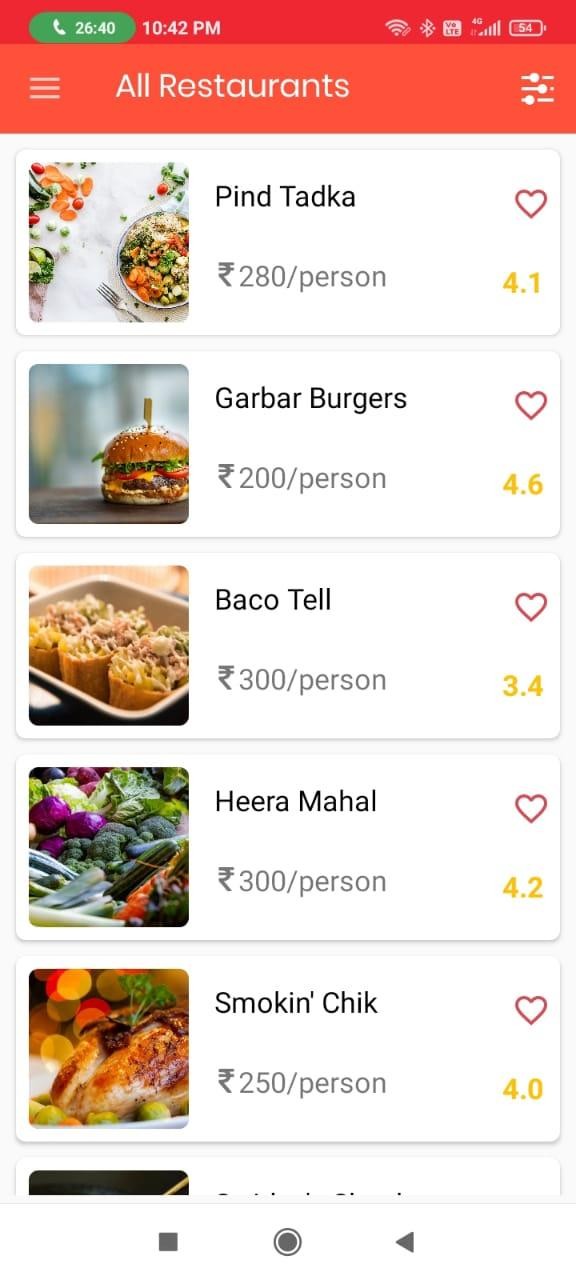


Fig 2.1. Registered Activity

Fig 2.2. Login Activity

Fig 2.3. Restaurant Activity

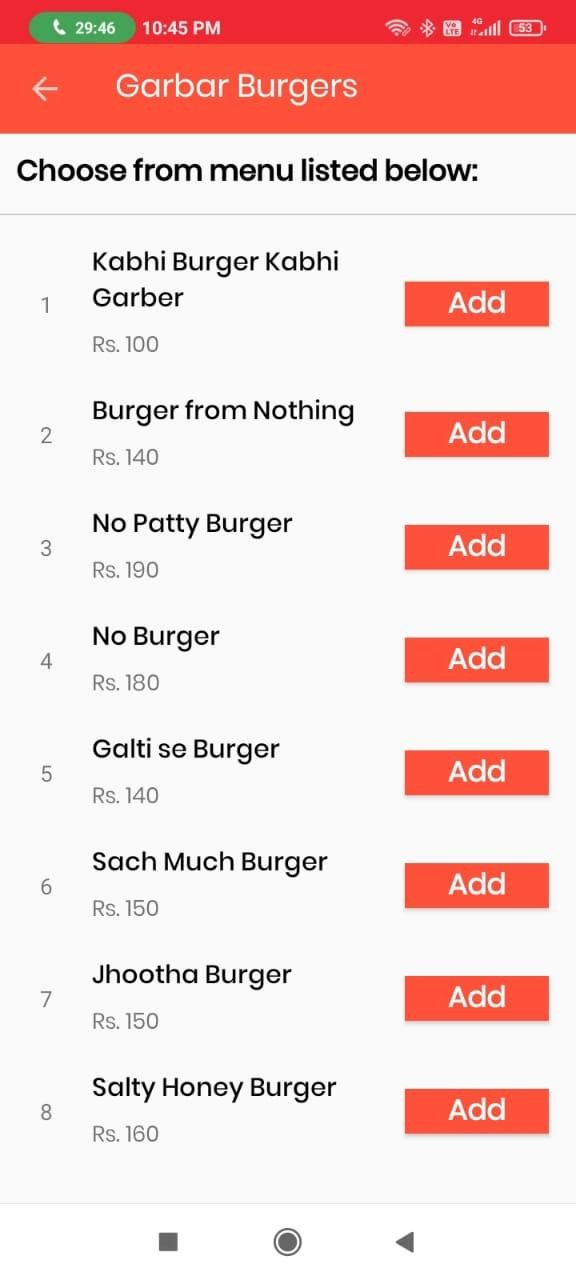


Fig 2.4. Menu Activity

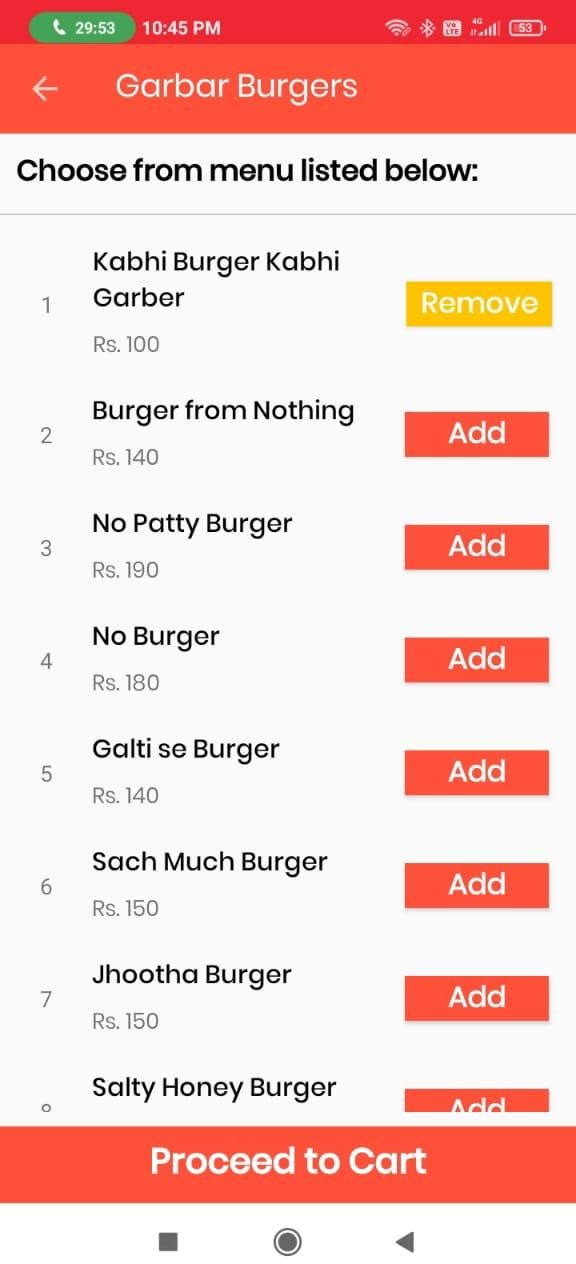


Fig 2.5.Choose menu Activity



Fig 2.6. My Cart Activity



Fig 2.7. Order place Activity

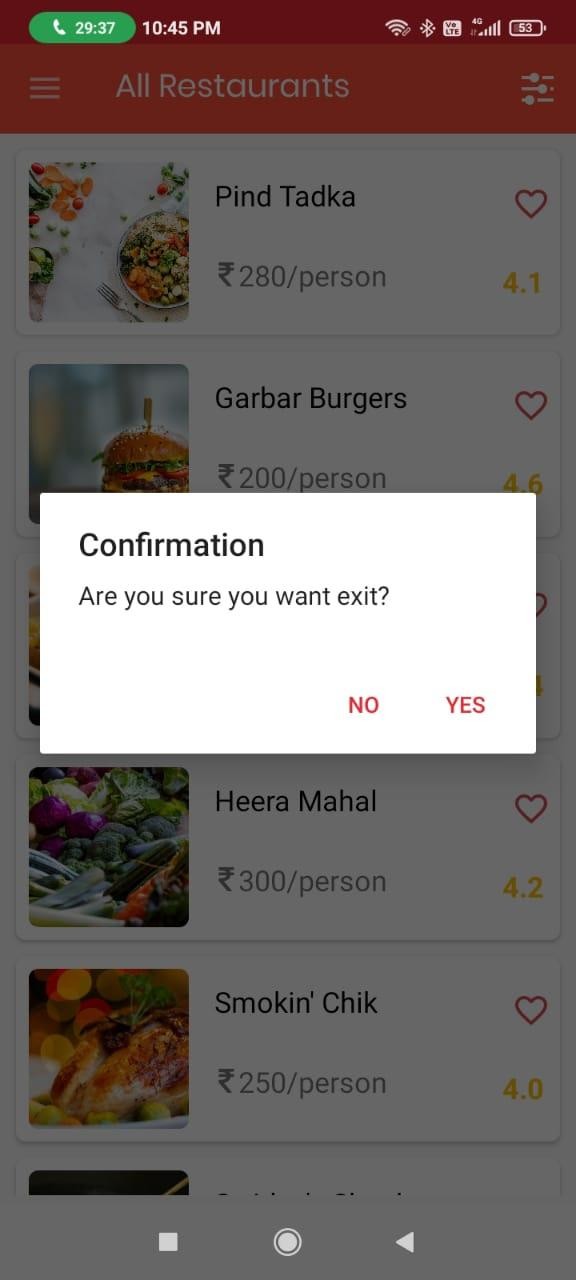


Fig 2.8. Exit dialog box



Fig 2.9. Splash Activity

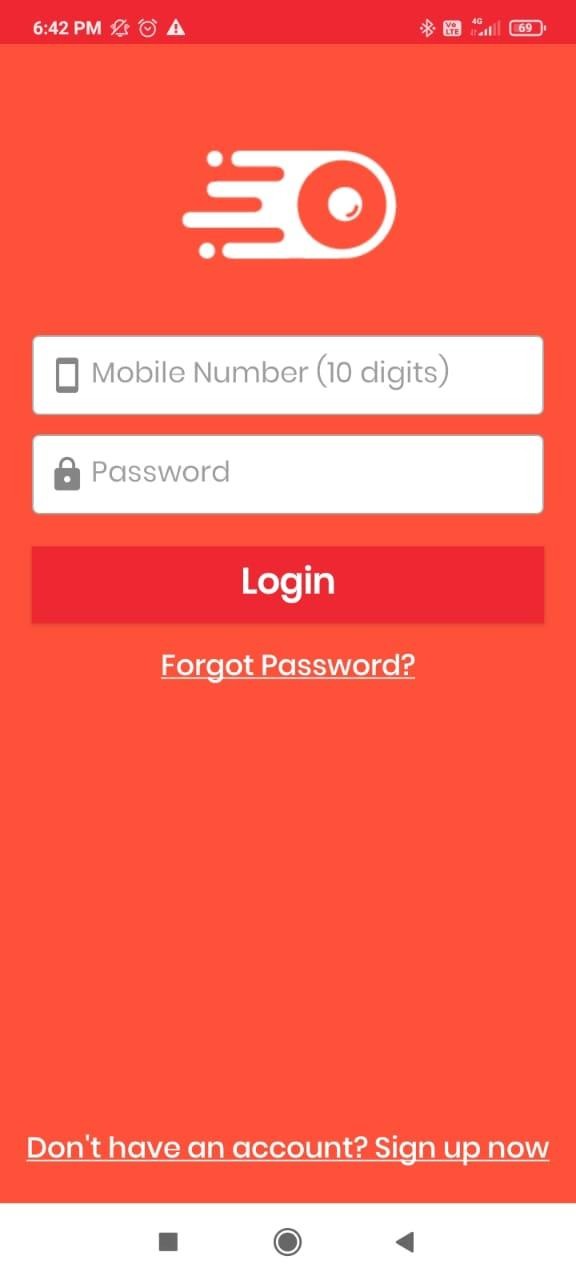


Fig 3.1. Login Activity

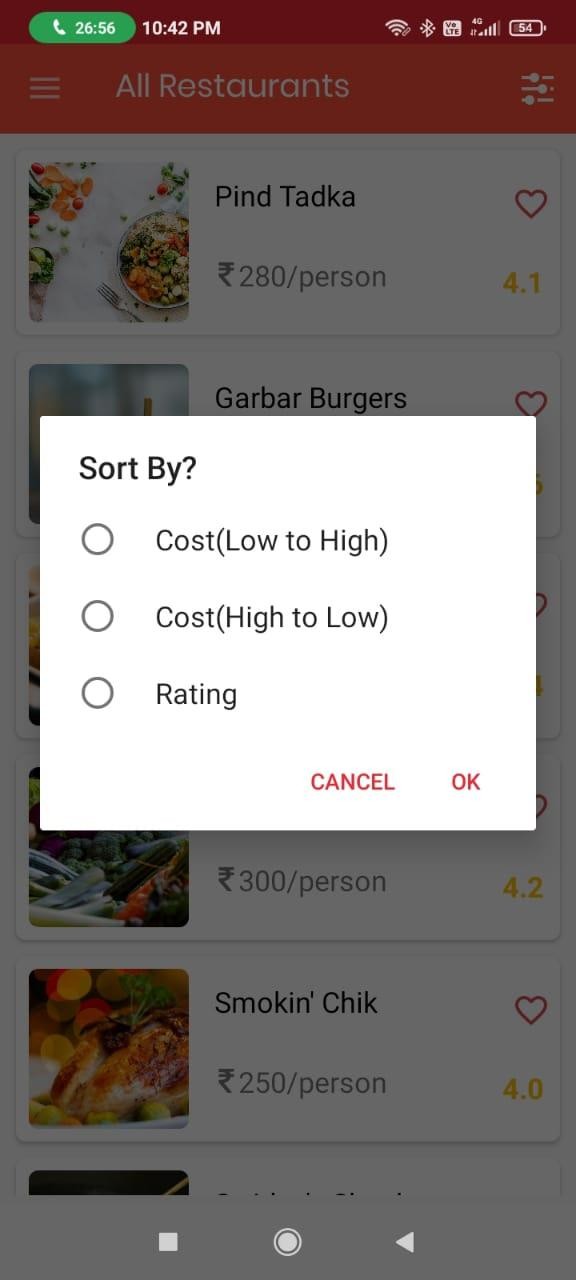


Fig 3.2. Sort dialog box

## System Testing and Implementation

### Introduction of testing

**Software Testing** is a method to check whether the actual software product matches expected requirements and to ensure that software product is [Defect](https://www.guru99.com/defect-management-process.html) free. It involves execution of software/system components using manual or automated tools to evaluate one or more properties of interest. The purpose of software testing is to identify errors, gaps or missing requirements in contrast to actual requirements.

Some prefer saying Software testing definition as a [White Box](https://www.guru99.com/white-box-testing.html) and [Black Box Testing.](https://www.guru99.com/black-box-testing.html) In simple terms, Software Testing means the Verification of Application Under Test (AUT). This Software Testing course introduces testing software to the audience and justifies the importance of software testing.

In this Software Testing tutorial, you will learn basics of software testing like:

* + - [What is Software Testing?](https://www.guru99.com/software-testing-introduction-importance.html#1)
    - [Why is Software Testing Important?](https://www.guru99.com/software-testing-introduction-importance.html#2)
    - [What are the benefits of Software Testing?](https://www.guru99.com/software-testing-introduction-importance.html#3)
    - [Testing in Software Engineering](https://www.guru99.com/software-testing-introduction-importance.html#4)
    - [Types of Software Testing](https://www.guru99.com/software-testing-introduction-importance.html#5)
    - [Testing Strategies in Software Engineering](https://www.guru99.com/software-testing-introduction-importance.html#6)
    - [Program Testing](https://www.guru99.com/software-testing-introduction-importance.html#7)

**Why Software Testing is Important?**

Software Testing is Important because if there are any bugs or errors in the software, it can be identified early and can be solved before delivery of the software product. Properly tested software product ensures reliability, security and high performance which further results in time saving, cost effectiveness and customer satisfaction.

**What is the need of Testing?**

Testing is important because software bugs could be expensive or even dangerous. Software bugs can potentially cause monetary and human loss, and history is full of such examples.

* + - In April 2015, Bloomberg terminal in London crashed due to software glitch affected more than 300,000 traders on financial markets. It forced the government to postpone a 3bn pound debt sale.
    - Nissan cars recalled over 1 million cars from the market due to software failure in the airbag sensory detectors. There has been reported two accident due to this software failure.
    - Starbucks was forced to close about 60 percent of stores in the U.S and Canada due to soft- ware failure in its POS system. At one point, the store served coffee for free as they were un- able to process the transaction.
    - Some of Amazon’s third-party retailers saw their product price is reduced to 1p due to a software glitch. They were left with heavy losses.
    - Vulnerability in Windows 10. This bug enables users to escape from security sandboxes through a flaw in the win32k system.
    - In 2015 fighter plane F-35 fell victim to a software bug, making it unable to detect targets correctly.
    - China Airlines Airbus A300 crashed due to a software bug on April 26, 1994, killing 264 in- nocents live
    - In 1985, Canada’s Therac-25 radiation therapy machine malfunctioned due to software bug and delivered lethal radiation doses to patients, leaving 3 people dead and critically injuring 3 others.
    - In April of 1999, a software bug caused the failure of a $1.2 billion military satellite launch, the costliest accident in history
    - In May of 1996, a software bug caused the bank accounts of 823 customers of a major U.S. bank to be credited with 920 million US dollars.

Here are the benefits of using software testing:

* + - **Cost-Effective:** It is one of the important advantages of software testing. Testing any IT project on time helps you to save your money for the long term. In case if the bugs caught in the earlier stage of software testing, it costs less to fix.
    - **Security:** It is the most vulnerable and sensitive benefit of software testing. People are look- ing for trusted products. It helps in removing risks and problems earlier.
    - **Product quality:** It is an essential requirement of any software product. Testing ensures a quality product is delivered to customers.
    - **Customer Satisfaction:** The main aim of any product is to give satisfaction to their custom- ers. UI/UX Testing ensures the best user experience.

### Program Testing

**Program Testing** in software testing is a method of executing an actual software program with the aim of testing program behavior and finding errors. The software program is executed with test case data to analyse the program behavior or response to the test data. A good program testing is one which has high chances of finding bugs.

### Summary

* + - Define Software Testing: Software testing is defined as an activity to check whether the ac- tual results match the expected results and to ensure that the software system is Defect free.
    - Testing is important because software bugs could be expensive or even dangerous.
    - The important reasons for using software testing are: cost-effective, security, product qual- ity, and customer satisfaction.
    - Typically Testing is classified into three categories [functional testing,](https://www.guru99.com/functional-testing.html) non-functional testing or performance testing, and maintenance.
    - The important strategies in software engineering are: unit testing, integration testing, valida- tion testing, and system testing.

### Implementation

**Test implementation** is the practice of **organizing and** [**prioritizing tests**.](http://tryqa.com/how-to-perform-test-prioritization-effort-allocation-in-test-process/) This is carried out by Test Analysts who implement the test designs as real test cases, test processes and test data.

* + - Test inputs
    - Expected results for each [**test case**](http://tryqa.com/test-case/)
    - Steps to be followed for each [**test process**](http://tryqa.com/what-is-fundamental-test-process-in-software-testing/)

These three are documented together and test data is stored in the form of database tables, flat files, etc.

Ensuring that the team is prepared for executing the [**test design**](http://tryqa.com/what-is-test-design-when-to-create-it/)is an important part of test implementation process.

Some of the **checks that could be performed** to confirm that the **team ready to execute tests** include:

* + - Ensuring that the test environment is in place
    - Ensuring every test case is well documented and reviewed
    - Putting test environment in a state of readiness
    - Checking against explicit and implicit entry criteria for the specified test level
    - Describing test environment as well as test data in great detail
    - Performing code acceptance check by running it on test environment

Granularity and related complexity of tasks taken up in the course of test implementation is often influenced by granularity **of** [test work products](http://tryqa.com/what-is-test-planning-what-are-work-products-in-testing/) like test cases, test conditions etc.

For example, if the tests are to be documented for using again in future for regression testing, the test documents will record step by step description of executing the test.

This detailed explanation will enable other [software testers](http://tryqa.com/software-tester/) to conduct the test reliably and consistently irrespective of their expertise.

Similarly, if regulatory guidelines are applicable to the test procedure, test compliance to relevant standards must be documented as evidence.

Test Managers are also responsible **for** creating a schedule for [test execution](http://tryqa.com/what-is-test-execution/), detailing the **order for execution** of both automated and manual tests.

They should diligently check for constraints like risks, priorities etc. that may necessitate the tests to be executed in a specific order or using a specific equipment. Test Managers should also **check for dependencies** on test data or test environment.

### Disadvantages of early test implementation

Implementing the **tests** early may have some disadvantages too.

* + - For example, if [**Agile lifecycle**](http://tryqa.com/what-is-agile-methodology-examples-when-to-use-it-advantages-and-disadvantages/) has been adopted for product development, the code itself may undergo drastic changes between consecutive iterations. This will render the whole test implementation useless.
    - In fact, any [**iterative development lifecycle**](http://tryqa.com/what-is-iterative-model-advantages-disadvantages-and-when-to-use-it/) will affect the code between iterations, even if it is not as drastic as that in the Agile lifecycle.
    - This will make pre-defined tests obsolete or require continuous and resource intensive maintenance.
    - Similarly, even in case of sequential lifecycles, if the project is badly managed and require- ments keep changing even when project is in an advanced state, early test implementation can be rendered obsolete.

Therefore**,** before starting the test implementation process**,** Test Manager should consider these important points:

* + - [Software development life cycl**e**](http://tryqa.com/what-are-the-software-development-life-cycle-sdlc-phases/) being used
    - Features that need to be tested
    - Probability of change in requirement late into project lifecycle
    - Possibility of changes in code between two iterations

### Advantages of early test implementation

Early test implementation offers some advantages too.

* + - Concrete tests, for example, deliver ready examples of appropriate behavior of the software if documented according to the test conditions.
    - Domain experts find it easier to verify the concrete tests rather than non-concrete business rules, which further enables them to detect faults in software specifications.

## System Security

### Introduction

The security of a computer system is a crucial task. It is a process of ensuring the confidentiality.

A system is said to be secure if its resources are used and accessed as intended under all the circumstances, but no system can guarantee absolute security from several of various malicious threats and unauthorized access.

The security of a system can be threatened via two violations:

* + **Threat:** A program that has the potential to cause serious damage to the system.
  + **Attack:** An attempt to break security and make unauthorized use of an asset.

Security violations affecting the system can be categorized as malicious and accidental. **Malicious threats**, as the name suggests are a kind of harmful computer code or web script designed to create system vulnerabilities leading to back doors and security breaches. **Accidental Threats**, on the other hand, are comparatively easier to be protected against. Example: [Denial of Service (DDoS) attack](https://www.geeksforgeeks.org/computer-network-denial-of-service-ddos-attack/).

Security can be compromised via any of the breaches mentioned:

* + **Breach of confidentiality:** This type of violation involves the unauthorized reading of data.
  + **Breach of integrity:** This violation involves unauthorized modification of data.
  + **Breach of availability:** It involves unauthorized destruction of data.
  + **Theft of service:** It involves the unauthorized use of resources.
  + **Denial of service:** It involves preventing legitimate use of the system. As mentioned before, such attacks can be accidental in nature.

Henceforth, based on the above breaches, the following security goals are aimed:

1. **Integrity:**

The objects in the system mustn’t be accessed by any unauthorized user & any user not having sufficient rights should not be allowed to modify the important system files and resources.

#### Secrecy:

The objects of the system must be accessible only to a limited number of authorized users. Not everyone should be able to view the system files.

#### Trap Door:

The designer of a program or system might leave a hole in the software that only he is

capable of using, the Trap Door works on similar principles. Trap Doors are quite diffi- cult to detect as to analyze them, one needs to go through the source code of all the components of the system.

1. **Logic Bomb:** A program that initiates a security attack only under a specific situation.
2. **Worm:** A computer worm is a type of malware that replicates itself and infects other computers while remaining active on affected systems. A computer worm replicates it- self in order to infect machines that aren’t already infested. It frequently accomplishes this by taking advantage of components of an operating system that are automatic and unnoticed by the user. Worms are frequently overlooked until their uncontrolled repli- cation depletes system resources, slowing or stopping other activity

Aside from the program threats, various system threats are also endangering the secu- rity of our system:

1. **Worm:**An infection program that spreads through networks. Unlike a virus, they target mainly LANs. A computer affected by a worm attacks the target system and writes a small program “hook” on it. This hook is further used to copy the worm to the target computer. This process repeats recursively, and soon enough all the systems of the LAN are affected. It uses the spawn mechanism to duplicate itself. The worm spawns copies of itself, using up a majority of system resources and also locking out all other processes.

The basic functionality of the worm can be represented as:

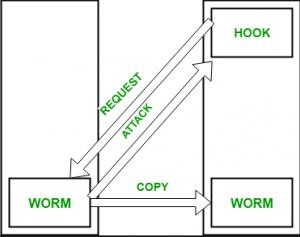


Fig 4.1. Worm

1. **Port Scanning:** It is a means by which the cracker identifies the vulnerabilities of the system to attack. It is an automated process that involves creating a TCP/IP connection to a specific port. To protect the identity of the attacker, port scanning attacks are launched from **Zombie Systems**, that is systems that were previously

independent systems that are also serving their owners while being used for such notorious purposes.

1. **Denial of Service:** Such attacks aren’t aimed for the purpose of collecting information or destroying system files. Rather, they are used for disrupting the legitimate use of a system or facility. These attacks are generally network-based. They fall into two categories:
2. Attacks in this first category use so many system resources that no useful work can be performed.

For example, downloading a file from a website that proceeds to use all available CPU time.

– Attacks in the second category involves disrupting the network of the facility. These attacks are a result of the abuse of some fundamental TCP/IP principles. the fundamental functionality of TCP/IP.

**Security Measures Taken –** To protect the system, Security measures can be taken at the following levels:

#### Physical:

The sites containing computer systems must be physically secured against armed and malicious intruders. The workstations must be carefully protected.

* + **Human:** Only appropriate users must have the authorization to access the system. Phishing(collecting confidential information) and Dumpster Diving(collecting basic in- formation so as to gain unauthorized access) must be avoided.
  + **Operating system:** The system must protect itself from accidental or purposeful secu- rity breaches.
  + **Networking System:** Almost all of the information is shared between different sys- tems via a network. Intercepting these data could be just as harmful as breaking into a computer. Henceforth, Network should be properly secured against such attacks.

Usually, Anti Malware programs are used to periodically detect and remove such viruses and threats. Additionally, to protect the system from Network Threats, a [Firewall](https://www.geeksforgeeks.org/computer-network-firewall-methodologies/) is also be used.

### Security in Software

Software security is an idea implemented to protect software against malicious attack and other hacker risks so that the software continues to function correctly under such potential risks. Security is necessary to provide integrity, authentication and availability.

Any compromise to integrity, authentication and availability makes a software unsecure. Software systems can be attacked to steal information, monitor content, introduce vulnerabilities and damage the behavior of software. Malware can cause DoS (denial of service) or crash the system itself.

Buffer overflow, stack overflow, command injection and SQL injections are the most common attacks

Buffer and stack overflow attacks overwrite the contents of the heap or stack respectively by writing bytes.

Command injection can be achieved on the software code when system commands are used predominantly. New system commands are appended to existing commands by the malicious attack. Sometimes system command may stop services and cause DoS.

SQL injections use malicious SQL code to retrieve or modify important information from database servers. SQL injections can be used to bypass login credentials. Sometimes SQL injections fetch important information from a database or delete all important data from a database.

The only way to avoid such attacks is to practice good programming techniques. System-level security can be provided using better firewalls. Using intrusion detection and prevention can also aid in stopping attackers from easy access to the system.

## Conclusion

Conclusion of the Project Online Food Ordering System:-Our project is only a humble venture to satisfy the needs to manage their project work. Several user friendly coding have also adopted. This package shall prove to be a powerful.

Package in satisfying all the requirements of the school. The objective of software planning is to provide a frame work that enables the manger to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses. At the end it is concluded that we have made effort on following points.

* A description of the background and context of the project and its relation to work already done in the area.
* Made statement of the aims and objectives of the project
* The description of Purpose. Scope, and applicability.
* We define the problem on which we are working in the project
* We describe the requirement Specifications of the system and the actions that can be done on these things.
* We understand the problem domain and produce a model of the system, which describes operations that can be performed on the system
* We included features and operations in detail, including screen layouts.
* We designed user interface and security issues related to system. Finally the system is implemented and tested according to test cases

## Future Scope

The online food ordering system is one of the most profitable marketing strategies for restaurant businesses. Online food ordering platforms also prevents missed orders due to busy phone lines or a lack of resources to monitor the phone. The face of the restaurant industry has shifted from the traditional dine-in culture to takeaways, online ordering, and home deliveries. Restaurants are quickly incorporating mobile food ordering apps in their restaurant management systems to streamline the entire order taking process.

Online food ordering platforms have disrupted the food ordering and delivery process in restaurants. People no more have to shout for their orders in a crowdy restaurant, or especially go out to eat at a restaurant. Furthermore, the online food ordering system for restaurants is a trend that is growing at a fast pace. Major Restaurants today just for the convenience factor and increasing revenue are listing their services even online.

## Bibliography

* + [www.quore.com](http://www.quore.com/)
  + Internshala
  + [www.wikipedia.com](http://www.wikipedia.com/)
  + [www.tutorials.com](http://www.tutorials.com/)
  + [www.javatpoints.com](http://www.javatpoints.com/)

## BRIEF PROFILE

### PERSONAL DETAILS:

* + - Full Name: ADITYA KUMAR
    - Email: [adityadee8@gmail.com](mailto:adityadee8@gmail.com)
    - LinkedIn Profile: https://www.linkedin.com/in/aditya-kumar-b2387a122

### ACADEMIC QUALIFICATIONS:

* + - **B.Tech. – Computer Science and Engineering**

Aravali College of Engineering and Management (2018-2022) : *CGPA – 7.02*

### Senior Secondary Education (12th) B.S.E.B.

A.B.S. COLLEGE (2017) : *Percentage – 58.8%*

### Secondary Education (10th ) B.S.E.B.

R.B.P.HIGH SCHOOL (2015) : *Percentage – 60%*

### TECHNICAL PROFICIENCIES:

* + - **Languages:** C, Python, Kotlin, HTML, CSS
    - **IT Constructs:** OOPS, OS and Firebase
    - **Development tools**: Android Studio, VS Code
    - **Build Management:** Git

### PROFESSIONAL EXPERIENCE:

* + - **Android Developer Intern, Dreamer Infotech (feb 2022 – June-2022):**
      * Worked as a part of Android Team to develop Mobile app. Modules Undertaken are: Android studio and Kotlin.
      * E-Commerce App: including all the aspects like Basic UI Design to Retroffit API Integration checkout, payment methods, customer success, etc. Also contributed in Integration of Payment Gateways and APIs.