Abstract

Tourism is a vital industry that significantly contributes to the economy of many countries. In our current day, social media became a main source of information for travelers seeking recommendations and reviews about various destinations. However, the current social media platforms often lack dedicated spaces that focus solely on promoting tourism. The purpose of this report is to address the development of a social media app along with an admin panel webpage. The primary objective behind this project is to provide a tourism-oriented platform that fosters user engagement within the travel and blogging community. This app will go beyond standard social media features, allowing users to share travel experiences, interact with other travelers and tourism-related pages, and discover new destinations.

The problem countered by this project is the scarcity of social media platforms that promote tourism which creates a struggle for travelers to find communities that shares their interests and offers valuable insights. Additionally, the existing platforms lack authenticity, unbiased opinions. These platforms market and sponsor content that neglect negative aspects of a location, which portrays an idealized picture of the place. Hence, leading to unrealistic expectations and potential disappointment.

This project aims to address these issues by developing a social media platform specifically designed to promote tourism. The platform will focus on providing travelers with a place to share, chat, discover authentic reviews and information about various destinations. By offering a community of genuine travel enthusiasts and bloggers, the platform will serve as a reliable source for people to explore new places confidently. Alongside with the platform, the accompanying admin panel will serve as a way of filtering and moderating content to keep the app’s space safe, positive, and to enforce the application’s policies.

The app while incorporating standard social media platform features such as likes, comments, and chats, offers some unique features like posting your own experience, moments in the form of an album and it goes by the name ‘activity’. An activity is either a user’s documentation to a certain place (or an event in it) regardless of him owning the place or not, or a capturing of his experience and moments in this place. An Activity is made up of a bunch of media with detailed descriptions, is it basically the pillar of the application where everything from descriptions, critiques, and information is done. Activities similar to posts, can be liked and commented on, also they can be bookmarked.

Furthermore, activities are categorized by their home cities and types each into a city and an activityType respectively. This approach makes the app more organized, the search easier, and the usage friendly.

This project promises quite well results. The app provided users with a well tourist-oriented community that shares true unhindered opinions and offer valuable insights about destinations.

In conclusion, this project faces the void in tourism, blogging, and travelling that the other platforms don’t take put in their considerations. The project’s approach delivers a community that ensures valuable interactions where unbiased, unsponsored endorsements and insights, descriptions, information for famous destinations are given.

**Chapter: 1 Introduction**

**1.1 Introduction**

This chapter introduces the project's aims, objectives, and scope, along with the technological constraints involved. It outlines the problems that led to the creation of the platform and the solution it offers.

The importance of this project is reflected by the lack of social media platforms that shed light on tourism and exploration directly and fairly. This lack escalates the need to a community where travelers among users can interact with each other, post their honest opinions on different locations, and exchange information. This project ensures all of the above problems are solved by supplying an application that has the same anatomy and backbone of modern social media platforms with some extra perks. Along with an admin panel that is responsible for monitoring and moderating the content for a friendly, positive, and policy abiding environment.

**1.2 Objectives**

Mashwerna, a social media app aims to add a platform where users can interact, get authentic reviews for destinations far from sponsored and paid content. The main objectives of the project are:

* Promote tourism.
* Provide a social media platform for users to share their travel experiences, reviews, and recommendations.
* Offer a user-friendly interface that facilitates the user’s experience.

**1.3 Project Scope:**

Our project addresses the gaps and limitations in the tourism platforms. It is designed to enhance the travel experience for users and blend it with a seamless social media experience.

The platform embraces common social media methodologies (likes, comments, chats …) along with some extra features like rating to help facilitate the user experience and get the most out of both the social media and tourism platforms in one.

**1.4 Technological Constraints**

**Software used: vscode, xampp, android studio**

**Languages: HTML, CSS, SQL, Javascript, Php**

**Frameworks & Libraries:**

* **Backend: Laravel**
* **Frontend: react.js, TailwindCSS**

**Hardware: Laptop**

**1.5 Problem**

Standard tourism and travel platforms all follow the same pattern causing some problems:

* Lack of interaction between users.
* Lack of authenticity and showing only positive aspects of a place.
* Biased opinions that are based on sponsorships and paid collaborations.

**1.6 Solution**

By blending social media and tourism, Mashwerna will:

* Provide users a space to chat, interact, and discover various locations.
* Capture their own moments and experiences in certain locations.
* Share their own businesses on the platform.
* Rate and review destinations.
* Serve authentic reviews, as the reviews are only from true users and no sponsored content will be available.

**Chapter: 2 Functional and Conceptional Study**

**2.1 Introduction**

This chapter outlines both the functional and non-functional requirements for the project. Additionally, it presents UML use case and data flow diagrams with all relevant details.

**2.2 Functional Requirements**

2.2.1 User

* Signup: to access all features, guests should register and login. Otherwise, they can only view the app and search.
* Login: users must login to access their profiles.
* Activities: users can crud (create, read, update, delete) their own activities and view other users’ activities. Also, they’re able to like, comment on, and bookmark them.
* Cities: users are able to view main cities and browse their activities.
* Search: users can choose their type (restaurants, hotels ...) and search based on the city name.
* Chats: users can send messages and chat with other users.
* Rating: users can give a rating out of 5 to any other user.

2.2.1 Admin

* Users: admins can view and delete users that violate the terms, also they can promote them to admins.
* Activities: admins can view and delete inappropriate activities.
* Comments: admins can view and delete inappropriate comments.
* Cities: admins can crud (create, read, update, delete) cities.
* ActivityTypes: admins can crud (create, read, update, delete) ActivityTypes.
* Search: admins can search for users, cities, ActivityTypes, and activities and view each.

**2.3 Non-functional Requirements**

**2.3.1 Security**

* Emails: email forms are checked during registration and login.
* Authentication: Jwt authentication is used to ensure the encryption of every request from and to the server.
* Passwords: passwords are hashed and stored in the database for extra security.

2.3.2 Performance

* Response time: the average response time for login and for rendering pages in a stable environment is 1 sec.
* Concurrency: the project should be able to handle a vast number of concurrent users.

2.3.3 Maintainability

* **Bug fixes**: the app will be continuously monitored for bugs.
* Enhancements: regular updates will be released with new features and improvements based on user feedback.

**2.4 UML Data Flow and Use Case Diagrams**

**2.4.1 Data Flow Diagrams**

A data flow diagram (DFD) visually represents how data moves through a system, focusing on the flow of information, inputs, outputs, and storage locations.

**2.4.1.1 Context Diagram (DFD level-0)**

The Context Diagram represents the system as a single high-level process and displays its relationships with external entities.

**2.4.1.1.1 Website: Context Diagram (DFD level-0)**

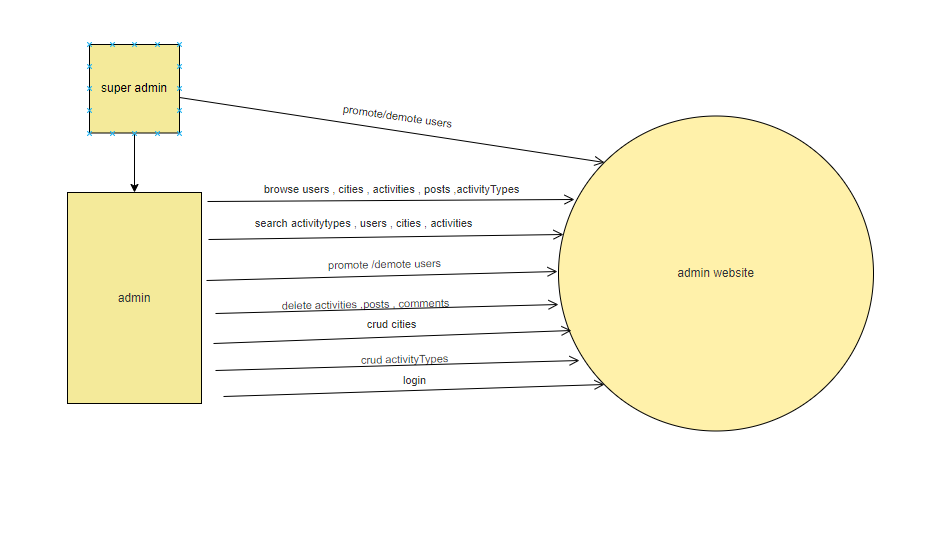
****

Figure 8: Website Context Diagram (DFD Level-0)

**3.4.1.1.2 Mobile App: Context Diagram (DFD level-0)**

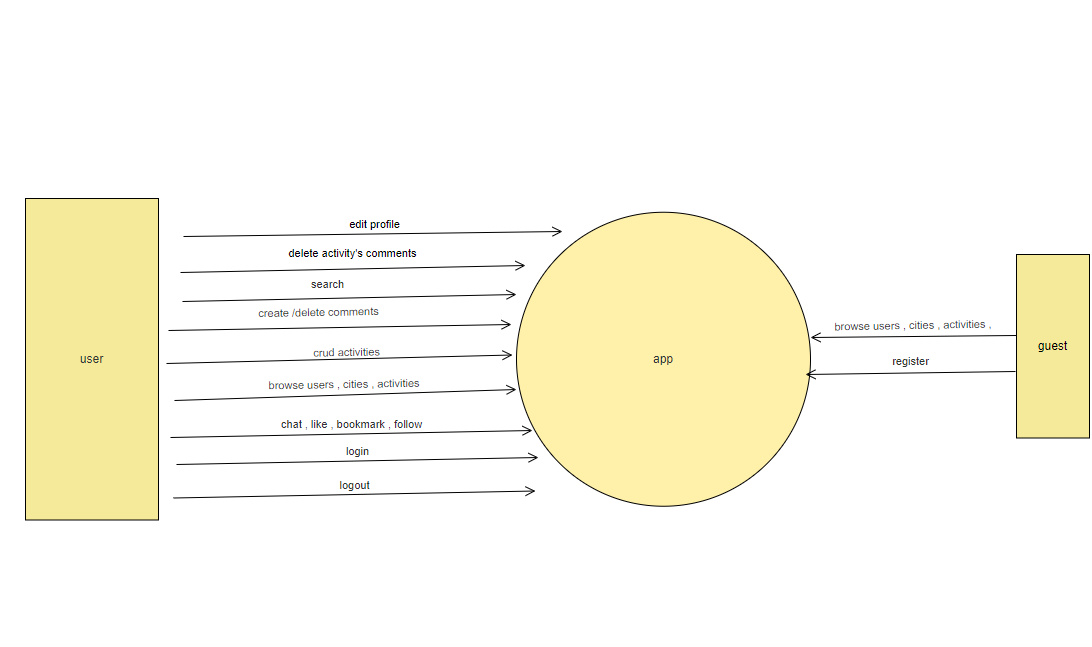
****

Figure 9 Mobile App Context Diagram (DFD Level-0)

**2.4.1.2 DFD Level 1 Diagram**

Level 1 DFD gives an overview of the full system in more details.

**2.4.1.2.1 Website: DFD Level 1 Diagram**

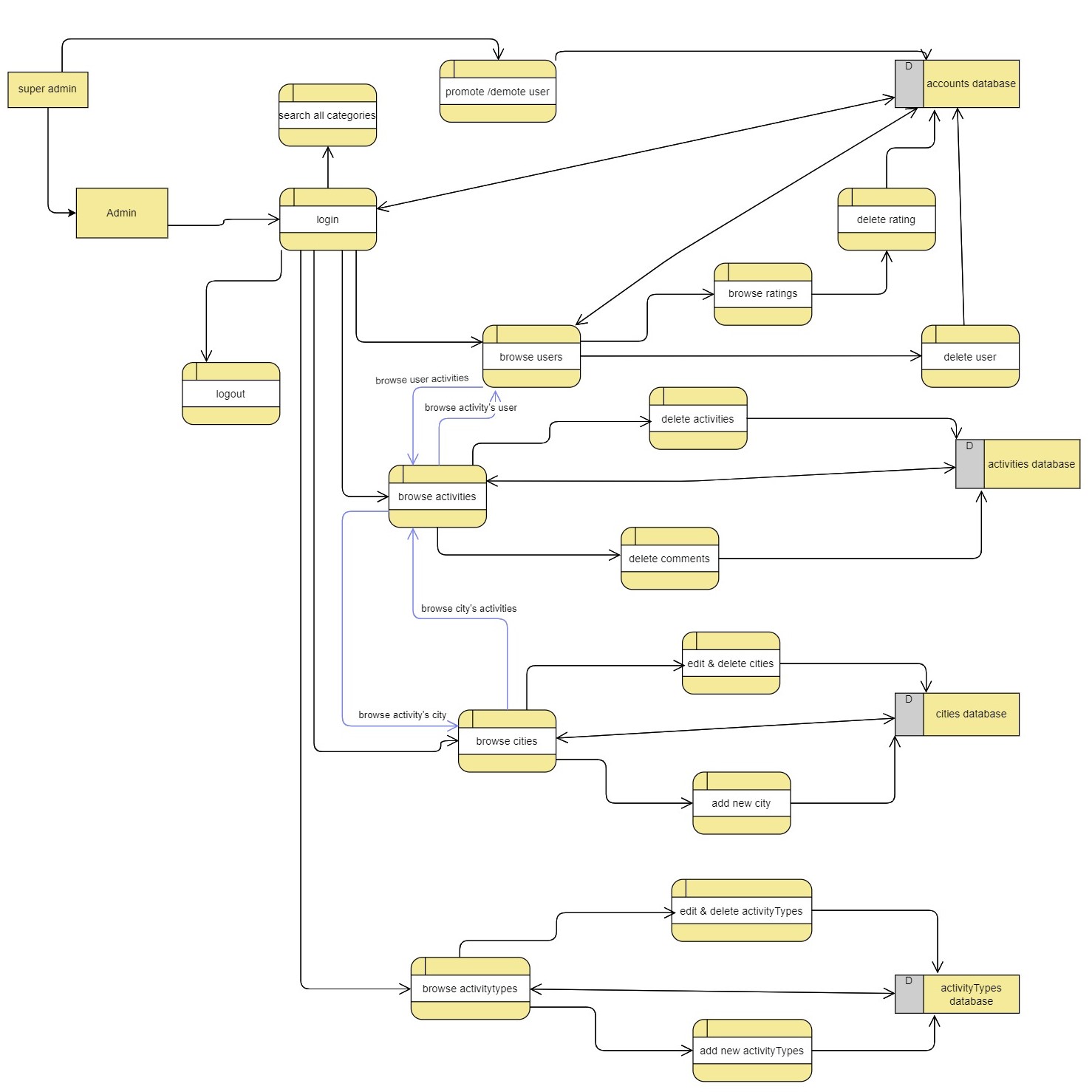
****

Figure 10: Website DFD Level 1 Diagram

**2.4.1.2.2 Mobile App: DFD Level 1 Diagram**

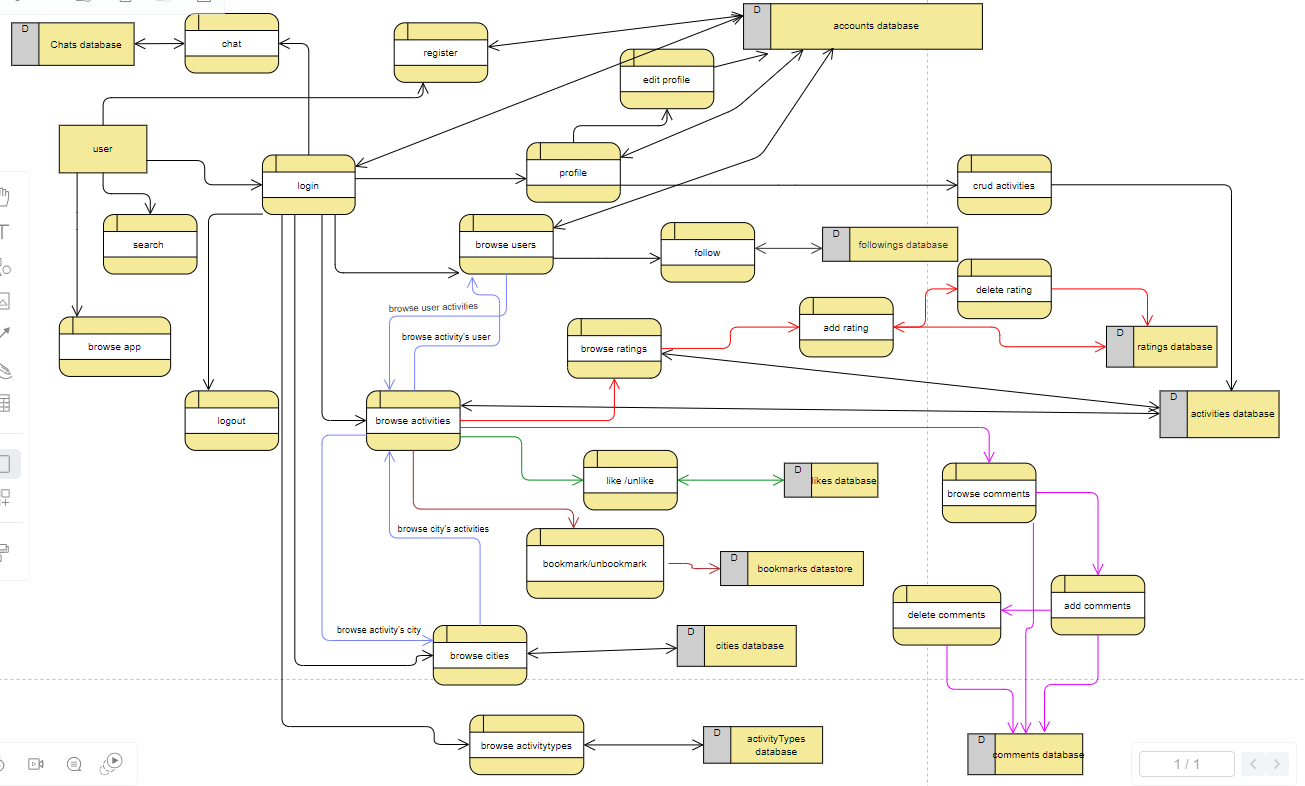
****

Figure 11: Mobile App DFD Level 1 Diagram

**Chapter: 3 System Design**

**3.1 ER Diagram**

The ER diagram represents all the models of the project with their attributes and the relations between them.

**3.2 Data Dictionary**

Table name: users

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **description** | **Key** | **Reference** |
| id | Int | User ID | PK |  |
| username | Varchar(255) | User Name |  |  |
| email | Varchar(255) | User Email |  |  |
| password | Varchar(255) | User Password |  |  |
| profile\_picture | Varchar(255) | Name of the picture stored |  |  |
| role\_id | Int | Indicates user role(1:admin ,2:user) | Fk | Roles(id) |

**Description:** This table stores all information about a user.

Table name: roles

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **description** | **Key** | **Reference** |
| id | Int | User ID | PK |  |
| role | Int | role | FK | Users(id) |

**Description:** This table stores roles of users.

Table name: followings

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **description** | **Key** | **Reference** |
| id | Int | User ID | PK |  |
| Follower\_id | Int | Id of follower | FK | Users(id) |
| Followed\_id | Int | Id of followed user | FK | Users(id) |

**Description:** This table stores all information of followings of followers.

Table name: chats

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **description** | **Key** | **Reference** |
| id | Int | User ID | PK |  |
| Message | Varchar(255) | Message sent |  |  |
| sender\_id | Int | Id of msg sender | FK | Users(id) |
| receiver\_id | Int | Id of msg receiver | FK | Users(id) |
| Isdeleted\_receiver | Tinyint(1) | Flag to check if receiver deleted the msg |  |  |
| Isdeleted\_sender | Tinyint(1) | Flag to check if sender deleted the msg |  |  |

**Description:** This table stores all information of chats.

Table name: notifications

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **description** | **Key** | **Reference** |
| Id | Int | User ID | PK |  |
| notification | Varchar(255) | Notification sent to user |  |  |
| sender\_id | Int | Id of msg sender | FK | Users(id) |
| receiver\_id | Int | Id of msg receiver | FK | Users(id) |
| activity\_id | Int | Id of activity | FK | activities(id) |

**Description:** This table stores all information of notifications.

Table name: likes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **description** | **Key** | **Reference** |
| Id | Int | User ID | PK |  |
| Activity\_id | Int | Id of liked activity | FK | activities(id) |
| user\_id | Int | Id of user | FK | Users(id) |

**Description:** This table stores all information of likes.

Table name: bookmarks

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **description** | **Key** | **Reference** |
| Id | Int | User ID | PK |  |
| Activity\_id | Int | Id of liked activity | FK | activities(id) |
| user\_id | Int | Id of user | FK | Users(id) |

**Description:** This table stores all information of bookmarks

Table name: comments

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **description** | **Key** | **Reference** |
| Id | Int | User ID | PK |  |
| Content | Varchar(255) | Comment’s content |  |  |
| Activity\_id | Int | Id of liked activity | FK | activities(id) |
| user\_id | Int | Id of user | FK | Users(id) |
| Is\_reply | Tinyint(1) | Flag to check if comment is a reply |  |  |
| Comment\_id | Int | Id of original comment if comment is a reply | Fk | Commnets(id) |

**Description:** This table stores all information of comments, comments can be a reply or a normal one.

Table name: activities

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **description** | **Key** | **Reference** |
| Id | Int | User ID | PK |  |
| Activity\_name | Varchar(255) | Name of activity |  |  |
| Description | Varchar(255) | Description of activity |  |  |
| Price | Varchar(255) | Price of activity |  |  |
| Location | Varchar(255) | Location of the activity |  |  |
| Picture | Varchar(255) | Name of profile picture |  |  |
| Likes\_count | Int | Number of like on activity |  |  |
| Comments\_count | Int | Number of comments on activity |  |  |
| Bookmarks\_count | Int | Number of bookmarks on activities |  |  |
| Rate\_count | Int | Number of ratings |  |  |
| Rate\_sum | Int | Sum of ratings |  |  |
| Rate\_average | Double | Rating average |  |  |
| Activity\_type\_id | Int | Type of the activitiy | Fk | Activity\_types(id) |
| City\_id | Int | City of this activity | FK | Cities(id) |
| User\_id | Int | User of the activity | Fk | Users(id) |

**Description:** This table stores all information of activties.

Table name: activity\_pictures

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **description** | **Key** | **Reference** |
| Id | Int | User ID | PK |  |
| media | Varchar(255) | Name of the media |  |  |
| Activity\_id | Int | Id of media’s activity | FK | activities(id) |

**Description:** This table stores all information of activity pictures which are photos of an activity.

Table name: activity\_types

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **description** | **Key** | **Reference** |
| Id | Int | User ID | PK |  |
| Name | Varchar(255) | Name of the activity\_type |  |  |
| icon | Varchar(255) | name of icon |  |  |

**Description:** This table stores all information of activity types which are the types of an activity.

Table name: cities

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **description** | **Key** | **Reference** |
| Id | Int | User ID | PK |  |
| City\_name | Varchar(255) | Name of the city |  |  |
| Description | Varchar(255) | Description of the city |  |  |
| Picture | Varchar(255) | Name of the profile\_picture of the city |  |  |

**Description:** This table stores all information of cities.

Table name: city\_pictures

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **description** | **Key** | **Reference** |
| Id | Int | User ID | PK |  |
| media | Varchar(255) | Name of the media |  |  |
| city\_id | Int | Id of media’s city | FK | activities(id) |

**Description:** This table stores all information of city pictures which are photos of a city.

**3.2.1 Diagram**

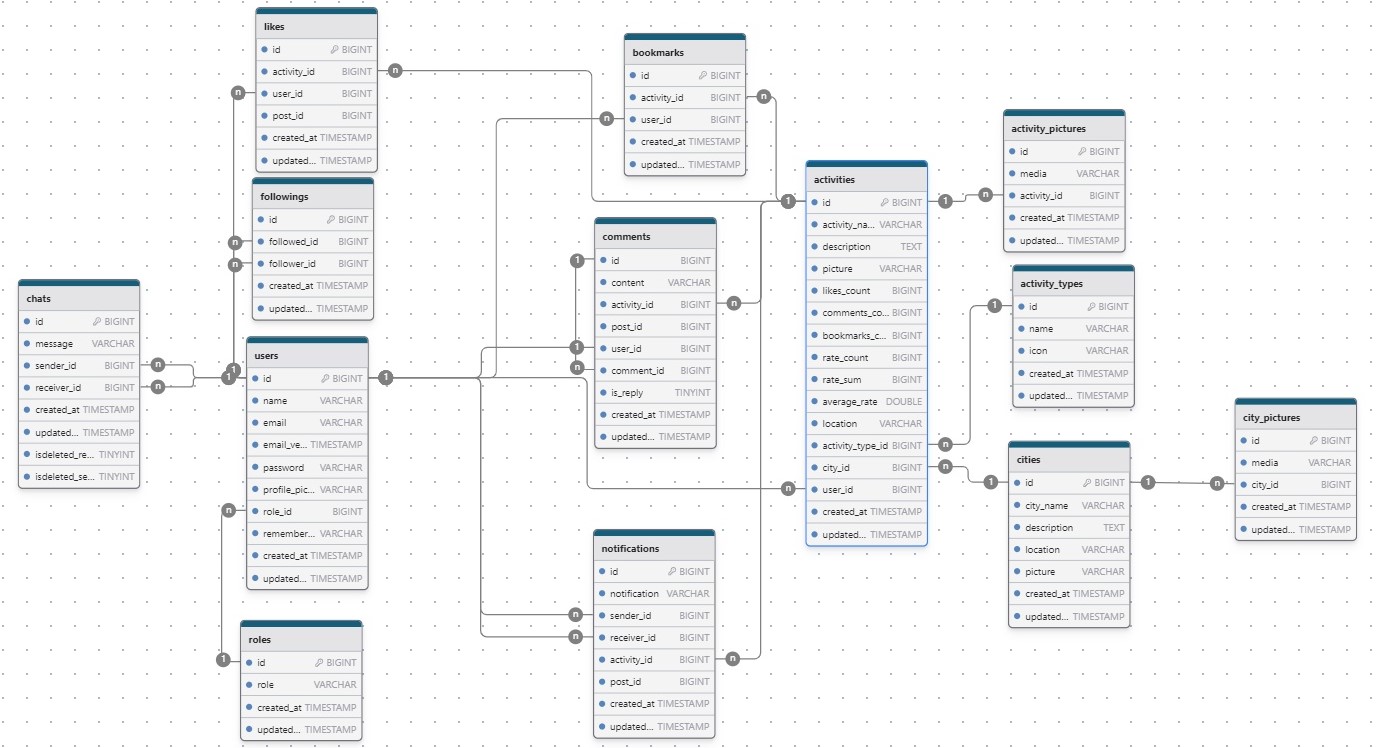
****

Figure 12: ER Diagram