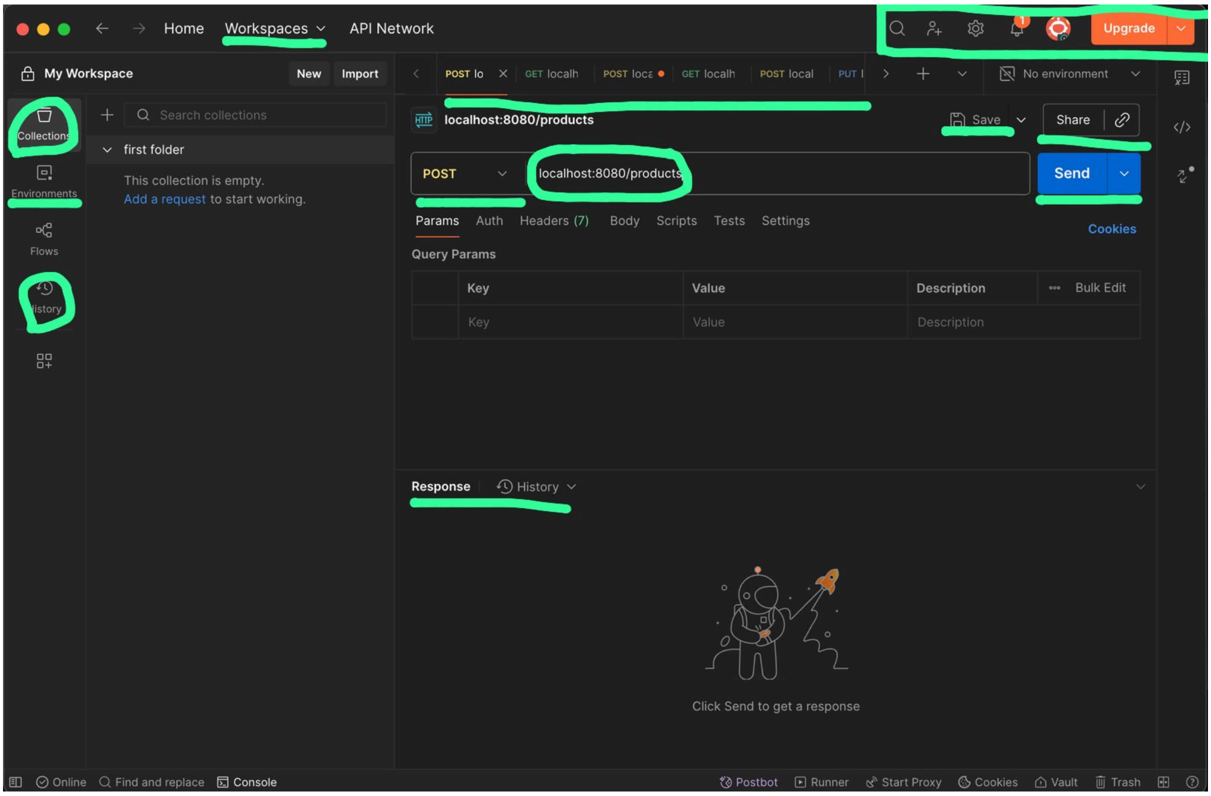
TASK NUMBER - 1

So the task is to create the design of a database for a postman-like application.We need to identify tables, columns, keys and constraints and then finally create an ER-diagram.
After a careful exploration of postman application and with some help through stackoverflow and google i have come up with this information.



POST localhost:8080/products

This collection is empty.
Add a request to start working.

Click Send to get a response

I have also highlighted so of the aspects of my information.

Tables

1. User
2. Workspace
3. Role
4. userworkspacerole(mapping table)
5. Environment
6. Collections
7. Requests
8. Request history

USER

Columns	Type	Constraints
user_id	int	PK, auto_increment
username	varchar	Unique, not null
email	varchar	Unique, not null
password	varchar	Not null

User_id is primary key
Email and username must be unique and not null
This table is a list of all the people who use the application.
It helps us keep track of who owns what and who can use the app.

WORKSPACE

Columns	Type	constrains
workspace_id	int	PK, auto_increment
name	varchar	Unique, not null

Workspace_id is primary key
Work space is kind of a virtual room where we can share our work together and do amends
users can organize their collections, requests, and environments in teams or projects.

ROLE

Columns	Type	Constraints
role_id	int	PK, auto_increment
role_name	varchar	Not null, unique

Here role_id is primary key
This table shows all possible number of roles like admin, member
It helps us to know what kind of authority each user have.

USERWORKSPACEROLE (MAPPING TABLE**)**

Columns	Types	Constraints
uwr_id	int	PK
user_id	int	FK to Users(user_id), NOT NULL
workspace_id	int	FK to workspace(workspace_id), NOT NULL
role_id	int	FK to Role(role_id), NOT NULL

Uwr_id is primary key
This table will connect user column to workspace column and assign the roles.
Because it is going to be a many to one relation or one to many like one user can have multiple workspace

ENVIRONMENT

Columns	Types	Constraints
environment_id	Int	PK, auto_increment
name	varchar	Not null
variables_json	TEXT	
workspace_id	Int	FK to workspace(workspace_id), NOT NULL

COLLECTION

Columns	Types	Constraints
collection_id	Int	PK, auto_increment
name	varchar	Not null
user_id	Int	FK to Users(user_id), NOT NULL
workspace_id	Int	FK to workspace(workspace_id), NOT NULL

Here collection_id is primary key
It is simply a folder which holds all our api requests which is located inside the workspace

It is firstly connected to user its original creator and then to the workspace

REQUESTS

Columns	Types	Constraints
request_id	int	PK, auto_increment
method	varchar	Not null
url	varchar	Not null
header_json	text	
bod_jason	text	
generated_at	datetime	Not null

Here request_id is the primary key
It is the most important table for the user to fetch the data so that it can be verified
It have multiple methods to perform like get, post, put, delete, patch
Every request is in a collection so if we want we can also use collection in this table as a foreign key

REQUESTHISTORY

Columns	Types	Constraints
history_id	int	PK, auto_increment
user_id	int	FK to Users(user_id), NOT NULL
request_id	int	FK to

		request(request_id), NOT NULL
generated_at	datetime	Not null

Here history_is is the primary key
Have taken user_id and request_id as the foreign key to show that just how many times a user is generating a request
This table will be some what like logs to store request for repeated use without doing it manually every time

TABLE RELATIONSHIPS

Users and Collections: One-to-many
Users and Workspaces and Roles: Many-to-many via UserWorkspaceRoles
Collections and Requests: One-to-many
Requests and RequestHistory: One-to-many
Workspaces and Collections: One-to-many

CONSTRAINTS

All of the id's are primary key and are set for auto increment because there can be multiple requests
All the foreign key represent relation to their main table
Unique and not null are also used to reduce data redundancy