Recommenders for Social Places(Bars, Restaurants, etc.)

Team Members:

1. Disha Patel (A20452866)
2. Nishant Pathare (A20444988)

1. Project Overview

Recommender Systems are algorithms that allow us to give the most relevant and accurate items to the user by mining important information from a large dataset(basically user's data). Recommendation engines gain information by figuring out data patterns in the data set by learning consumers' choices and outputs the outcomes that co-relates to their needs and interests.

Our aim was to use a **yelp api** to create an interactive website to provide recommendation features as well as restaurants based on the zipcodes to the user.

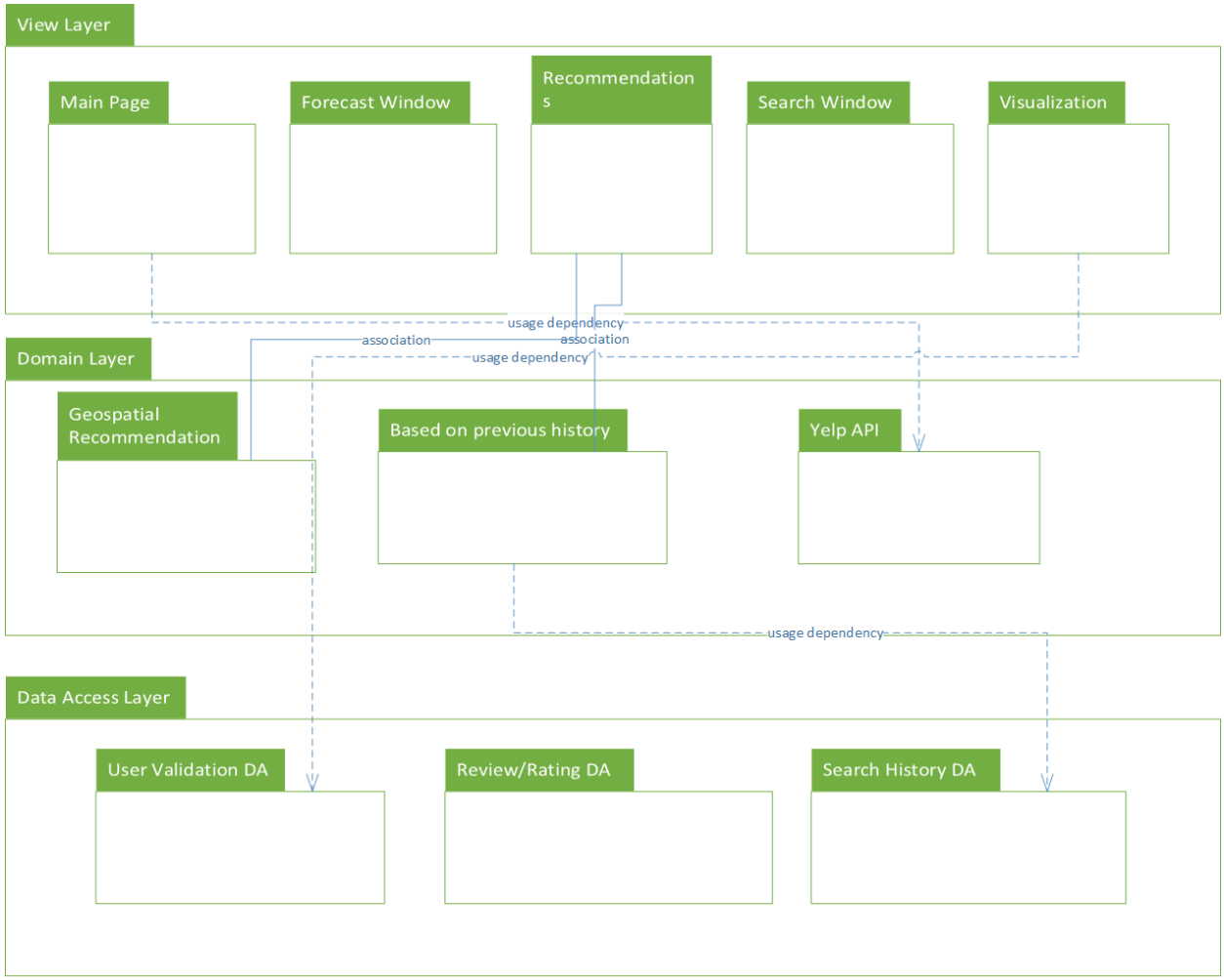
Here, the user can enter a zip-code and select the categories of food and as a result he/she will get a list of top 10 restaurants and recommended restaurants. The user can view reviews and write reviews. However, if the user wants to write a review he has to login first. We used mongodb atlas to store user reviews and have used it for creating a recommendation system using python.

We have also implemented visualizations for the users based on restaurants rating, price and review counts. We have used charts.js libraries for implementing data visualization

2. Requirements and Features

- Enter zip code
- Select categories
- Search Restaurants
- Show recommended restaurants based on user history
- Geospatial search compatibility
- Read/Write review
- View review/ratings/price comparisons using bar chart etc

3. Package Diagram



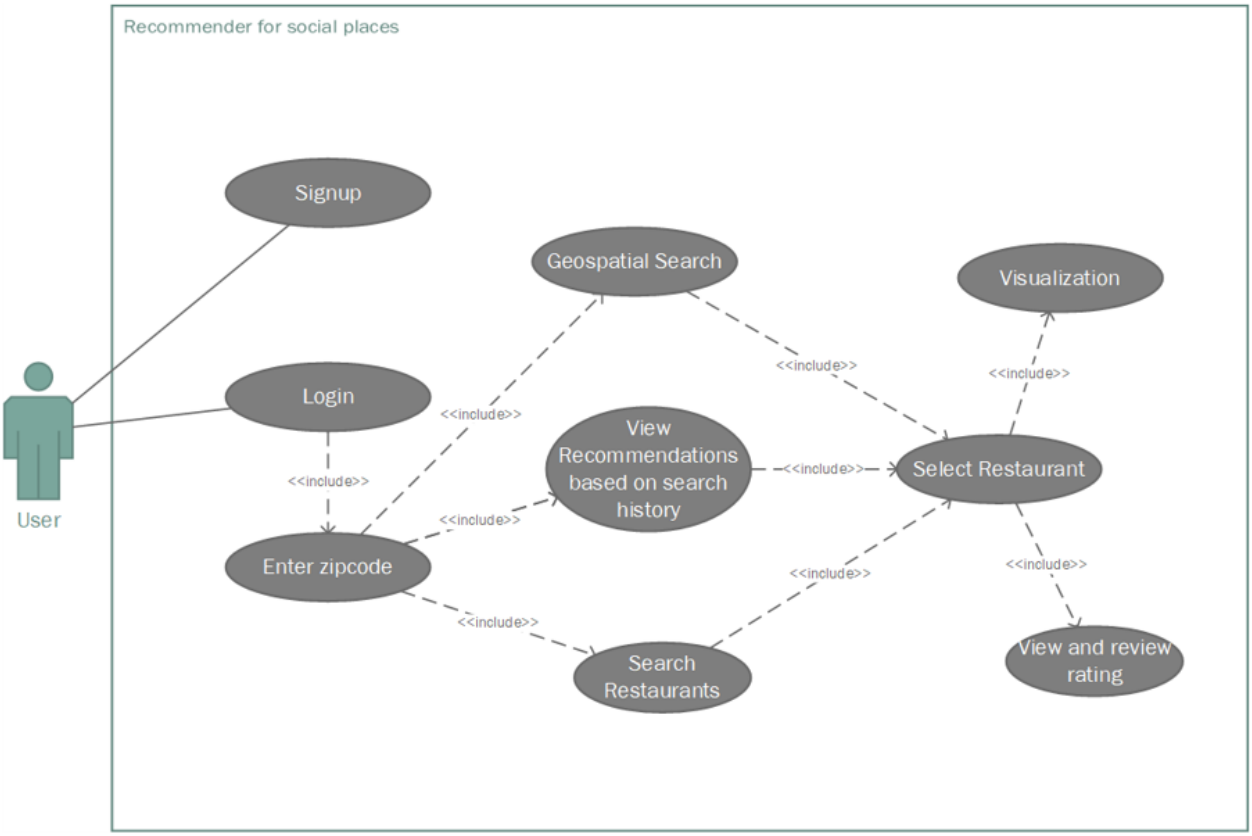
4. Use Case

- ➢ **Actors**
  - Users
  - System(comprises all information)
- ➢ **Use case list**

No	Use Case	Description
1.	Register a user	User can register using userid and password
2.	Login	User can access the content of the website using valid credentials

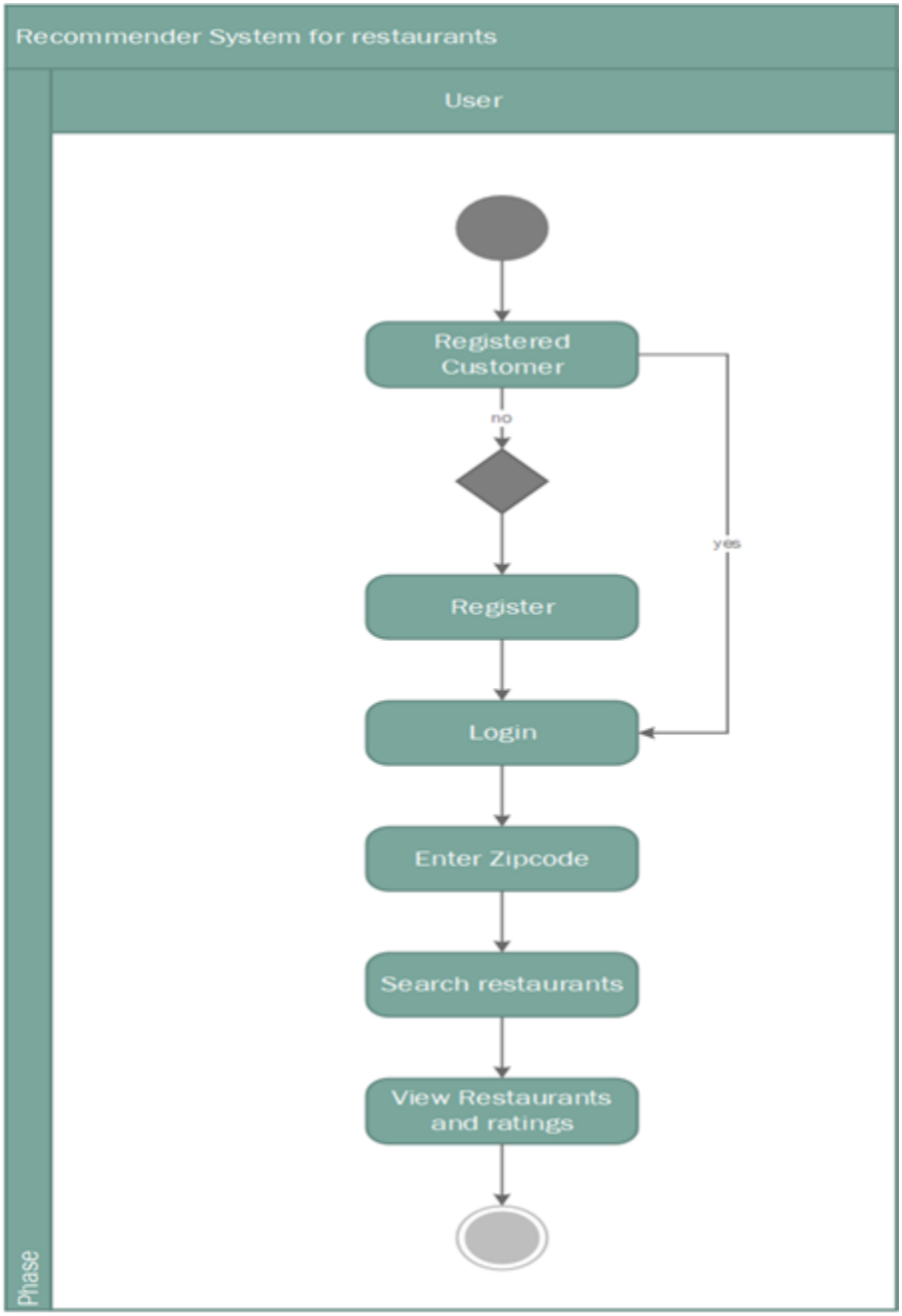
3.	Search Restaurants	User can search for restaurants
4.	View recommendation	User can view recommended restaurants based on search history, zip code
5.	Give/view ratings	User can view ratings as well give ratings
6.	View visualization	User can view the visualization based on rating/review counts/price

➢ **Use Case Diagram:**
A use case diagram is a representation of interaction between user and the system, and with different use cases with which he is involved. They provide the simplified and graphical representation of what actually the system must do.

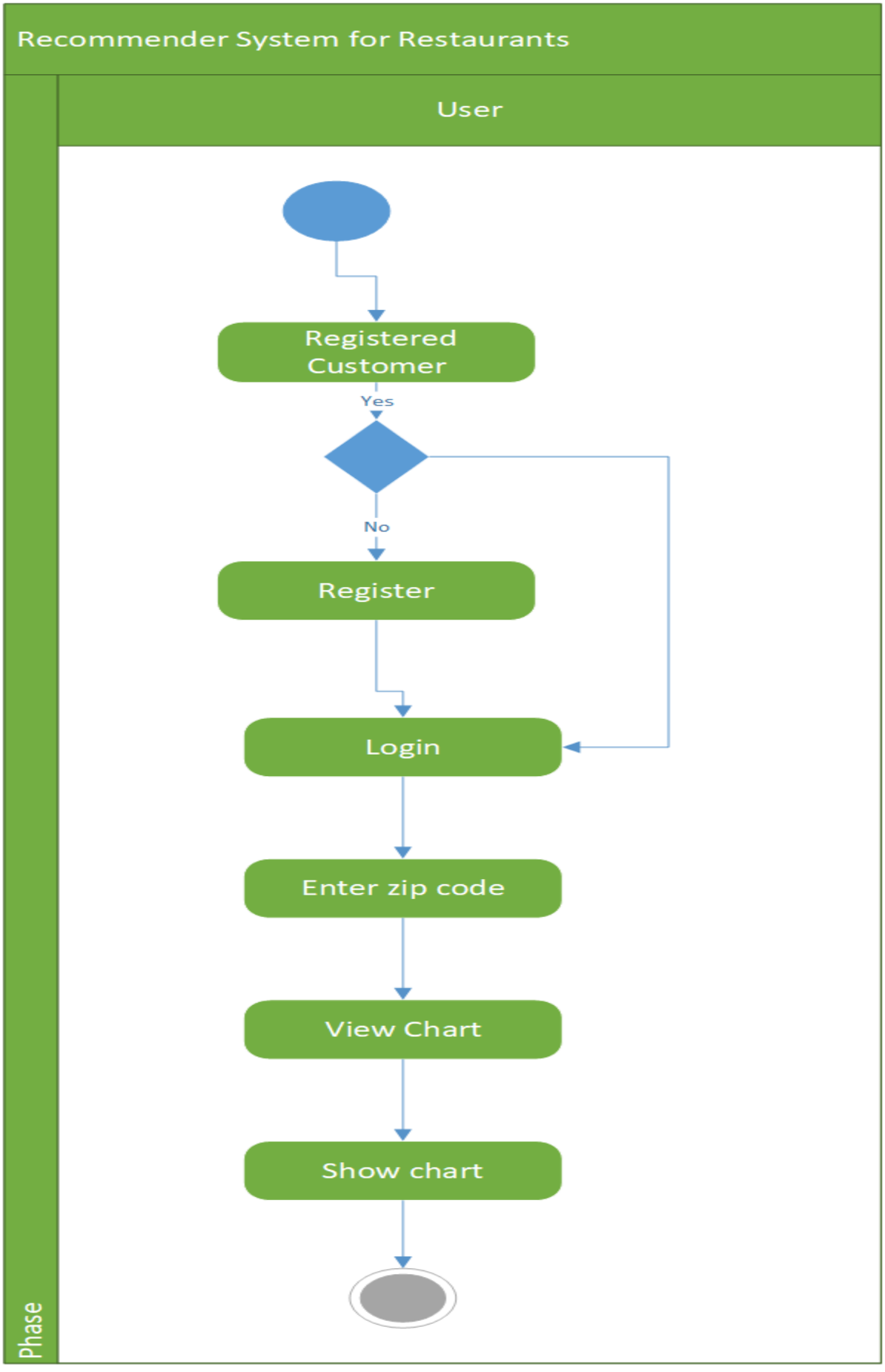


5. Activity Diagram

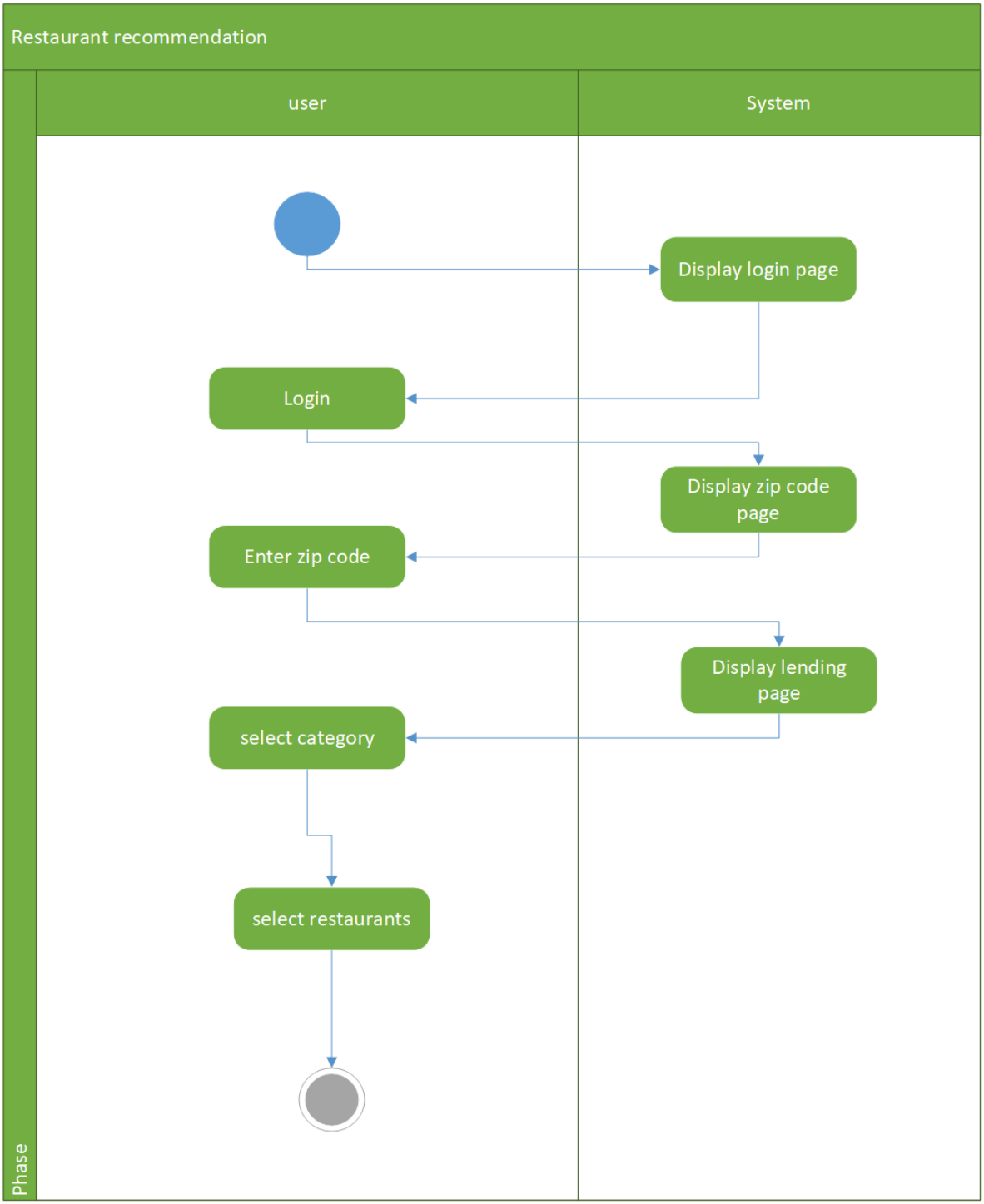
- Activity Diagram for Registration



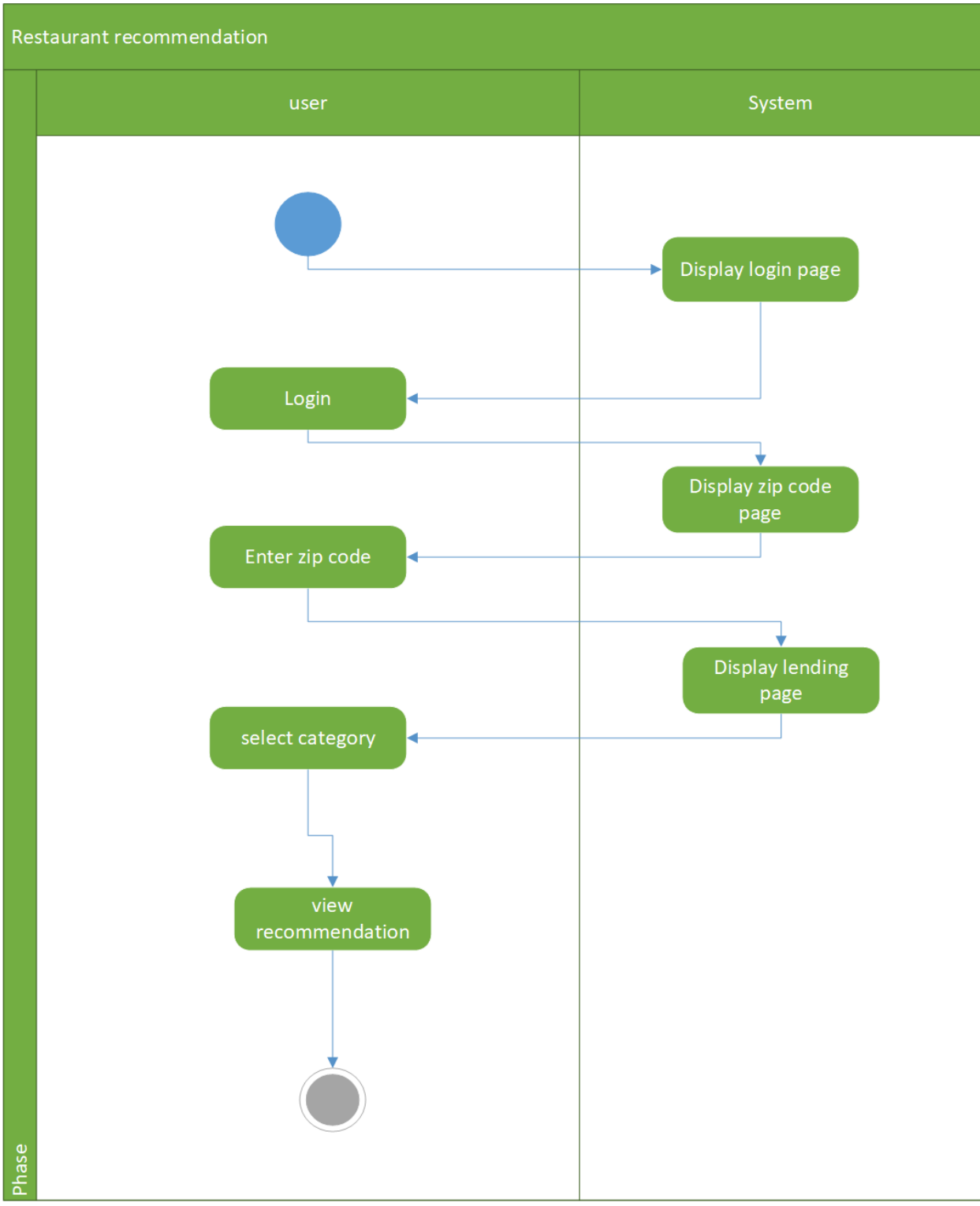
- Activity Diagram for View Chart


Recommender System for Restaurants
User
Registered Customer
Yes
No
Register
Login
Enter zip code
View Chart
Show chart
Phase

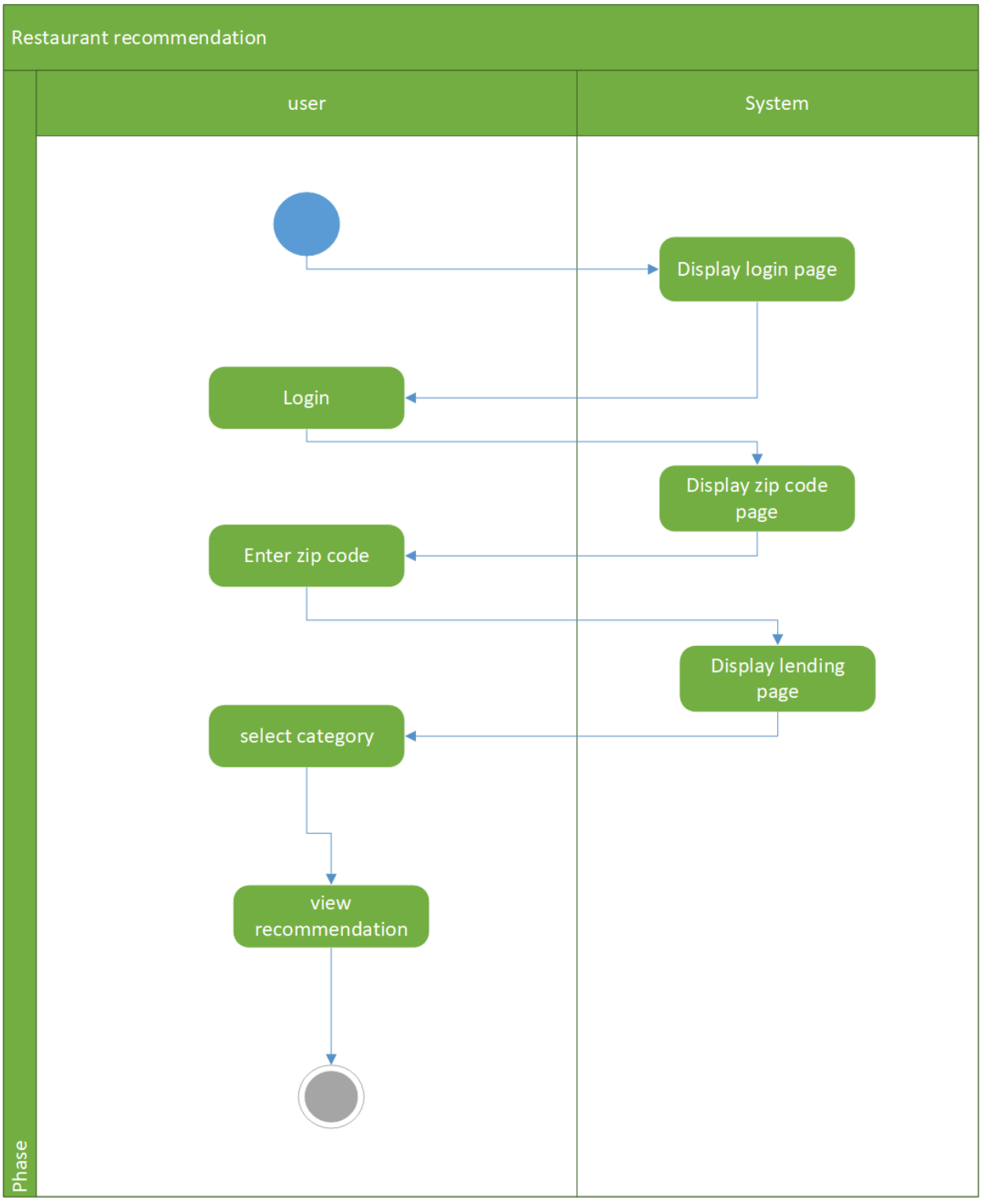
➢ Activity Diagram for Select Restaurant



➢ Activity Diagram for View Recommendation

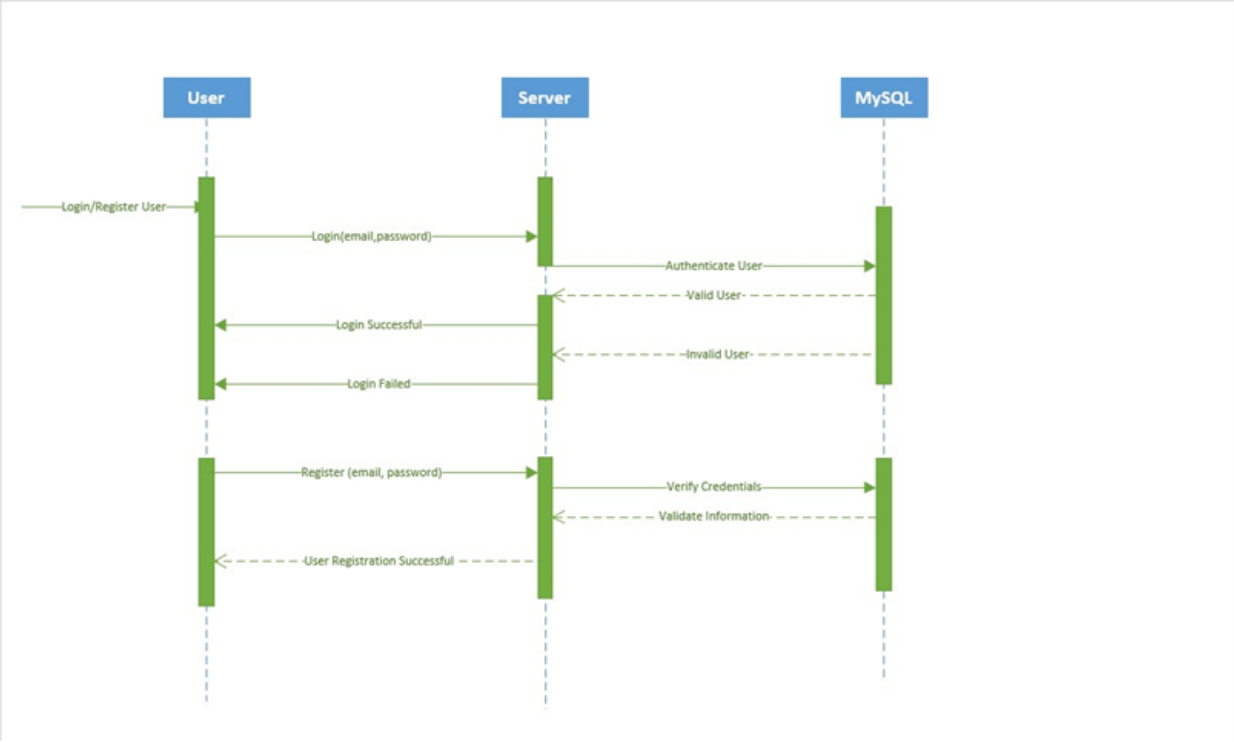


➢ Activity Diagram for Write Review

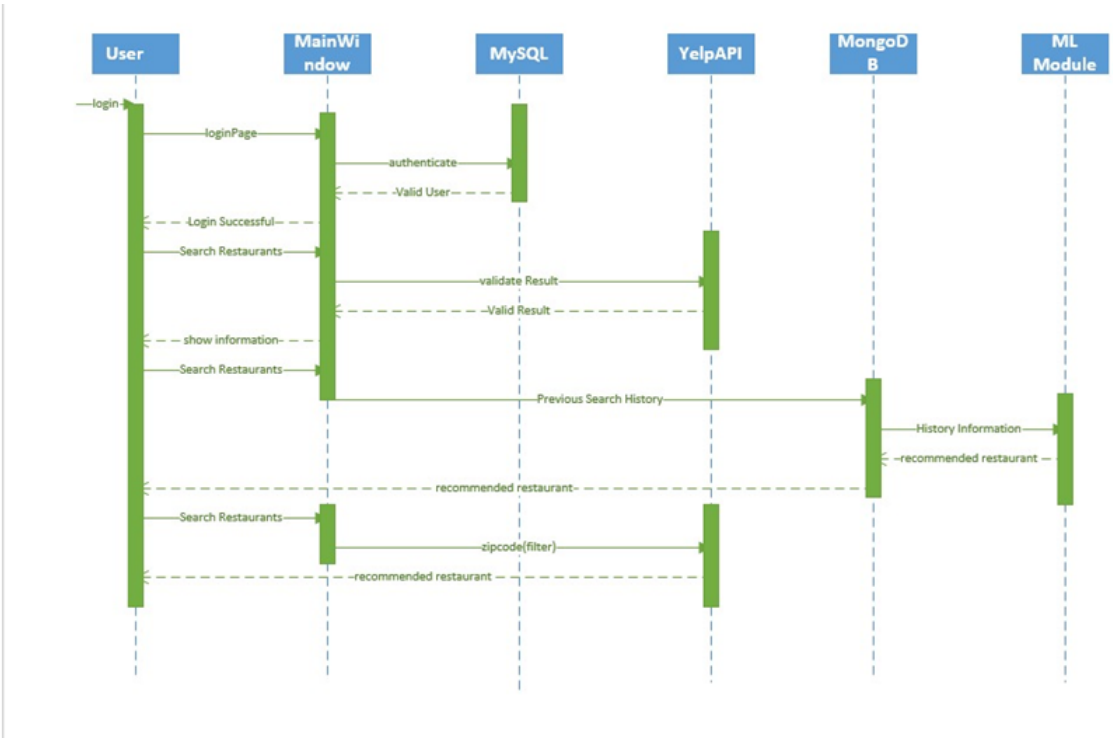

Restaurant recommendation
user
System
Display login page
Login
Display zip code page
Enter zip code
Display lending page
select category
view recommendation
Phase

6. Sequence Diagram

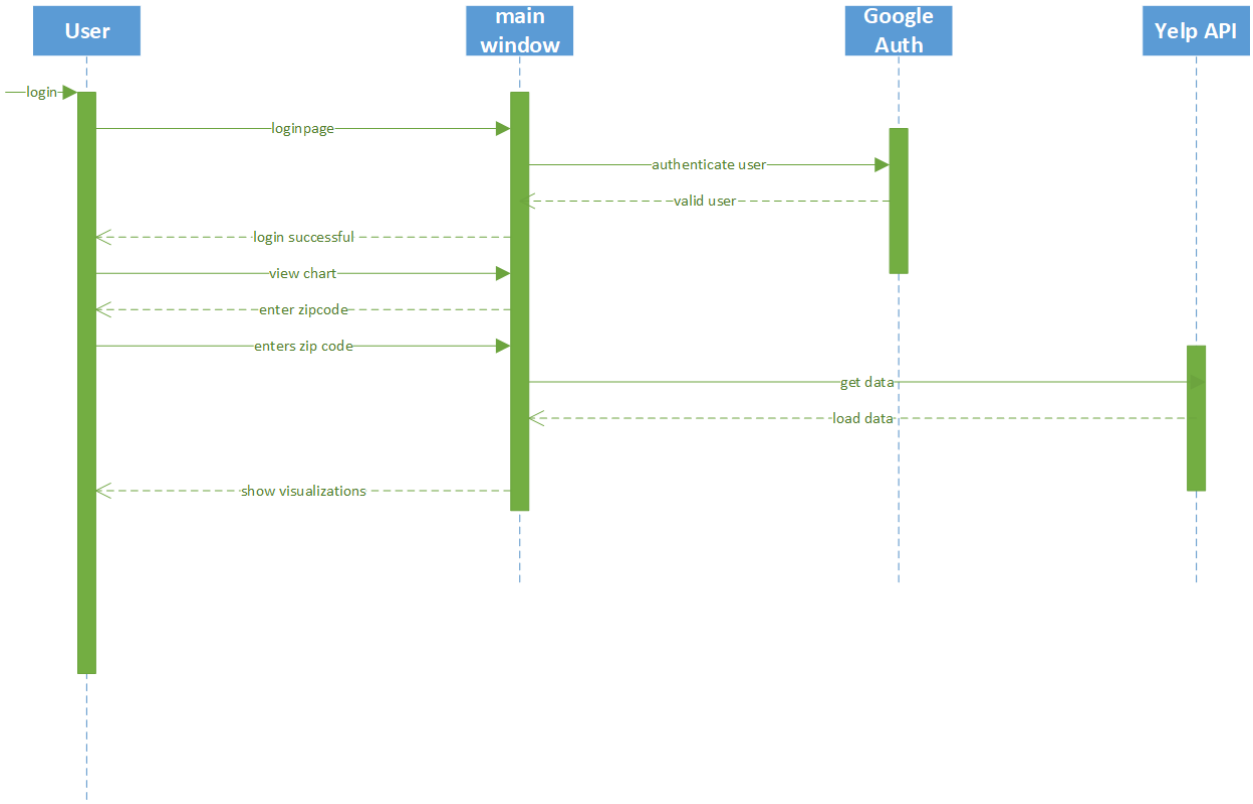
- User login and Registration



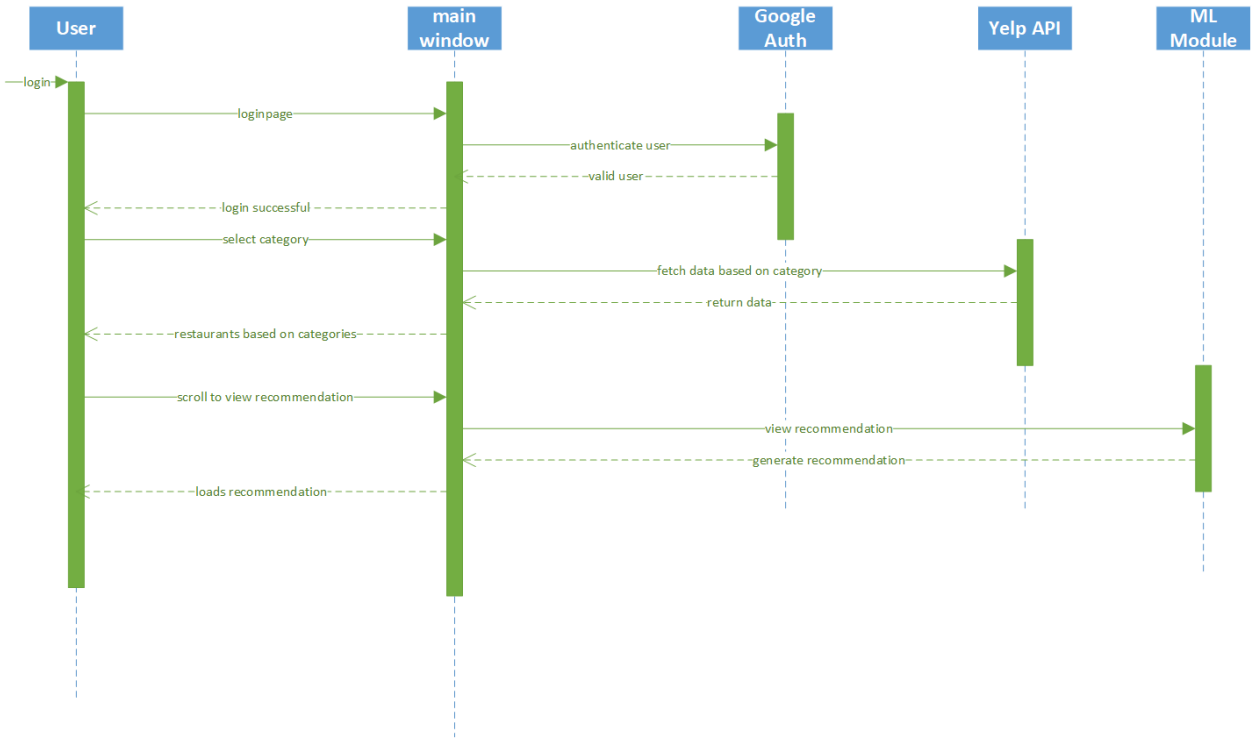
- Search Restaurants



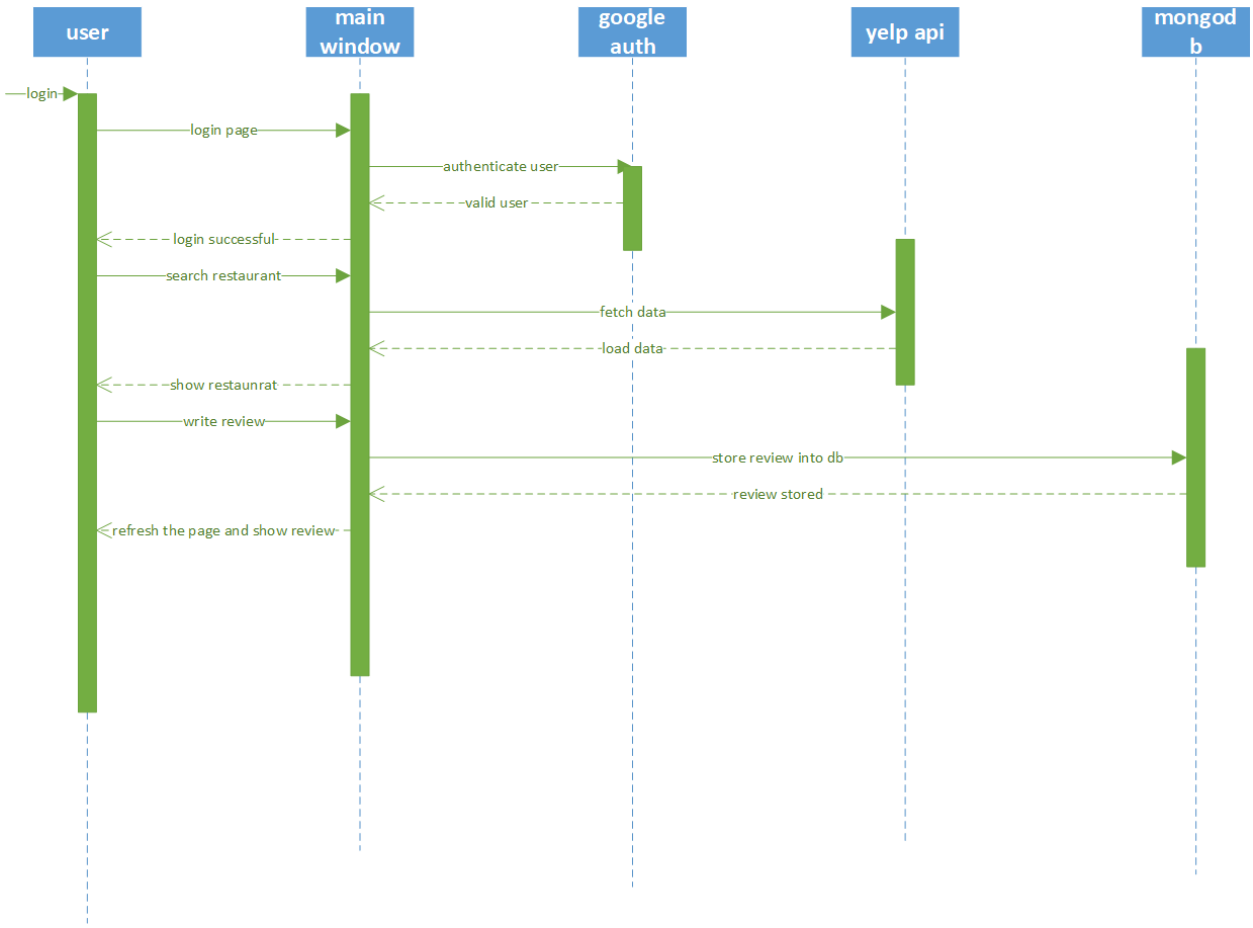
➢ Visualization Sequence diagram



➢ Show recommendations



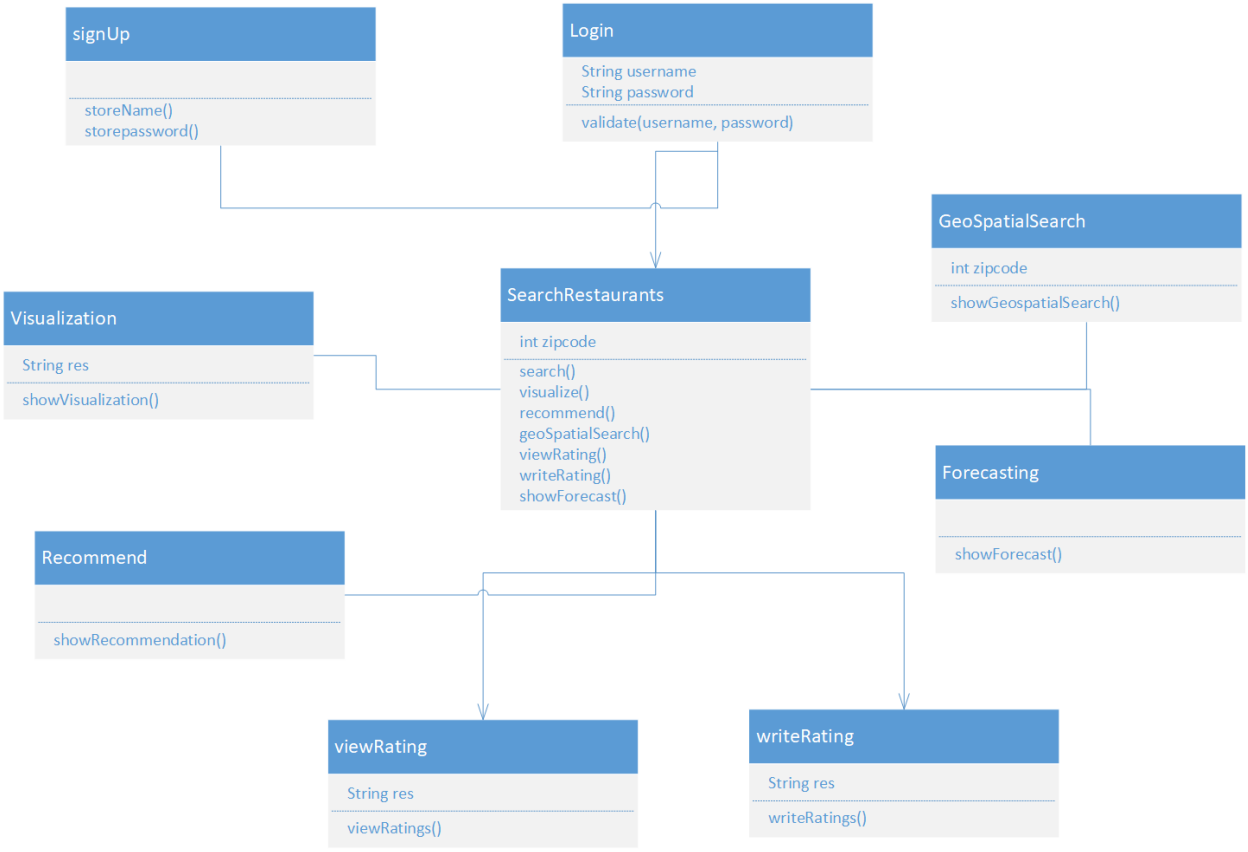
➢ Write Review



7. Complete List of Classes

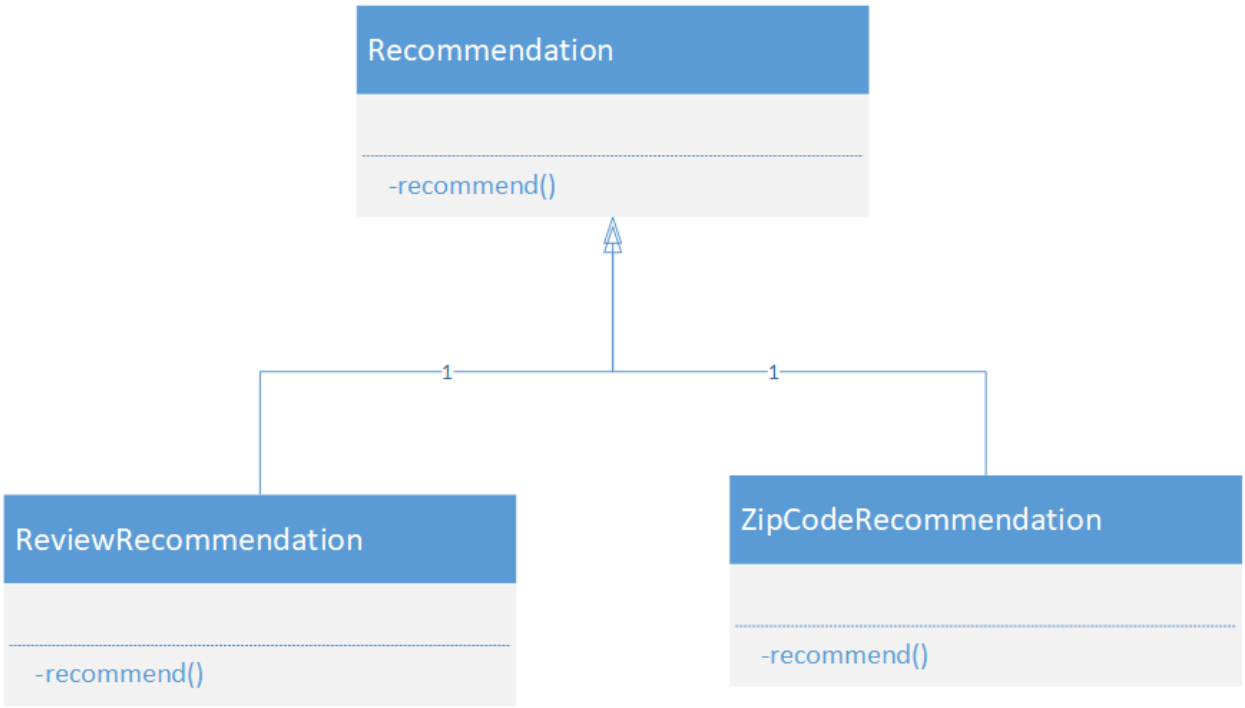
- signUp
- Login
- Visualization
- Recommend
- View Rating
- Write Rating
- Search Restaurants
- Forecasting
- Geospatial Search

8. Complete Design/Class Diagram

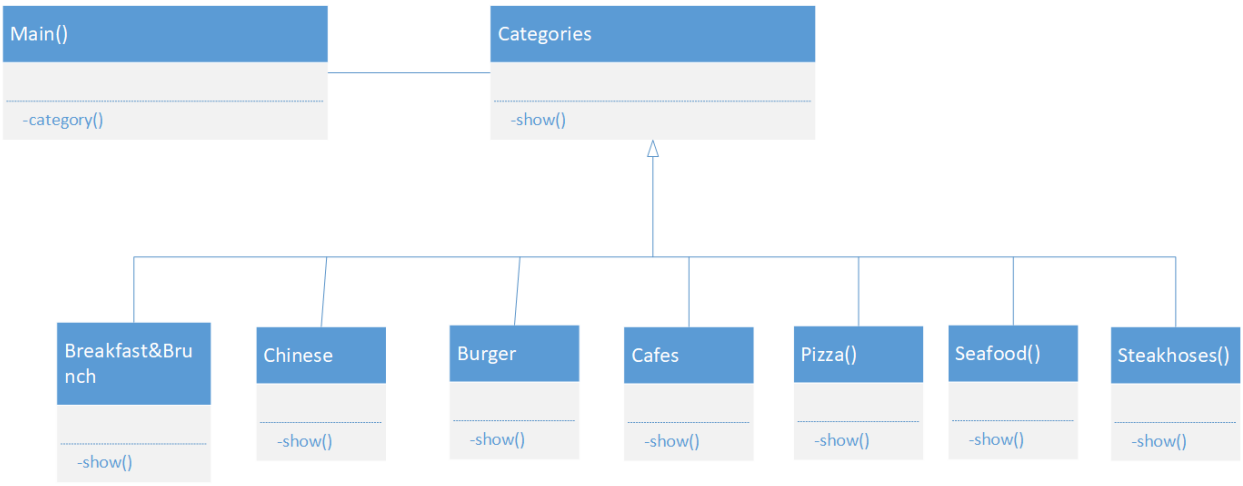


9. Class diagrams for Design Patterns used

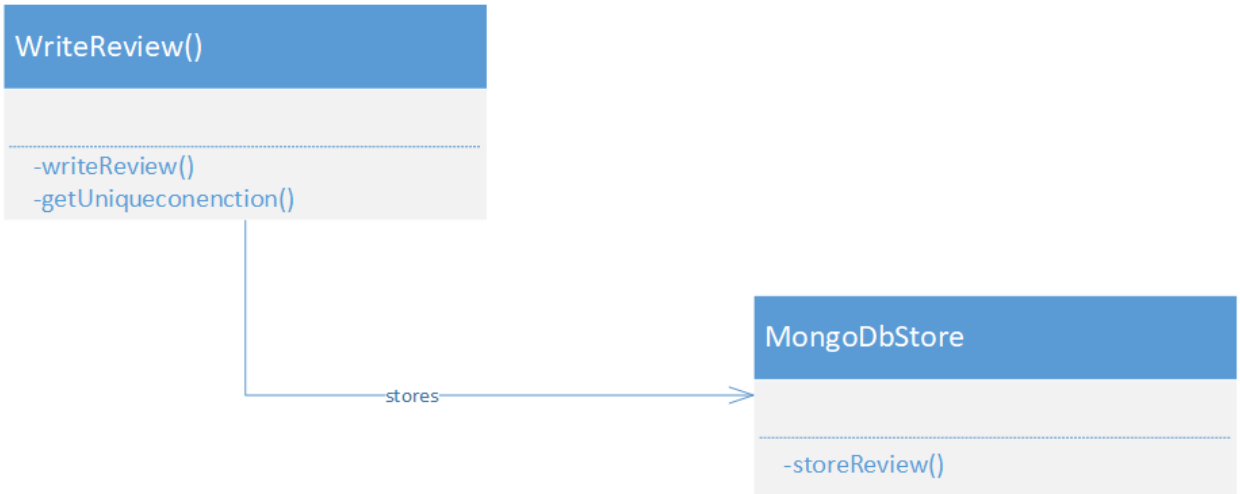
- **Template Design Pattern for Recommendation** :
  In the Template pattern, an abstract class exposes a defined template to execute its methods. Its subclasses can override the method implementation as per need, but the invocation is to be in the same way as defined by an abstract class. This pattern comes under behavior pattern category



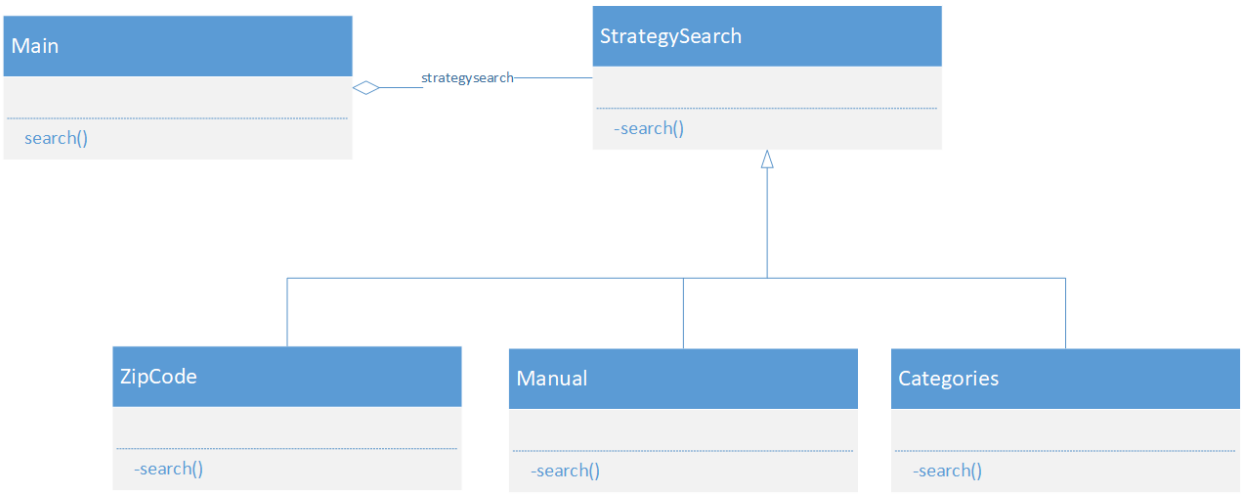
- **Composite Design Pattern for Categories** :
  It allows us to build structures of the object in the form of trees that contain both composition of the object and individual objects as nodes.Here, Composite design pattern is used in the Selection Class. It consists of three subclasses: Sort, search and by column. This design pattern is generally used, when a hierarchy is used during the coding process. Using this design pattern, we can apply the same operations over both the composite and individual. This pattern allows us to create part to whole hierarchies. It allows clients to treat both the individual as well as the composite object with uniformity i.e. both will implement all the attributes and methods of the composite object.



➢ Singleton Design Pattern for Database Connection:
Used for establishing connection with the database module. Singleton pattern is a software design pattern that restricts the instantiation of a class to one object. This is useful when exactly one object is needed to coordinate actions across the system. The concept is sometimes generalized to systems that operate more efficiently when only one object exists, or that restrict the instantiation to a certain number of objects. The term comes from the mathematical concept of a singleton.



➢ Strategy Design Pattern for Search Restaurants:
Many different algorithms exists for the same task.We don't want to support all the algorithms if we don't need them. If we need a new algorithm, we want to add it easily without disturbing the application using the algorithm.Many related classes differ only in their behavior. Strategy allows to configure a single class with one of many behaviors.I have defined a family of algorithms for search and encapsulated into on and make them interchangeable depending on customers need.



➢ Client Server Design Pattern:
A Client-Server Architecture consists of two types of components: clients and servers. A server component perpetually listens for requests from client components. When a request is received, the server processes the request, and then sends a response back to the client. Here the user tries to login and server validates where the user is authenticated or not.