1. **How can we design the system in a way that every company will be able to serve games on their gaming site from their domain?**

To design a system where each company can serve games from their own domain, we should adopt a multi-tenant architecture. Each tenant (company) will have its own subdomain or custom domain, and the system should route requests to the appropriate tenant's resources. This can be achieved using the following methods:

* **Domain Mapping**: Configure the server to recognize custom domains and map them to the correct tenant. This can be done using DNS settings and web server configurations (e.g., NGINX or Apache virtual hosts).
* **Tenant Identification**: Use the HTTP request's Host header to identify the tenant. This requires modifying the application to parse the domain and determine which tenant the request is for.
* **Separate Data Stores**: While optional, separating the databases or using schema-based multi-tenancy can enhance security and data management.

1. **What modification should be done to the users table at gPlatform to support this change?**

To support multiple tenants, the users table should be modified to include a reference to the tenant (company). This can be done by adding a tenant\_id column, which links each user to the company they belong to. Here's a possible modification:

ALTER TABLE users ADD COLUMN tenant\_id INT NOT NULL;

The tenant\_id would be a foreign key referencing a tenants table that stores information about each tenant (company).

**Example tenants table:**

CREATE TABLE tenants (

tenant\_id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(255) NOT NULL,

domain VARCHAR(255) NOT NULL UNIQUE

);

1. **Considering we have 1 backend cluster that serves all companies, how can we validate a user login on one gaming domain in such a way that it does not give access to a different gaming domain?**

To ensure that user authentication is domain-specific, the following steps should be taken:

* **Tenant-Aware Authentication**: When a user attempts to log in, the authentication process should include the tenant\_id in the validation. This means that the login credentials (e.g., email and password) must be checked in conjunction with the tenant\_id.
* **Session Management**: Ensure that session tokens or JSON Web Tokens include the tenant\_id as part of the payload. This ensures that any authenticated session is tied to a specific tenant.

**Key Points:**

* **Tenant Resolution**: Use the domain from the HTTP request to resolve the tenant\_id.
* **Scoped Queries**: Ensure all user-related queries are scoped to the tenant\_id to prevent cross-tenant data leakage.
* **Token Scoping**: Include tenant\_id in authentication tokens to enforce tenant-specific access control.

By implementing these strategies, we can ensure that each company's users can only access their respective gaming site, maintaining isolation and security across tenants.