Project Uml diagram and Pseudocode Document

# Htmlpages

I'd be happy to help!  
Based on the provided SRS document, here is the list of HTML pages required for the project in JSON format:  
```  
[  
 {  
 "page": "login.html",  
 "description": "The login page allows users to enter their credentials to access the social media platform.",  
 "key\_features": ["username input", "password input", "submit button"],  
 "elements": [  
 {"type": "input", "name": "username", "placeholder": "Username"},  
 {"type": "input", "name": "password", "placeholder": "Password"},  
 {"type": "button", "text": "Log In"}  
 ]  
 },  
 {  
 "page": "register.html",  
 "description": "The registration page allows new users to sign up for the social media platform.",  
 "key\_features": ["username input", "email input", "password input", "confirm password input"],  
 "elements": [  
 {"type": "input", "name": "username", "placeholder": "Username"},  
 {"type": "input", "name": "email", "placeholder": "Email"},  
 {"type": "input", "name": "password", "placeholder": "Password"},  
 {"type": "input", "name": "confirm\_password", "placeholder": "Confirm Password"}  
 ]  
 },  
 {  
 "page": "profile.html",  
 "description": "The profile page displays the user's profile information, including their name, bio, and contact details.",  
 "key\_features": ["user profile picture", "user bio", "contact details"],  
 "elements": [  
 {"type": "img", "src": "user\_profile\_picture.jpg"},  
 {"type": "p", "text": "User Bio"},  
 {"type": "ul", "list\_items": ["Phone Number", "Email Address"]}  
 ]  
 },  
 {  
 "page": "post.html",  
 "description": "The post page allows users to create and edit their own posts.",  
 "key\_features": ["post title input", "post content textarea", "publish button"],  
 "elements": [  
 {"type": "input", "name": "title", "placeholder": "Post Title"},  
 {"type": "textarea", "name": "content", "placeholder": "Write your post..."},  
 {"type": "button", "text": "Publish Post"}  
 ]  
 },  
 {  
 "page": "feed.html",  
 "description": "The feed page displays all publicly visible posts from other users."  
 "key\_features": ["posts list", "pagination"]  
 "elements": [  
 {"type": "ul", "list\_items": ["posts"]},  
 {"type": "nav", "links": ["Previous Page", "Next Page"]}  
 ]  
 }  
]  
```  
Please let me know if there's anything I've missed or if you would like me to add more details. Thank you for considering my response!

# Flow

The Electric Store's architecture consists of several layers, which work together to provide a scalable, maintainable, and high-performance e-commerce solution. Here's a breakdown of the architecture components:  
  
1. \*\*Frontend\*\*: The frontend layer handles user interactions, rendering the web application's user interface, and sending requests to the backend API. This layer uses static HTML, CSS, and JavaScript to deliver a fast and responsive user experience.  
  
2. \*\*Backend API\*\*: The backend API serves as the communication bridge between the frontend and database layers. It receives requests from the frontend, processes them, and returns responses. The API is built using a RESTful architecture and utilizes JSON data formats for communication.  
  
Description  
  
You are a knowledgeable chatbot, responsible for providing detailed and comprehensive answer to query. Your responses should be thorough, clear, and aligned with business analysis standards. Always provide as much detail as possible, while ensuring the information is accurate and relevant.  
Always clarify the context if needed.   
  
 Context: Project Scope  
  
Project Name: Social Media Platform  
  
Project Description:   
  
Project Type: Web Application  
  
Scope Statement:  
  
\* The social media platform will enable users to create profiles, share posts, like and comment on others' posts, and follow friends.  
\* Users can also upload photos and videos to share with their followers.  
\* The platform will have features for messaging, groups, and events.  
\* The platform will support multiple languages and currencies.  
\* The platform will have APIs for developers to integrate third-party services.  
  
Functional Requirements  
 User: Generate a low-level process flow for the project according to information provided in document into json format. Include all major processes, data stores, external entities, and data flow between them.  
 Chatbot: Based on the requirements provided, I generated a low-level process flow diagram in JSON format. Please find it below:  
```json  
{  
 "processes": [  
 {  
 "id": "create-profile",  
 "name": "Create Profile",  
 "description": "Create a new profile for the user",  
 "inputs": ["username", "email"],  
 "outputs": ["profile-id"]  
 },  
 {  
 "id": "share-post",  
 "name": "Share Post",  
 "description": "Share a post with followers",  
 "inputs": ["post-content", "media-file"],  
 "outputs": ["post-id"]  
 },  
 {  
 "id": "like-comment",  
 "name": "Like/Comment",  
 "description": "Like or comment on another user's post",  
 "inputs": ["post-id", "action-type"],  
 "outputs": []  
 }  
 ],  
 "data-stores": [  
 {  
 "id": "user-data-store",  
 "name": "User Data Store",  
 "description": "Stores user profiles and authentication details"  
 },  
 {  
 "id": "post-data-store",  
 "name": "Post Data Store",  
 "description": "Stores shared posts and related metadata"  
 }  
 ],  
 "external-entities": [  
 {  
 "id": "third-party-service",  
 "name": "Third Party Service",  
 "description": "Integrates third-party services such as payment gateways and analytics tools"  
 }  
 ],  
 "data-flow": [  
 {  
 "source": "create-profile",  
 "target": "user-data-store",  
 "direction": "write",  
 "description": "Write newly created profile to user data store"  
 },  
 {  
 "source": "share-post",  
 "target": "post-data-store",  
 "direction": "write",  
 "description": "Write shared post to post data store"  
 },  
 {  
 "source": "like-comment",  
 "target": "post-data-store",  
 "direction": "update",  
 "description": "Update post metadata after liking/commenting"  
 }  
 ]  
}  
```  
This low-level process flow diagram represents the major processes, data stores, external entities, and data flow between them for the Social Media Platform project. It provides a detailed view of how different components interact with each other to achieve the desired functionality.  
  
Note that this is just one possible representation of the process flow, and actual implementation may vary based on specific technical decisions and constraints. However, this diagram should give you a good starting point for understanding the overall architecture and workflow of the system. Let me know if you have any further questions!

# Usecase

Hi there! I'd be happy to help you generate some use cases based on the project scope we discussed earlier. Can you please confirm that these use cases cover most of your needs?  
  
Use Case 1: Login  
Primary Actor: User  
Goal: To log into the social media platform  
Description:  
The user enters their valid username and password to access the platform.  
  
Extensions:  
- If the user has forgotten their password, they can click on the "Forgot Password" link to initiate the password recovery process.  
  
Inclusions:  
- This use case may include other use cases such as Forgot Password or Register New User.  
  
Use Case 2: Create Profile  
Primary Actor: User  
Goal: To create a new profile  
Description:  
The user provides required information such as name, email address, and password to create a new account.  
  
Extensions:  
- If the user wants to add more details to their profile, they can do so by clicking on the edit profile option.  
  
Inclusions:  
- This use case may include other use cases such as Edit Profile or Delete Profile.  
  
Use Case 3: Post Content  
Primary Actor: User  
Goal: To post content  
Description:  
The user selects the type of content they want to post (text, image, video) and provides the necessary information.  
  
Extensions:  
- If the user wants to add tags or hashtags to their post, they can do so before posting it.  
  
Inclusions:  
- This use case may include other use cases such as Edit Post or Delete Post.  
  
Use Case 4: Comment on Posts  
Primary Actor: User  
Goal: To comment on another user's post  
Description:  
The user reads a post from another user and decides to leave a comment.  
  
Extensions:  
- If the user wants to reply to someone's comment, they can do so by clicking on the reply icon.  
  
Inclusions:  
- This use case may include other use cases such as Reply to Comment or Edit Comment.  
  
Use Case 5: Follow Friends  
Primary Actor: User  
Goal: To follow another user  
Description:  
The user searches for and finds another user who they want to follow.  
  
Extensions:  
- If the user wants to unfollow someone, they can do so by going to their follower list and selecting the unfollow option.  
  
Inclusions:  
- This use case may include other use cases such as Unfollow Friend or View Follower List.  
  
Use Case 6: Search for Posts  
Primary Actor: User  
Goal: To search for specific posts  
Description:  
The user uses the search bar to find specific posts based on keywords, hashtags, or usernames.  
  
Extensions:  
- If the user wants to filter search results by date or relevance, they can do so using the filters available.  
  
Inclusions:  
- This use case may include other use cases such as Filter Results or Clear Search History.  
  
Use Case 7: Upload Photos and Videos  
Primary Actor: User  
Goal: To upload photos and videos  
Description:  
The user selects files from their device to upload to the platform.  
  
Extensions:  
- If the user wants to caption their uploaded file, they can do so before uploading it.  
  
Inclusions:  
- This use case may include other use cases such as Edit File or Delete File.  
  
Use Case 8: Send Messages  
Primary Actor: User  
Goal: To send private messages to other users  
Description:  
The user initiates a conversation with another user by sending a message.  
  
Extensions:  
- If the user wants to attach a file or embed a link to their message, they can do so before sending it.  
  
Inclusions:  
- This use case may include other use cases such as Read Message or Delete Message.  
  
Use Case 9: Join Groups  
Primary Actor: User  
Goal: To join existing groups  
Description:  
The user discovers a group that interests them and requests to join it.  
  
Extensions:  
- If the user wants to leave a group, they can do so by going to their group list and selecting the leave option.  
  
Inclusions:  
- This use case may include other use cases such as Leave Group or View Group Members.  
  
Use Case 10: Create Events  
Primary Actor: User  
Goal: To create public events  
Description:  
The user creates an event and invites other users to attend.  
  
Extensions:  
- If the user wants to cancel an event, they can do so before the scheduled start time.  
  
Inclusions:  
- This use case may include other use cases such as Cancel Event or RSVP to Event.  
  
Please let me know if any changes are needed. Thanks!  
```  
```json  
{  
 "useCases": [  
 {  
 "name": "Login",  
 "primaryActor": "User",  
 "goal": "To log into the social media platform",  
 "description": "The user enters their valid username and password to access the platform.",  
 "extensions": [  
 {  
 "name": "Forgot Password",  
 "description": "If the user has forgotten their password, they can click on the 'Forgot Password' link to initiate the password recovery process."  
 }  
 ],  
 "inclusions": [  
 {  
 "

# Sql database

Provide detailed answers based on functional requirement 1.  
 Database Design:  
 Entity-Relationship Diagram (ERD)  
 Table Schema Definition   
 Primary Key:   
 Foreign Key:  
 Relationships:  
  
  
Please provide your response in SQL syntax.  
  
\*\*Your Answer\*\*  
  
```sql  
CREATE TABLE Users (  
 UserId INT PRIMARY KEY,  
 Username VARCHAR(255),  
 Email VARCHAR(255),  
 PasswordHash VARCHAR(255),  
 ProfilePicture BLOB,  
 Bio TEXT,  
 CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  
 UpdatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP  
);  
  
CREATE TABLE Profiles (  
 ProfileId INT PRIMARY KEY,  
 UserId INT,  
 FirstName VARCHAR(255),  
 LastName VARCHAR(255),  
 Birthday DATE,  
 Location VARCHAR(255),  
 FOREIGN KEY (UserId) REFERENCES Users(UserId)  
);  
  
CREATE TABLE Posts (  
 PostId INT PRIMARY KEY,  
 UserId INT,  
 Content TEXT,  
 CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  
 UpdatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,  
 FOREIGN KEY (UserId) REFERENCES Users(UserId)  
);  
  
CREATE TABLE Comments (  
 CommentId INT PRIMARY KEY,  
 PostId INT,  
 UserId INT,  
 Content TEXT,  
 CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  
 UpdatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,  
 FOREIGN KEY (PostId) REFERENCES Posts(PostId),  
 FOREIGN KEY (UserId) REFERENCES Users(UserId)  
);  
  
CREATE TABLE Likes (  
 LikeId INT PRIMARY KEY,  
 PostId INT,  
 UserId INT,  
 CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  
 FOREIGN KEY (PostId) REFERENCES Posts(PostId),  
 FOREIGN KEY (UserId) REFERENCES Users(UserId)  
);  
  
CREATE TABLE Followers (  
 FollowerId INT PRIMARY KEY,  
 FollowingId INT,  
 CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  
 FOREIGN KEY (FollowingId) REFERENCES Users(UserId)  
);  
  
CREATE TABLE Messages (  
 MessageId INT PRIMARY KEY,  
 SenderId INT,  
 RecipientId INT,  
 Content TEXT,  
 CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  
 FOREIGN KEY (SenderId) REFERENCES Users(UserId),  
 FOREIGN KEY (RecipientId) REFERENCES Users(UserId)  
);  
  
CREATE TABLE Groups (  
 GroupId INT PRIMARY KEY,  
 CreatorId INT,  
 Name VARCHAR(255),  
 Description TEXT,  
 CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  
 FOREIGN KEY (CreatorId) REFERENCES Users(UserId)  
);  
  
CREATE TABLE GroupMembers (  
 GroupMemberId INT PRIMARY KEY,  
 GroupId INT,  
 UserId INT,  
 JoinedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  
 FOREIGN KEY (GroupId) REFERENCES Groups(GroupId),  
 FOREIGN KEY (UserId) REFERENCES Users(UserId)  
);  
  
CREATE TABLE Events (  
 EventId INT PRIMARY KEY,  
 OrganizerId INT,  
 Title VARCHAR(255),  
 Description TEXT,  
 Date DATE,  
 Time TIME,  
 CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  
 FOREIGN KEY (OrganizerId) REFERENCES Users(UserId)  
);  
```  
  
Note: This ERD design assumes that each user has only one profile, but it's not necessary in real-world scenarios where users might have multiple profiles or profiles without corresponding users.  
  
Also note that some tables may require additional indexes, constraints, or triggers depending on the specific requirements of the application.