Maine Meetup Mapper (P1)

Kristina, Logan, Mason, Phil

Project Description

A web platform connecting communities in Maine through events.

Discover, follow, and attend events based on personalized interests.

Key Features:

- Discover, RSVP, and Review events
- User Account System
- Personalized Event Recommendations
- Event Notifications

Functional Requirements

User and Event Management:

- Account System
- Easy event creation and RSVP tracking

Event Discovery:

- Event Categories
- Event suggestions based on user preferences
- Review System

Calendar Integration:

- Download, sync, or save events to your calendar

Non-Functional Requirements

Performance:

- Fast query returns and event management
- Efficient calendar integration

Security:

- Password hashing
- Encrypted database
- Secure user interface

Platform:

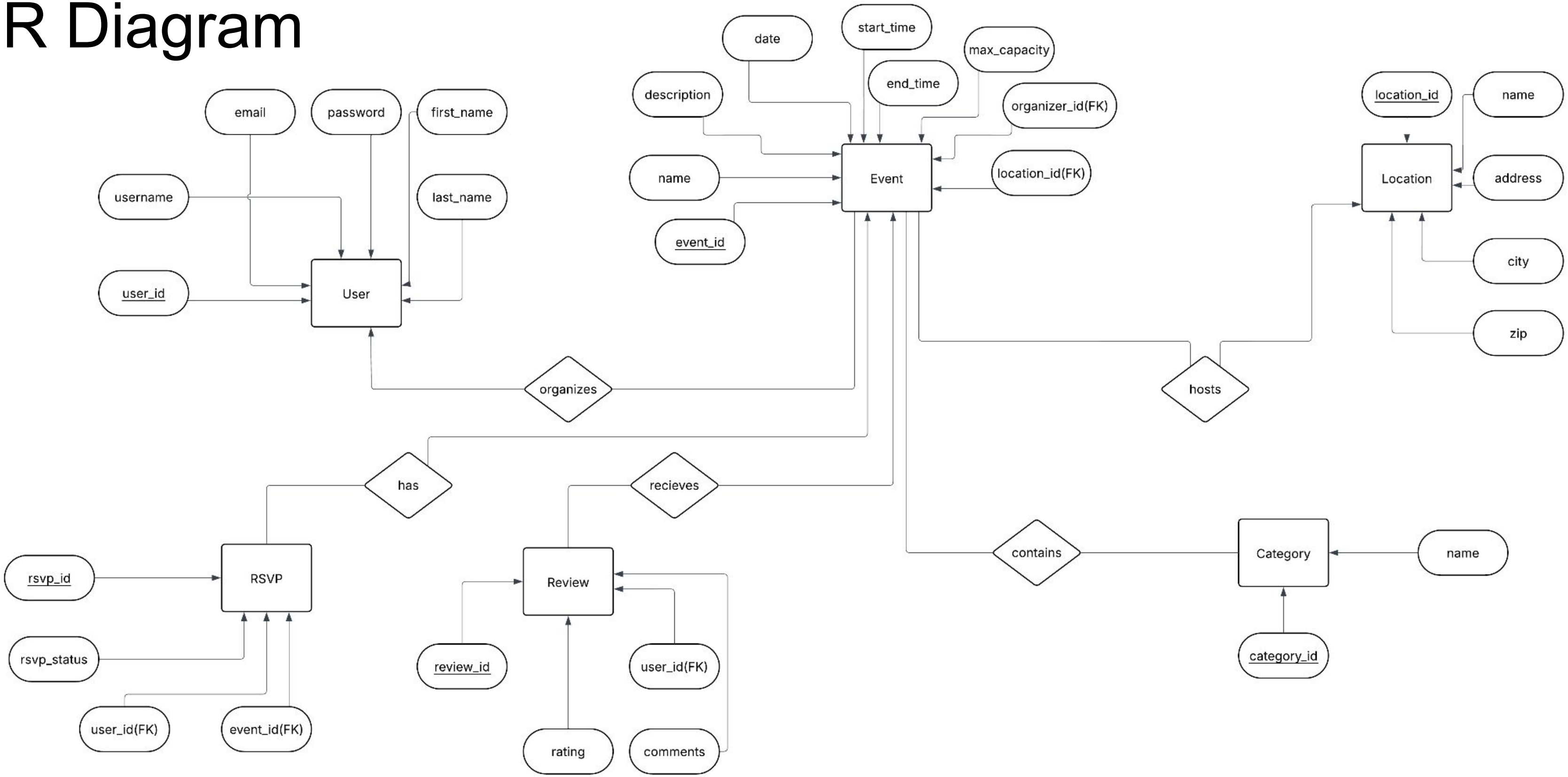
- Web-based interface for broad accessibility

Future Goals

Unimplemented or Unplanned Features

- User “Follow” System
- Automatic Calendar Integration
- Interest “Tags”
- More Advanced User Categories

ER Diagram


email
password
first_name
username
last_name
user_id
User
date
start_time
max_capacity
end_time
description
organizer_id(FK)
name
Event
location_id(FK)
event_id
location_id
name
Location
address
city
zip
organizes
hosts
has
recieves
contains
rsvp_id
RSVP
rsvp_status
user_id(FK)
event_id(FK)
Review
review_id
user_id(FK)
rating
comments
Category
name
category_id

Database structure(6 Entities)

User : People using the app(attendees and organisers)

Event : Meetups/gatherings with date, time, location

Location : Venues where events happen (reusable)

Category : Event types (Hiking, Food, Music, etc.)

RSVP : Tracks who's attending which events

Review : User feedback and ratings on events

Key Design Decisions

Location as Separate Entity:

-Same venues host multiple events

-Prevents duplicate data - Better for normalization

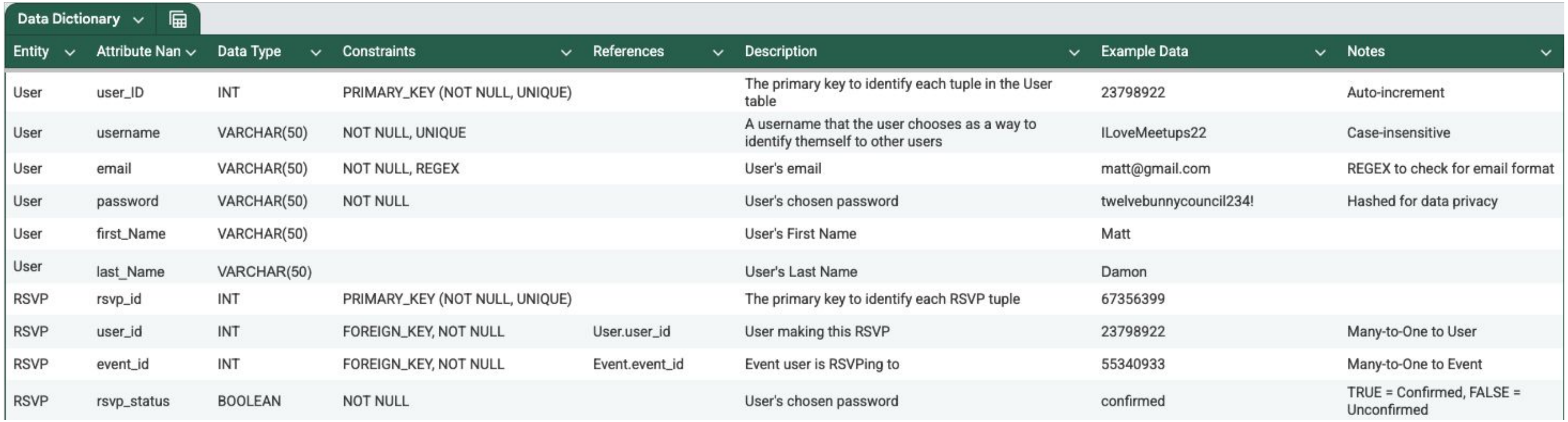
Category Entity:

-Enables event filtering by type

-Core feature for meetup discovery

Data Dictionary

The data dictionary provides a detailed description of all entities and their attributes, including their constraints, dependencies, and example data. Informed by the ER diagram, it provides clarity for all team members involved including clients and shareholders who may be interested in what sample data will look like, as well as direction for the future development phase.



Entity	Attribute Nan	Data Type	Constraints	References	Description	Example Data	Notes
User	user_ID	INT	PRIMARY_KEY (NOT NULL, UNIQUE)		The primary key to identify each tuple in the User table	23798922	Auto-increment
User	username	VARCHAR(50)	NOT NULL, UNIQUE		A username that the user chooses as a way to identify themself to other users	ILoveMeetups22	Case-insensitive
User	email	VARCHAR(50)	NOT NULL, REGEX		User's email	matt@gmail.com	REGEX to check for email format
User	password	VARCHAR(50)	NOT NULL		User's chosen password	twelvebunnycouncil234!	Hashed for data privacy
User	first_Name	VARCHAR(50)			User's First Name	Matt	
User	last_Name	VARCHAR(50)			User's Last Name	Damon	
RSVP	rsvp_id	INT	PRIMARY_KEY (NOT NULL, UNIQUE)		The primary key to identify each RSVP tuple	67356399	
RSVP	user_id	INT	FOREIGN_KEY, NOT NULL	User.user_id	User making this RSVP	23798922	Many-to-One to User
RSVP	event_id	INT	FOREIGN_KEY, NOT NULL	Event.event_id	Event user is RSVPing to	55340933	Many-to-One to Event
RSVP	rsvp_status	BOOLEAN	NOT NULL		User's chosen password	confirmed	TRUE = Confirmed, FALSE = Unconfirmed

Data Dictionary Considerations

- **Structure:** Our data dictionary lives in one table, with the entity each attribute belongs to in the rightmost column, and the attributes grouped by entity for ease of viewing.

- **Data Type:** Some of this was fairly intuitive but thought was given to the length of the varchar() type. We tried to maximize efficiency while also considering what is the reasonable maximum number of characters a user may want to input to this field.

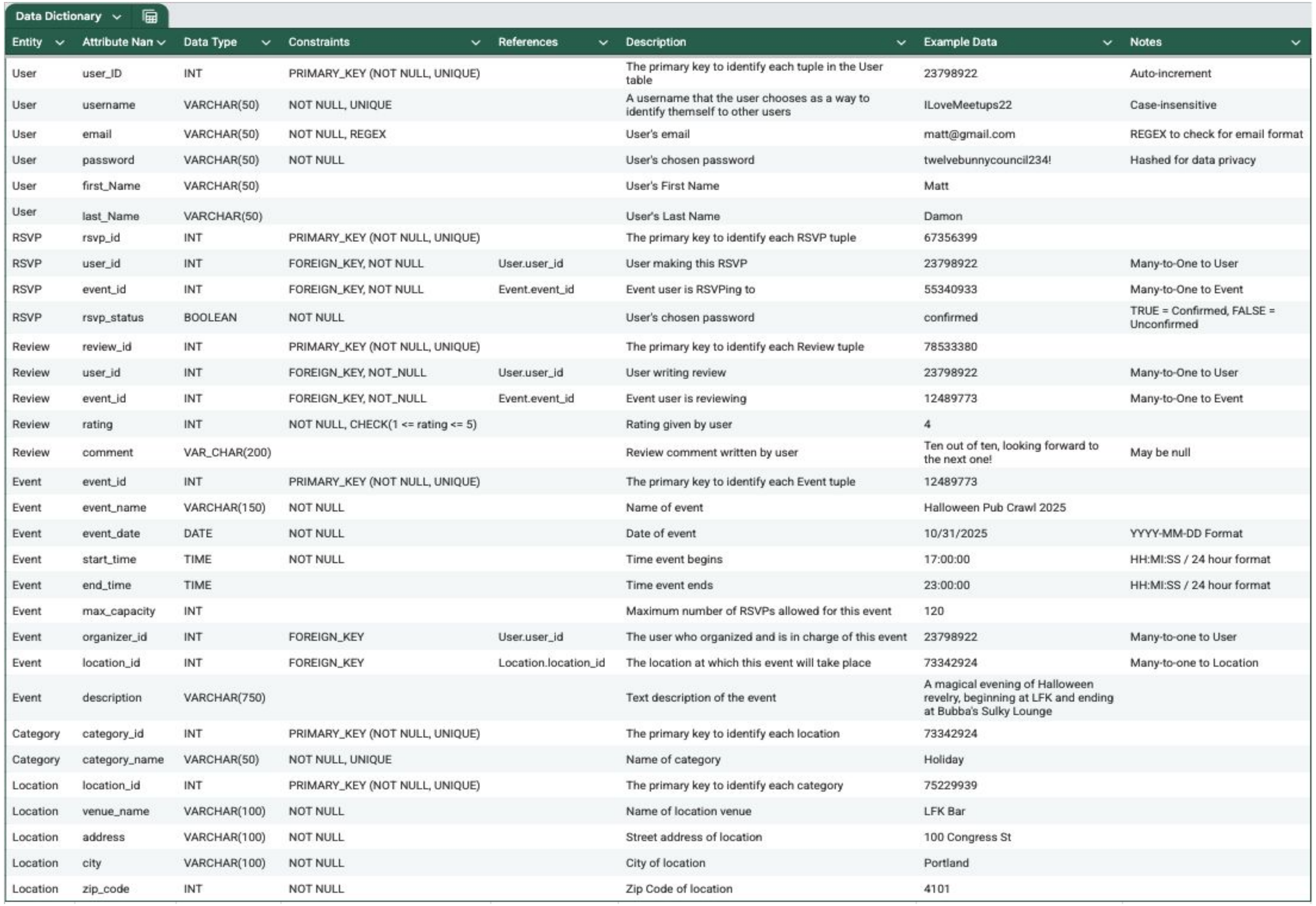
- **Constraints:** This indicates what constraints are put on each attribute, such as NOT NULL, UNIQUE, etc. Primary keys and foreign keys are indicated here, as well as a couple uses of specific constraints, such as REGEX to check that a user's input in the email field is in fact an email address, and checking whether the integer is between 1 and 5 for the number of stars on a review rating.

- **References:** The References column further clarifies what foreign entity each foreign key points to.

- More detailed information about each attribute is offered in the Description field, and Notes provides a space to include information that doesn't fit perfectly into the other fields

Full Data Dictionary

Links to Google Sheet

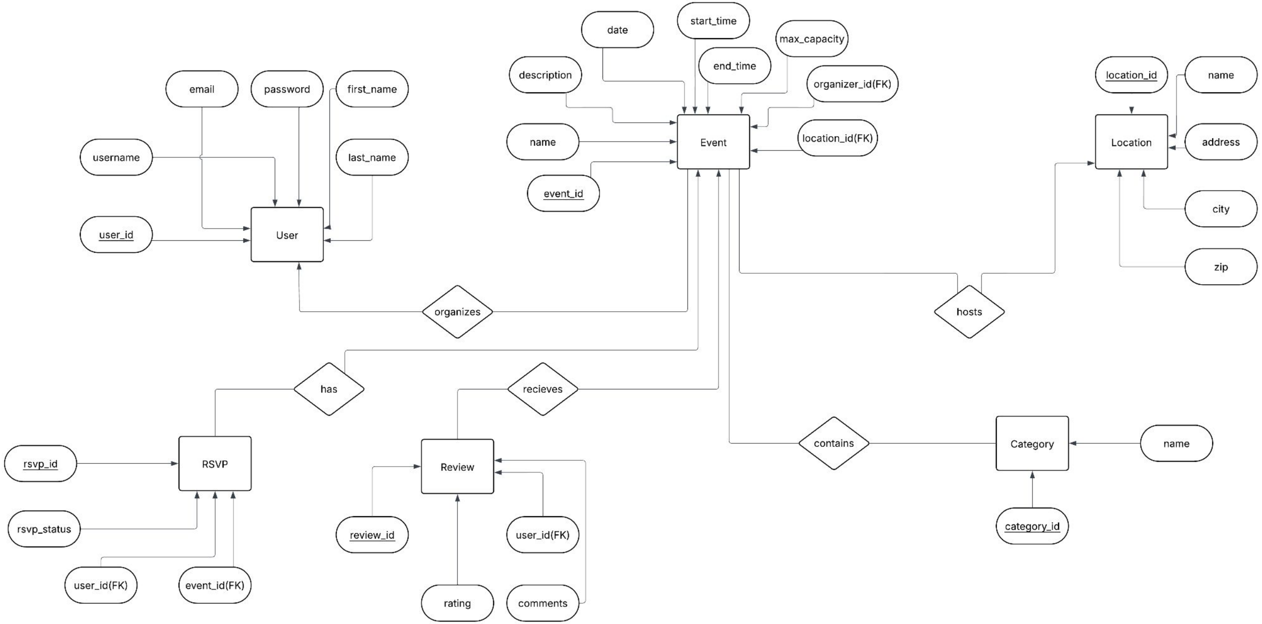


Entity	Attribute Nan	Data Type	Constraints	References	Description	Example Data	Notes
User	user_ID	INT	PRIMARY_KEY (NOT NULL, UNIQUE)		The primary key to identify each tuple in the User table	23798922	Auto-increment
User	username	VARCHAR(50)	NOT NULL, UNIQUE		A username that the user chooses as a way to identify themself to other users	ILoveMeetups22	Case-insensitive
User	email	VARCHAR(50)	NOT NULL, REGEX		User's email	matt@gmail.com	REGEX to check for email format
User	password	VARCHAR(50)	NOT NULL		User's chosen password	twelvebunnycouncil234!	Hashed for data privacy
User	first_Name	VARCHAR(50)			User's First Name	Matt	
User	last_Name	VARCHAR(50)			User's Last Name	Damon	
RSVP	rsvp_id	INT	PRIMARY_KEY (NOT NULL, UNIQUE)		The primary key to identify each RSVP tuple	67356399	
RSVP	user_id	INT	FOREIGN_KEY, NOT NULL	User.user_id	User making this RSVP	23798922	Many-to-One to User
RSVP	event_id	INT	FOREIGN_KEY, NOT NULL	Event.event_id	Event user is RSVPing to	55340933	Many-to-One to Event
RSVP	rsvp_status	BOOLEAN	NOT NULL		User's chosen password	confirmed	TRUE = Confirmed, FALSE = Unconfirmed
Review	review_id	INT	PRIMARY_KEY (NOT NULL, UNIQUE)		The primary key to identify each Review tuple	78533380	
Review	user_id	INT	FOREIGN_KEY, NOT_NULL	User.user_id	User writing review	23798922	Many-to-One to User
Review	event_id	INT	FOREIGN_KEY, NOT_NULL	Event.event_id	Event user is reviewing	12489773	Many-to-One to Event
Review	rating	INT	NOT NULL, CHECK(1 <= rating <= 5)		Rating given by user	4	
Review	comment	VAR_CHAR(200)			Review comment written by user	Ten out of ten, looking forward to the next one!	May be null
Event	event_id	INT	PRIMARY_KEY (NOT NULL, UNIQUE)		The primary key to identify each Event tuple	12489773	
Event	event_name	VARCHAR(150)	NOT NULL		Name of event	Halloween Pub Crawl 2025	
Event	event_date	DATE	NOT NULL		Date of event	10/31/2025	YYYY-MM-DD Format
Event	start_time	TIME	NOT NULL		Time event begins	17:00:00	HH:MI:SS / 24 hour format
Event	end_time	TIME			Time event ends	23:00:00	HH:MI:SS / 24 hour format
Event	max_capacity	INT			Maximum number of RSVPs allowed for this event	120	
Event	organizer_id	INT	FOREIGN_KEY	User.user_id	The user who organized and is in charge of this event	23798922	Many-to-one to User
Event	location_id	INT	FOREIGN_KEY	Location.location_id	The location at which this event will take place	73342924	Many-to-one to Location
Event	description	VARCHAR(750)			Text description of the event	A magical evening of Halloween revelry, beginning at LFK and ending at Bubba's Sulky Lounge	
Category	category_id	INT	PRIMARY_KEY (NOT NULL, UNIQUE)		The primary key to identify each location	73342924	
Category	category_name	VARCHAR(50)	NOT NULL, UNIQUE		Name of category	Holiday	
Location	location_id	INT	PRIMARY_KEY (NOT NULL, UNIQUE)		The primary key to identify each category	75229939	
Location	venue_name	VARCHAR(100)	NOT NULL		Name of location venue	LFK Bar	
Location	address	VARCHAR(100)	NOT NULL		Street address of location	100 Congress St	
Location	city	VARCHAR(100)	NOT NULL		City of location	Portland	
Location	zip_code	INT	NOT NULL		Zip Code of location	4101	

Normal Forms: First Normal Form

- Every attribute is atomic
- No duplicate rows
- Unique columns
- No order to data.

All of these are true for our DB.

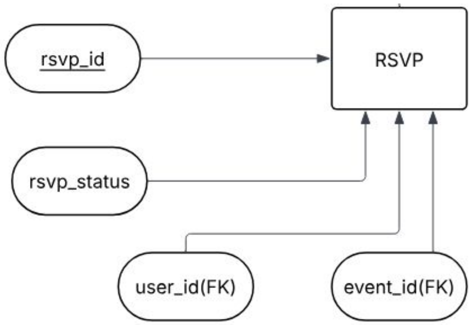


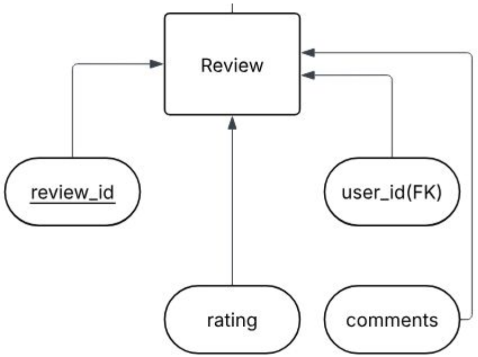
Normal Forms: Second Normal Form

- 1NF
- No partial dependencies.

Our database automatically satisfies 2NF as there are no composite primary keys.

In RSVP and Review, we technically could use composite keys of User/Event IDs and it would still pass. This was just our design choice, not really influenced much by normalization.



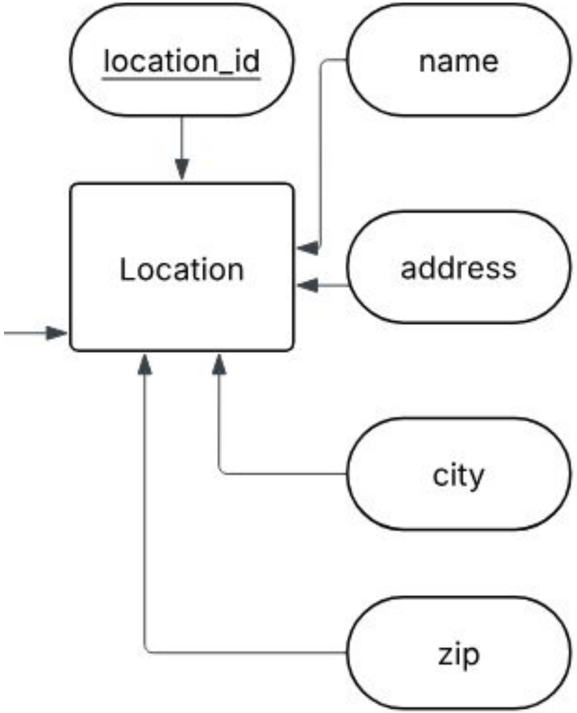


Normal Forms: Third Normal Form

- 2NF
- No transitive dependencies.

Every attribute in every entity relies solely on the primary key, and nothing is derived from other non-key attributes.

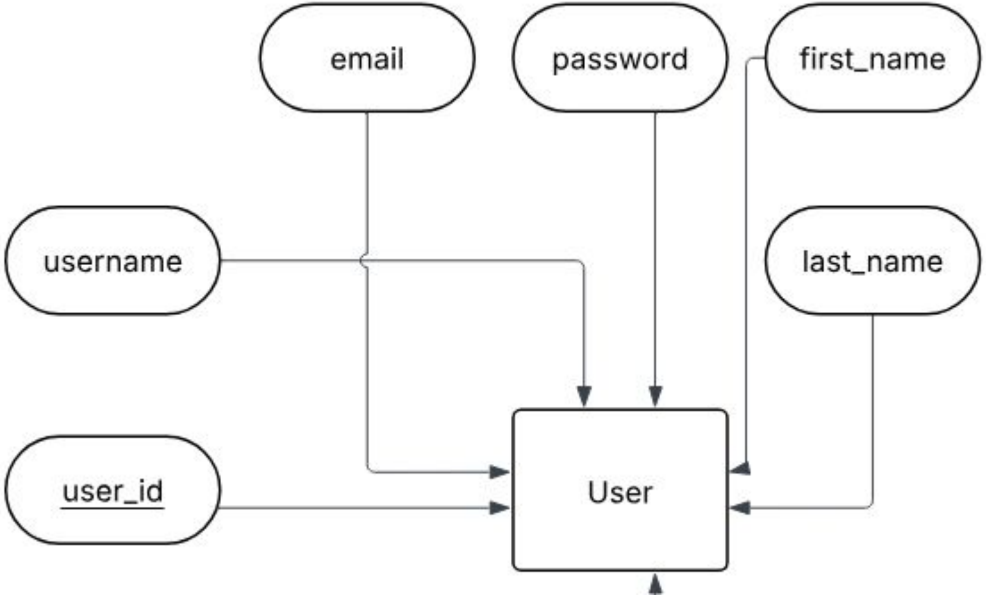
Could make the argument about city/zip depending on each other, however for our scope that's not necessary.



Normal Forms: BCNF

- Every non trivial FD needs to have $\underline{\alpha}$ as a superkey.

Similar to the discussion with 3NF, each entity only has one FD.
Entity_ID → all other attributes



Thank You