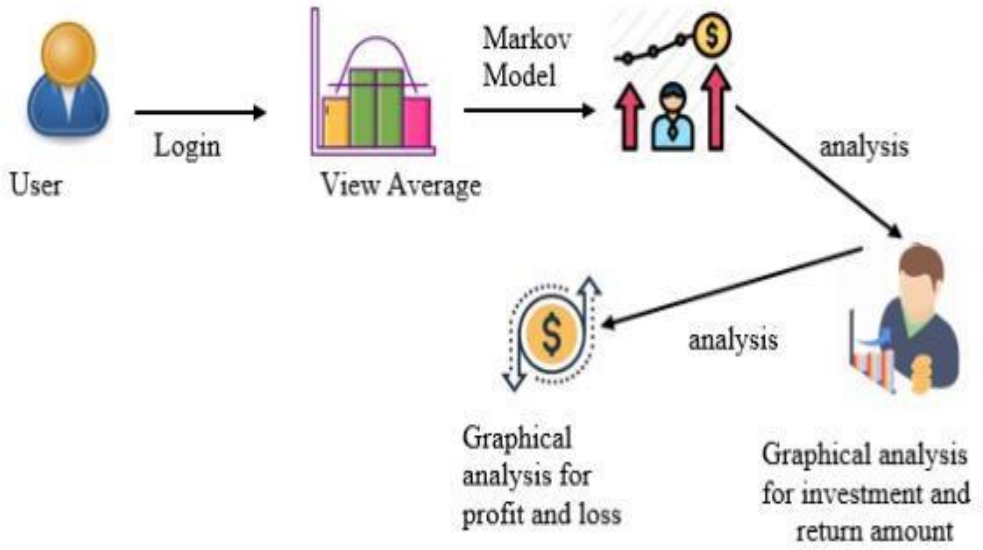
HOW DATA DRIVEN ENTEREPRENEUR ANALYSIS IMPERFECT INFORMATION FOR BUSINESS OPPORTUNITY EVALUATION

PHASE 2 – THE DESIGN PHASE

CONTENTS

1.0 Architecture

An Architecture Diagram is a graphical representation of a set of concepts, that are part of an architecture, including their principles, elements and components. It is a coherent set of concepts for a structure.



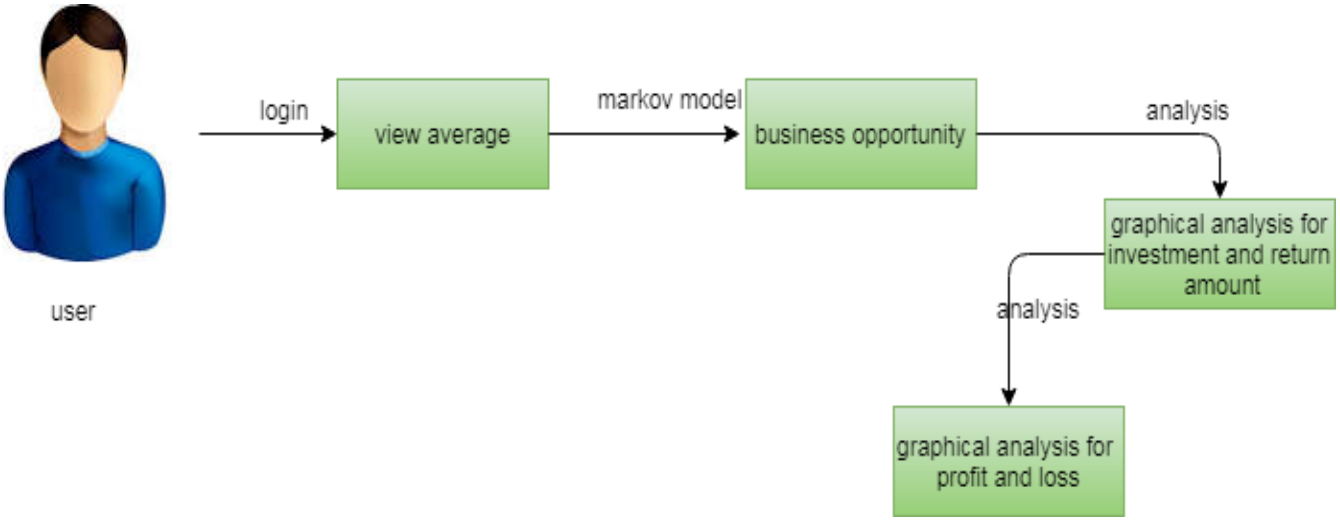
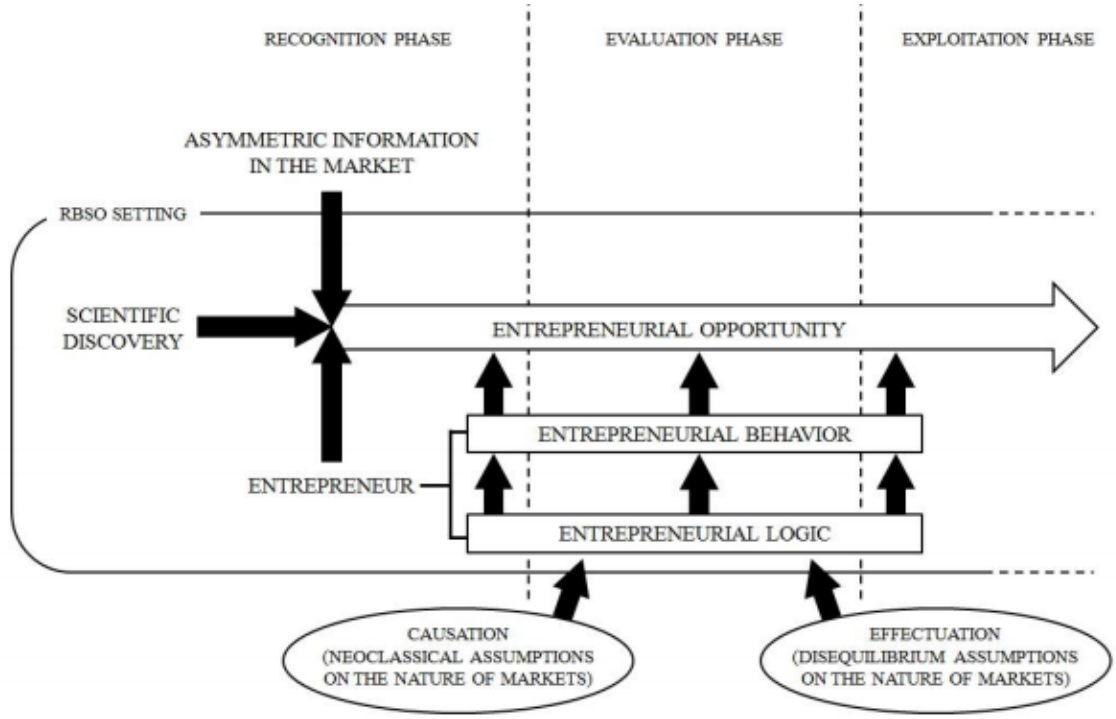


Figure 1.0 Entrepreneur analysis work flow diagram



MODULE DESCRIPTION:

The modules are implemented as given in the following ways

- **Average Analysis**

The first step for over all data set analyzing for average. This data set contains for Investment details, Return amount details, Profit details, and another one is Loss details. They will be finding as Investment average, return amount average, profit average, and another one is loss average. This average analysis for very useful in perfect decision making in business opportunity evaluations.

- **Business opportunity**

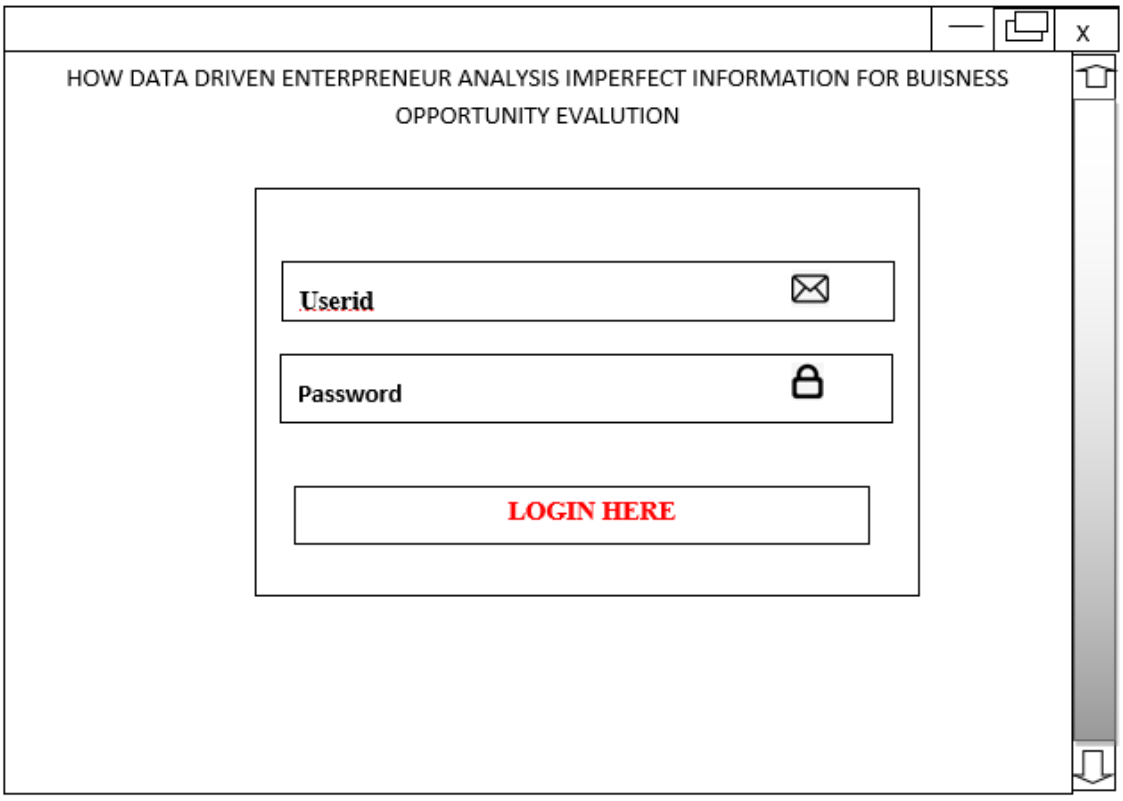
One of the next process in business opportunity. This evaluation in calculated and analyzing for best way used in one of popular machine learning algorithms. In this machine learning algorithms for Markov Chain Model algorithm. This algorithm is explained as one

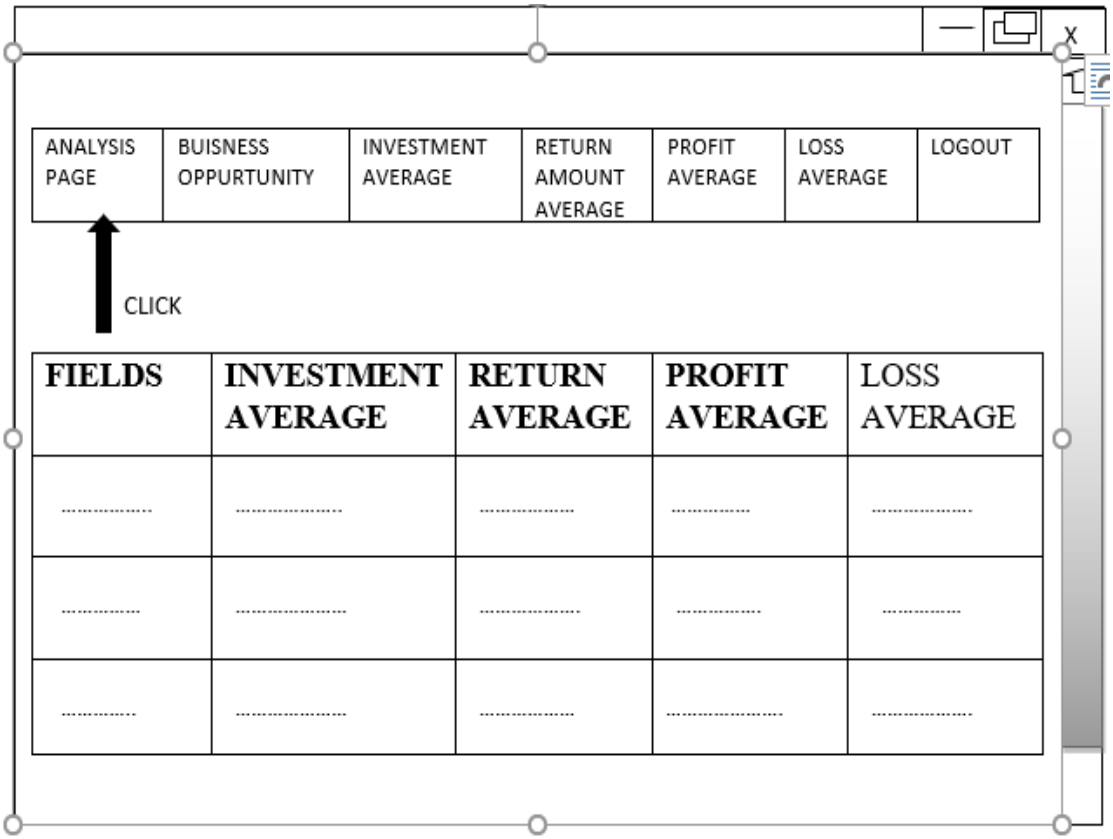
process is defends as another one previses process. They will be find out the profit and return amount is concede as the main process in profit values in defines is one main process is the business opportunity.

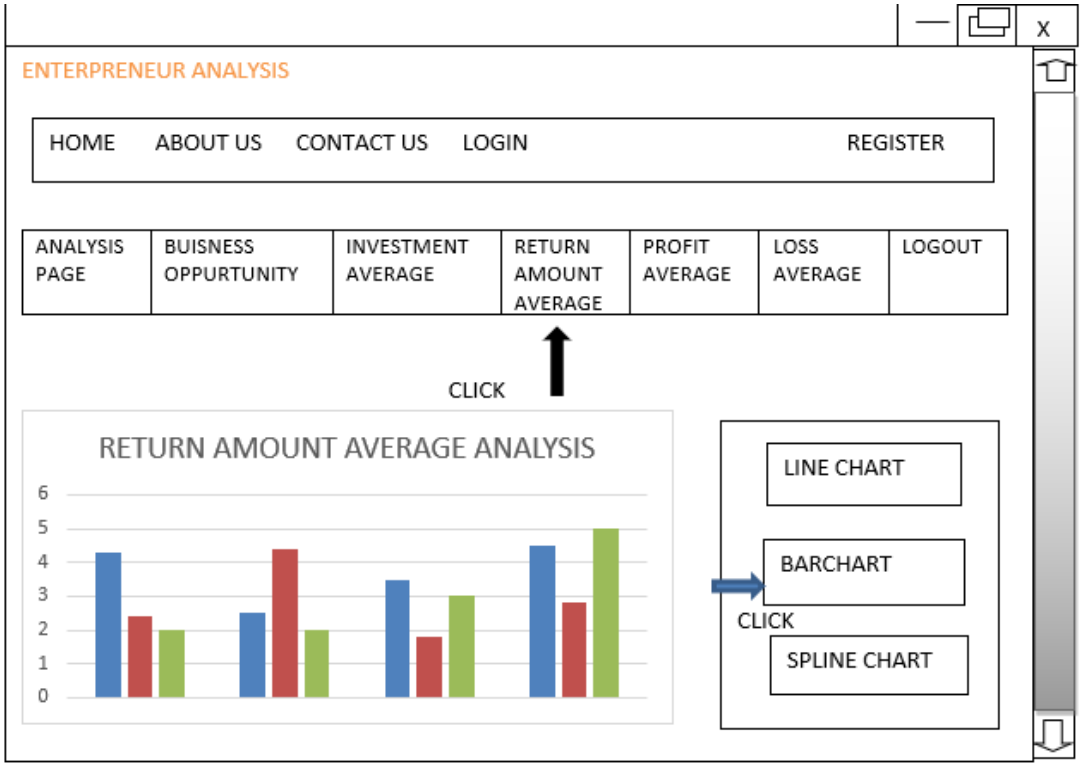
- **Graphical Analysis**

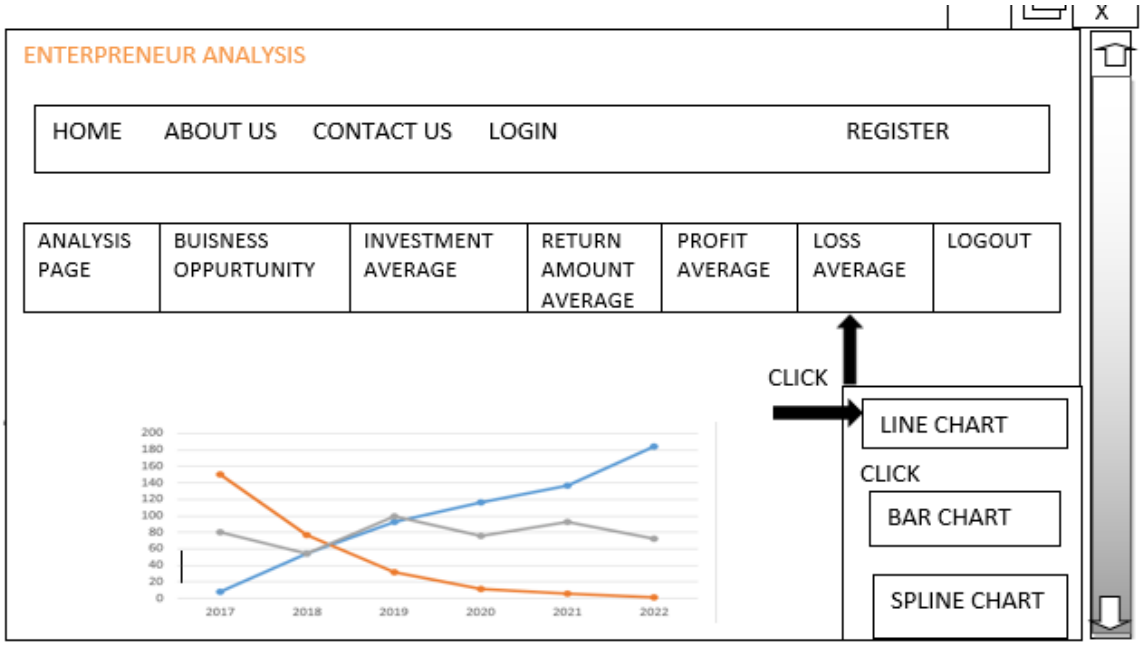
User Find out the evaluation process is one by one .This graphical process mainly used in easy way to analyzing and understanding the business opportunity .This sections is explains as investment average graphical analysis ,next one is calculates as return amount is calculated as and graphical statement ,another one is profit average analyzing in the process, and final is main process is calculated as the loss average calculated and another is completed process.

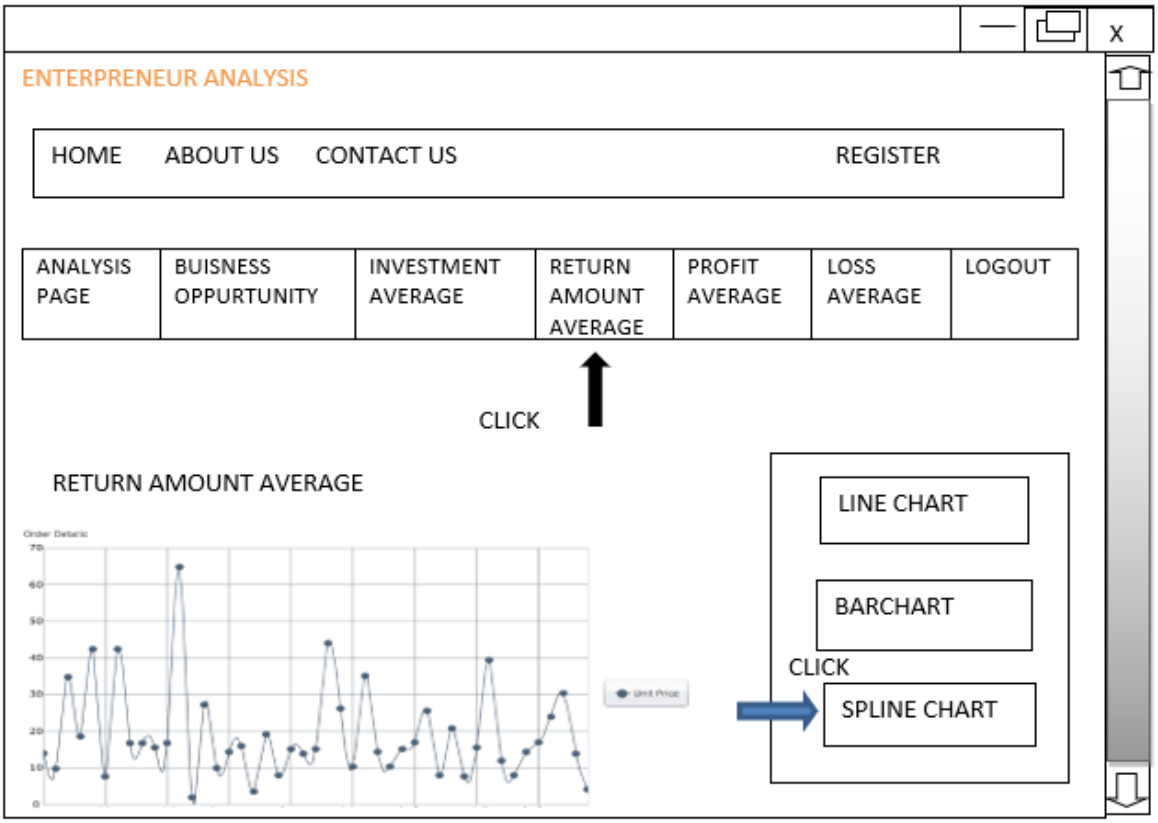
2.0 SCREEN LAYOUT

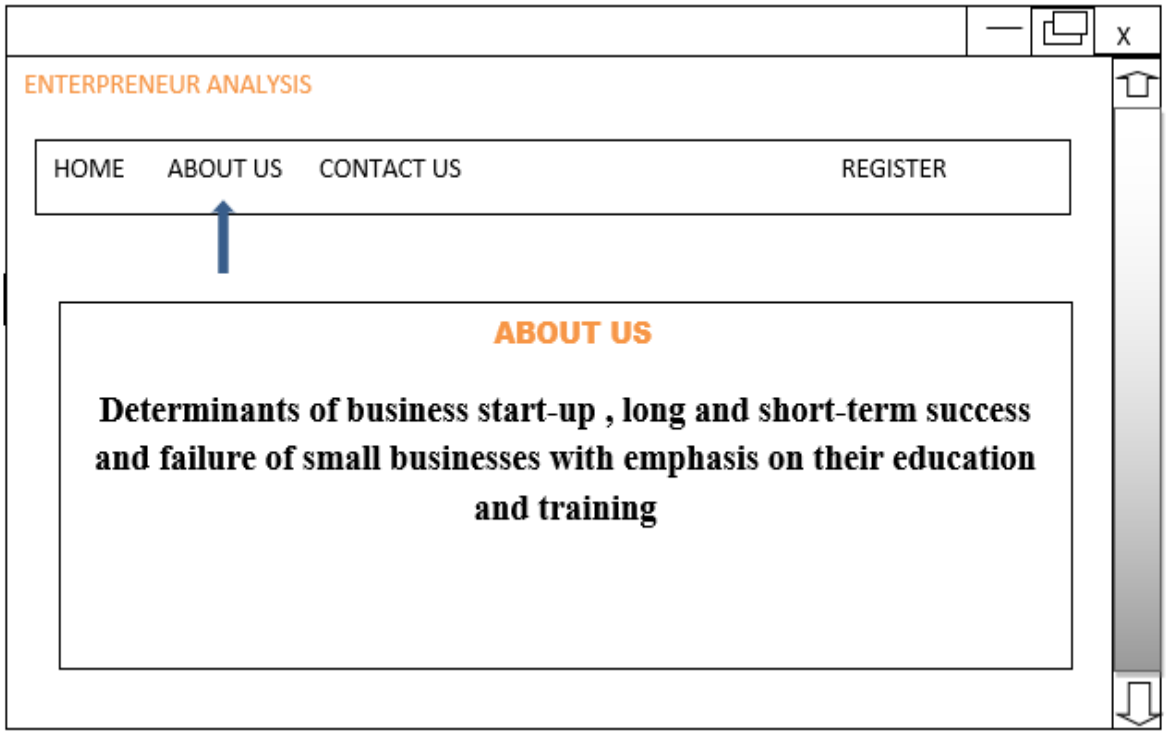


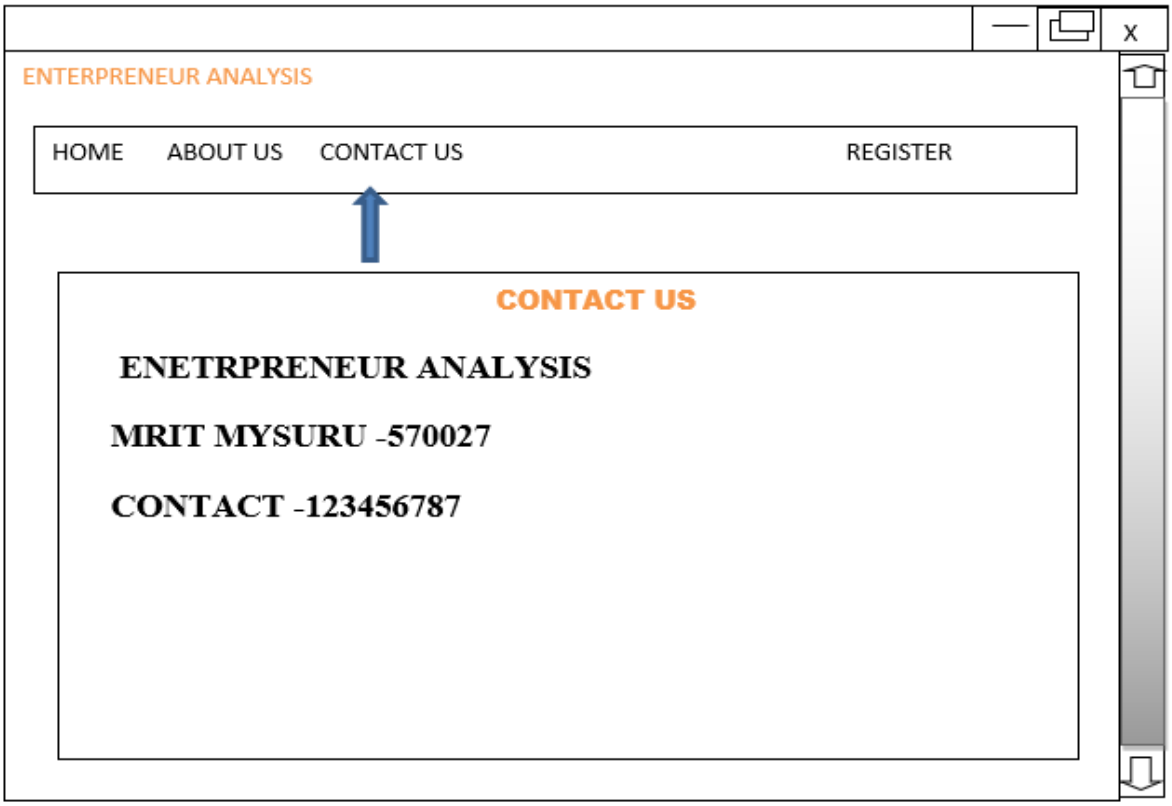

ANALYSIS PAGE
BUISNESS OPPURTUNITY
INVESTMENT AVERAGE
RETURN AMOUNT AVERAGE
PROFIT AVERAGE
LOSS AVERAGE
LOGOUT
CLICK
FIELDS
INVESTMENT AVERAGE
RETURN AVERAGE
PROFIT AVERAGE
LOSS AVERAGE


ENTERPRENEUR ANALYSIS
HOME
ABOUT US
CONTACT US
LOGIN
REGISTER
ANALYSIS PAGE
BUISNESS OPPURTUNITY
INVESTMENT AVERAGE
RETURN AMOUNT AVERAGE
PROFIT AVERAGE
LOSS AVERAGE
LOGOUT
CLICK
RETURN AMOUNT AVERAGE ANALYSIS
6
5
4
3
2
1
0
LINE CHART
BARCHART
CLICK
SPLINE CHART


ENTERPRENEUR ANALYSIS
HOME
ABOUT US
CONTACT US
LOGIN
REGISTER
ANALYSIS PAGE
BUISNESS OPPURTUNITY
INVESTMENT AVERAGE
RETURN AMOUNT AVERAGE
PROFIT AVERAGE
LOSS AVERAGE
LOGOUT
CLICK
LINE CHART
CLICK
BAR CHART
SPLINE CHART
2017
2018
2019
2020
2021
2022


ENTERPRENEUR ANALYSIS
HOME
ABOUT US
CONTACT US
REGISTER
ANALYSIS PAGE
BUISNESS OPPURTUNITY
INVESTMENT AVERAGE
RETURN AMOUNT AVERAGE
PROFIT AVERAGE
LOSS AVERAGE
LOGOUT
CLICK
RETURN AMOUNT AVERAGE
LINE CHART
BARCHART
CLICK
SPLINE CHART


ENTERPRENEUR ANALYSIS
HOME
ABOUT US
CONTACT US
REGISTER
ABOUT US
Determinants of business start-up , long and short-term success and failure of small businesses with emphasis on their education and training


ENTERPRENEUR ANALYSIS
HOME
ABOUT US
CONTACT US
REGISTER
CONTACT US
ENETRPRENEUR ANALYSIS
MRIT MYSURU -570027
CONTACT -123456787

3.0 DATABASE DESIGN

3.0.1 ER DIAGRAM

An Entity-Relationship diagram (ERD) is a data modeling technique that graphically illustrates an information system's entities and relationships between those entities. An ERD is a conceptual and representational model of data used to represent the entity framework infrastructure.

Entity relation diagram of our project consist of admin who can see his result after direct registration and this software doesn't get controlled by any one a single user can perform is his task simultaneously.

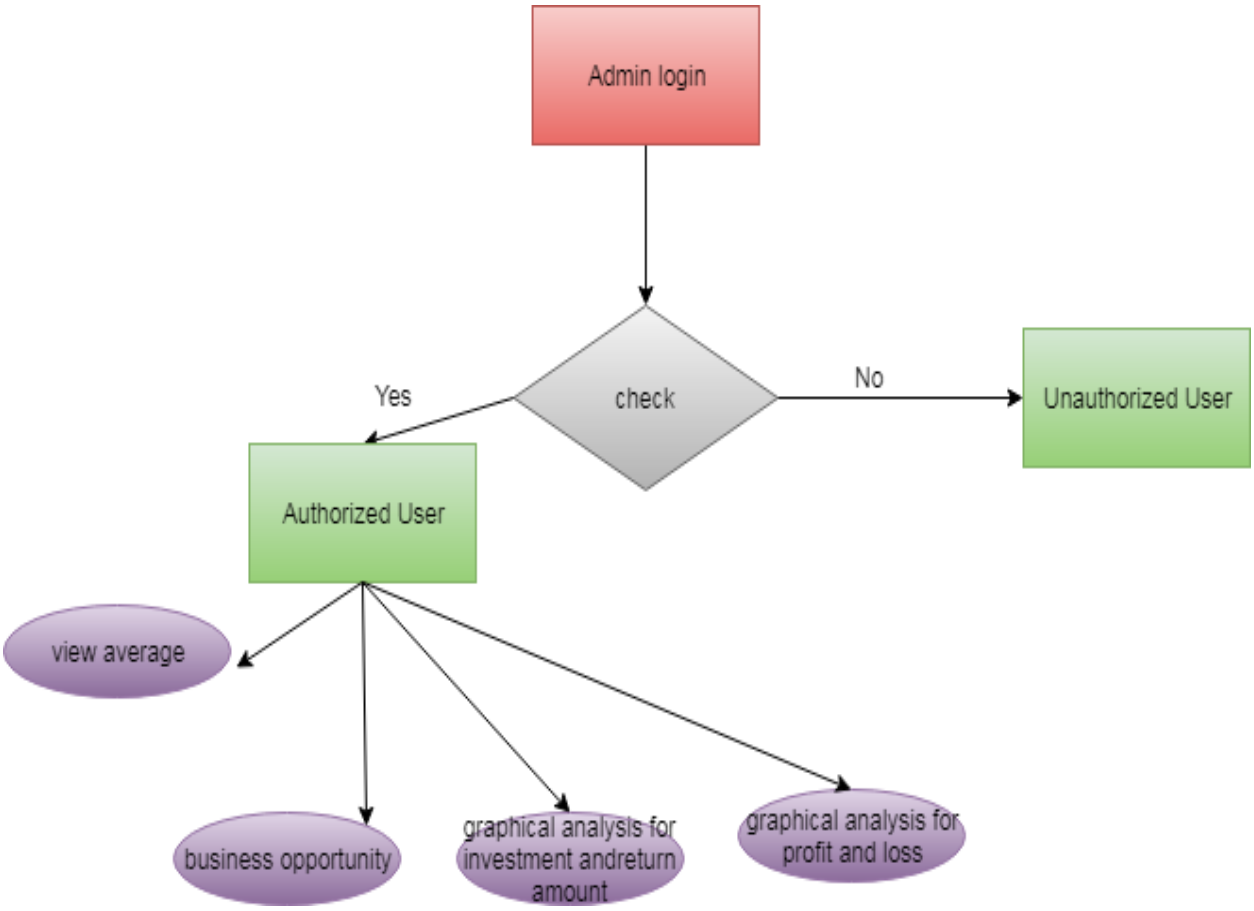
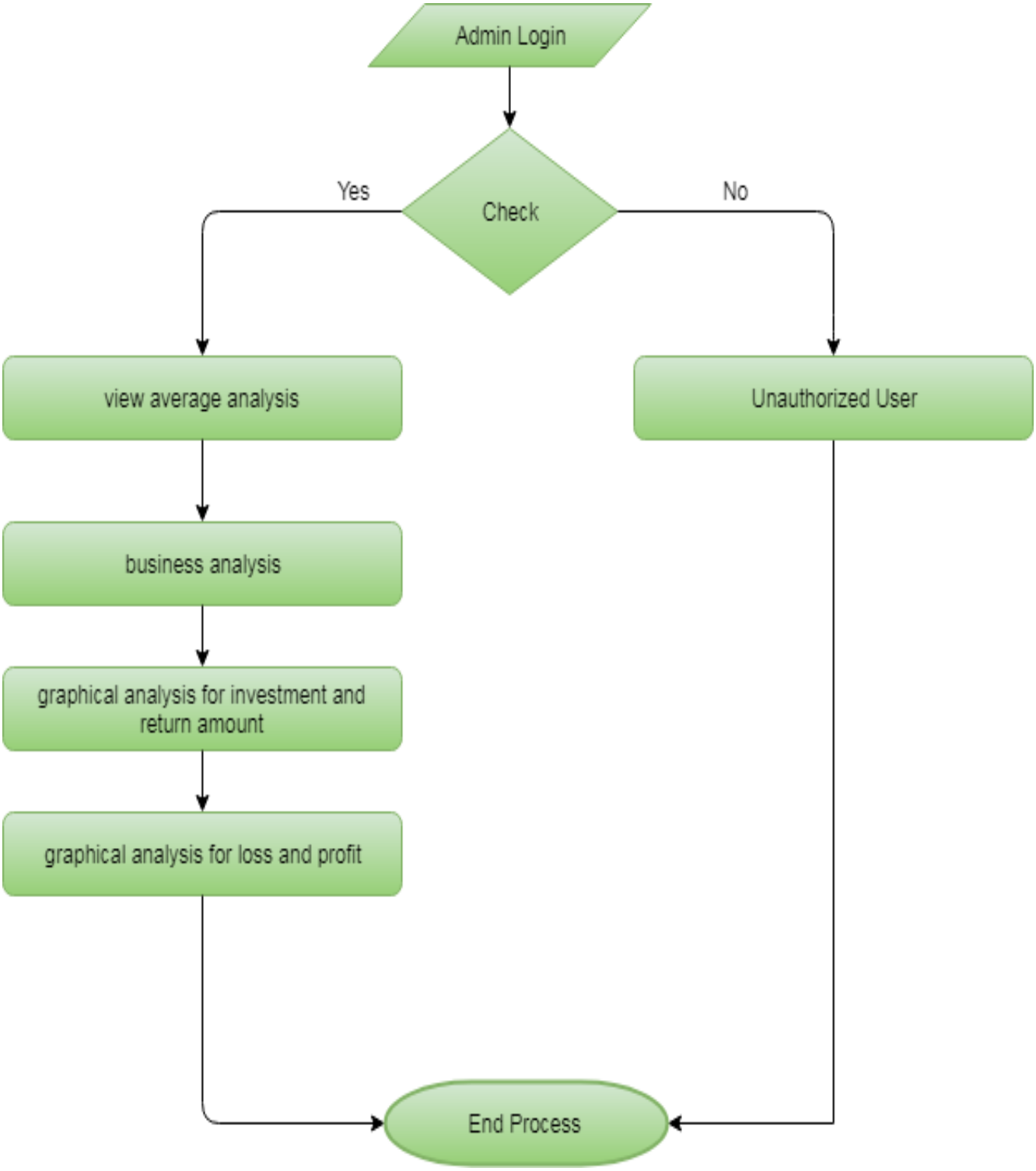


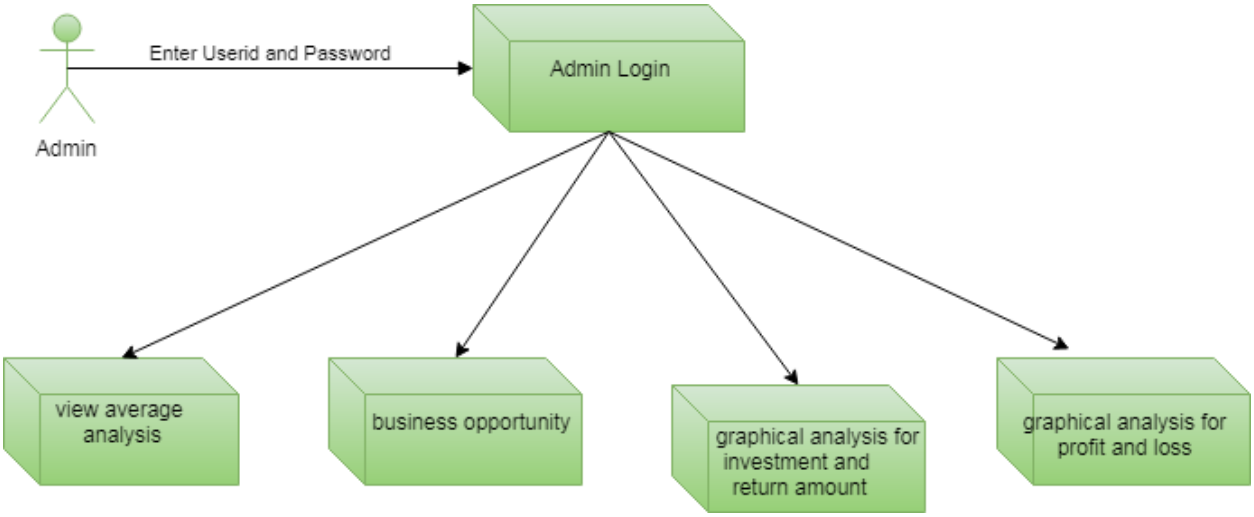
Figure: Er diagram

3.0.2 Data Flow Diagram

Data-flow diagram is a way of representing a flow of a data of a process or a system and information about the outputs and inputs of each entity and the process itself.

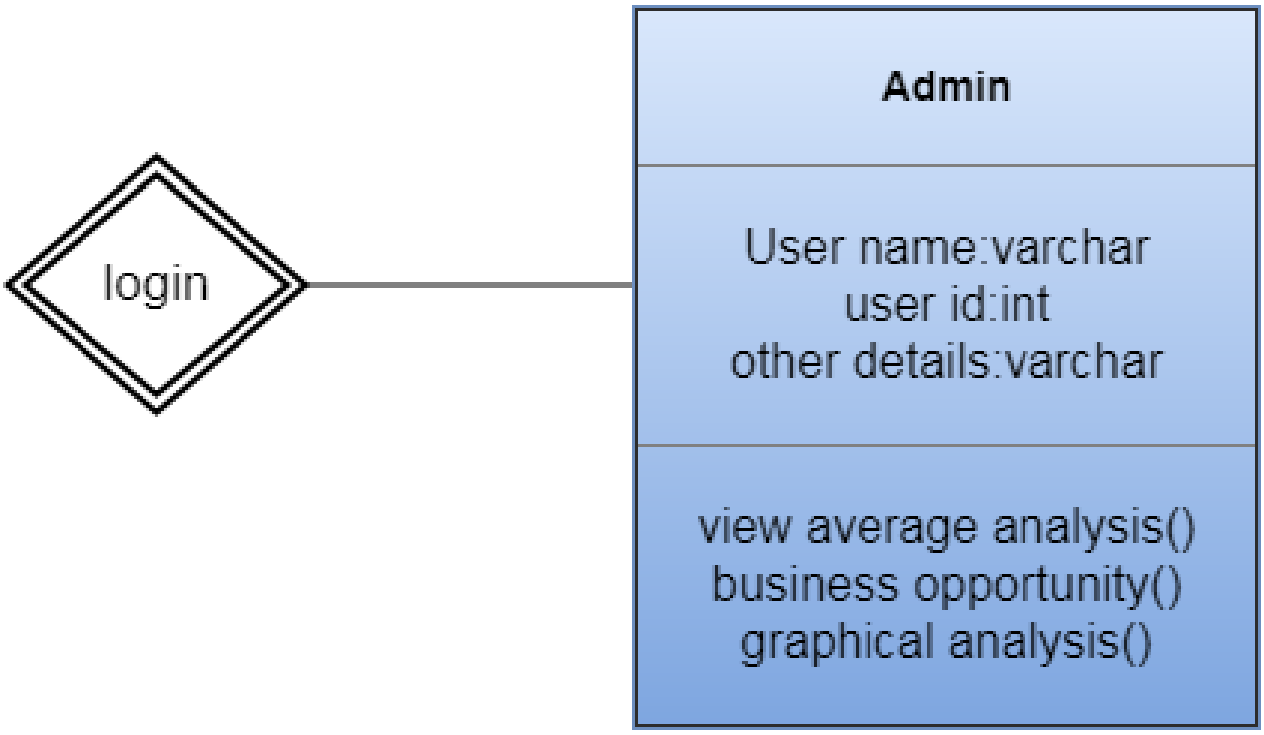


3.0.3 COMPONENT DIAGRAM

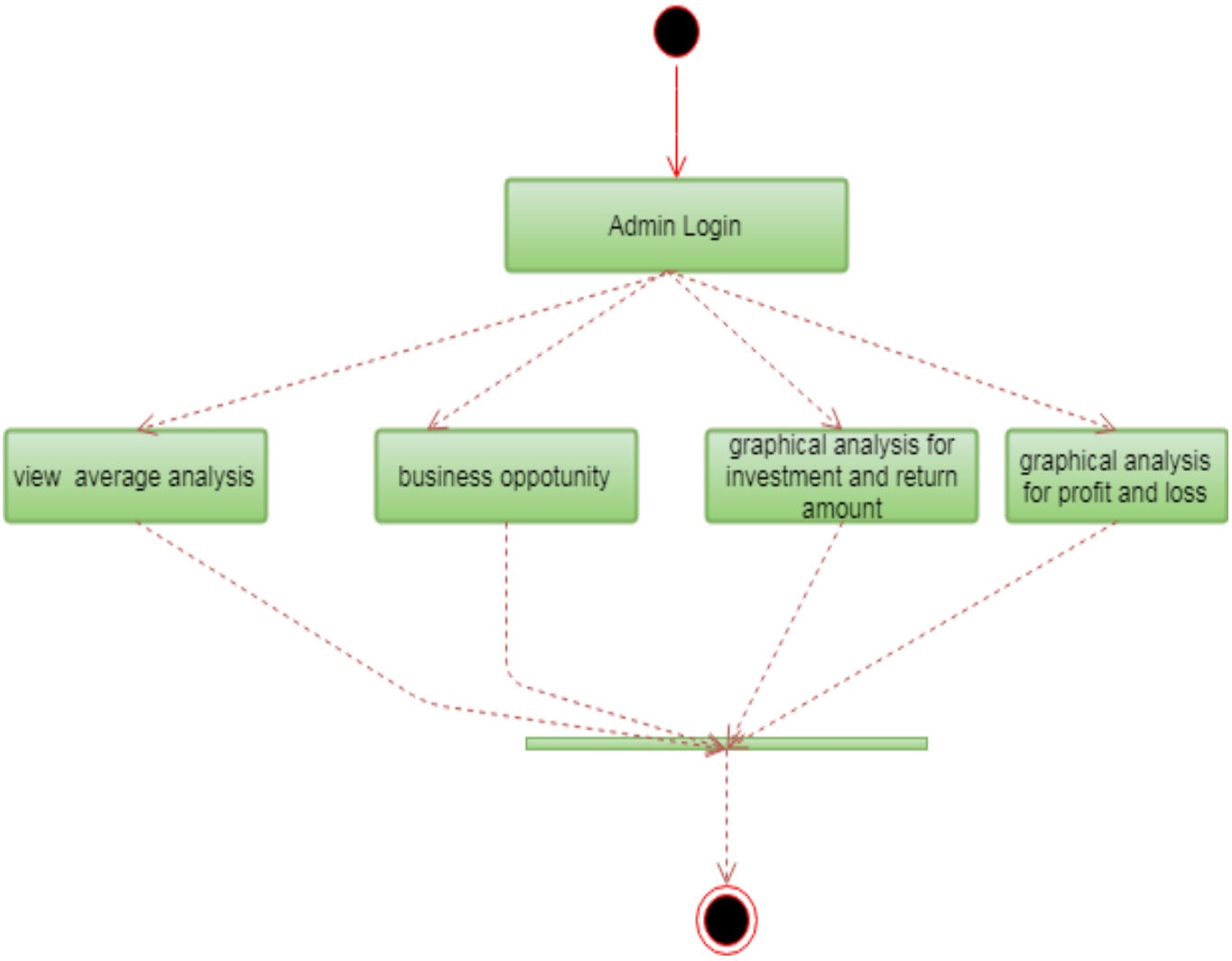


A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development.

3.0.4 CLASS DIAGRAM



3.0.5 ACTIVITY DIAGRAM



Activity diagram is defined as a UML diagram that focuses on the execution and flow of the behavior of a system instead of implementation. It is also called object-oriented flowchart. Activity diagrams consist of activities that are made up of actions which apply to behavioral modeling technology

3.0.6 Table Structure

Table Name: BusinessDetails

Column	Type	Default
id	int(11)	-
year	int(11)	NULL
field	varchar(300)	NULL
investment	int(11)	NULL
returnamount	int(11)	NULL
profit	int(11)	NULL
profitreason	varchar(1000)	NULL
loss	int(11)	NULL
lossreason	varchar(1000)	NULL
currentissues	varchar(1000)	NULL
strategy	varchar(100)	NULL

Table : main table

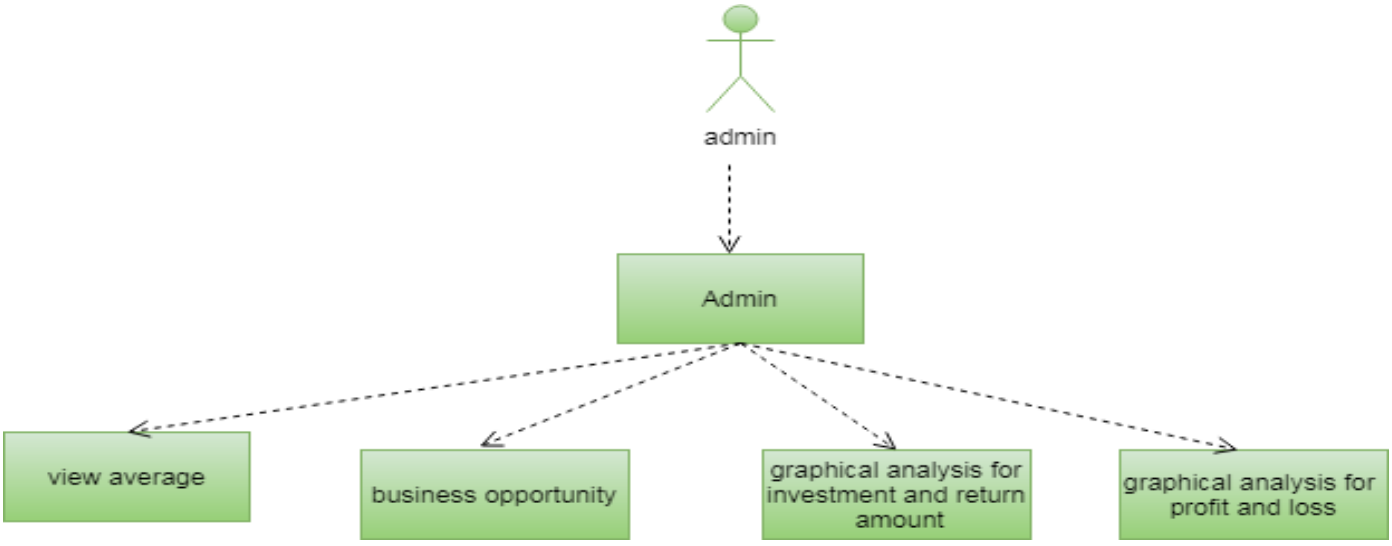
ADMIN TABLE

No.	Attribute name	Data type	Constraints	Description
1	Admin id	Varchar(10)	Primary key	Admin id
2	Password	Varchar(10)		Admin password

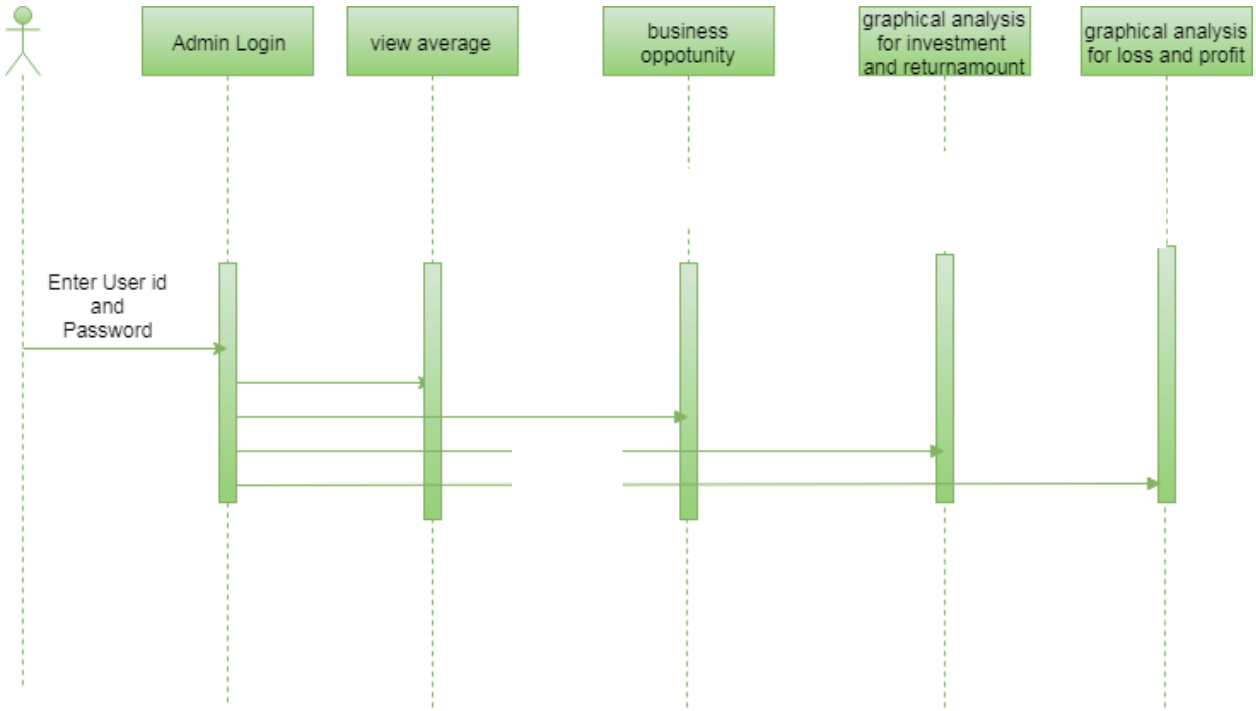
Table : Admin Details

4.0 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.



5.0 Sequence Diagram



6.0 System testing

SYSTEM TEST

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, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units 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 tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

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

Functional testing is centered on the following items:

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

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures 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.

System Test

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.

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 its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

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.

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 step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

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.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6.1 Test Cases:

TC#	Description	Expected Result	Actual Result	Status of Execution Pass/Fail
TC01	Execute/run the application	Application should run without any interrupts	------	------
TC02	Verification of Login Page	Enter User Name and Password. It should verify with database.	------	------
TC03	Verification of Admin Page input User Name and password	If Admin Login Name & Password is valid then it should navigate to respective Admin home page. If invalid then show message that Input Username & Password is wrong.	-----	-------

6.2 INPUT AND OUTPUT DESIGN

INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

OBJECTIVES

1.Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3.When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct

source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2.Select methods for presenting information.

3.Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

7.0 LIMITATION AND DRAWBACK

Although our model has enabled us to examine a new method of evaluating a hidden market process, several assumptions, limitations, and related extensions to this research need to be acknowledged. First, our key assumptions pose inherent limitations on our model. For example, although our assumption about an exogenous and independent change in the valuation of investments does not necessarily change our insights, the relaxation of this assumption could lead to more profound insights into the market. Second, a DP allows for nonlinearity, path-dependence, and unpredictability. These properties are important,

assuming that a Markov model is a close representation of decision-making in real-world entrepreneurial contexts. Third, we did not account for dependence between market factors (i.e., spill-over effects within the entrepreneur's accumulated information) and selection bias for market factors. Nor did we check the validity of the received information, which raises the question of how an entrepreneur can ensure that he/she is inputting the right information. It would be fruitful for researchers to examine both the selection and validation of market factors (e.g., financial) that

we investigated, as well as factors that we overlooked (e.g., political and regulatory). Finally, it would be helpful to examine how entrepreneurs adapt to market realities while their internal processes and technologies evolve. These areas, if explored, could provide important insights for the fields of strategy, OM, and entrepreneurship.