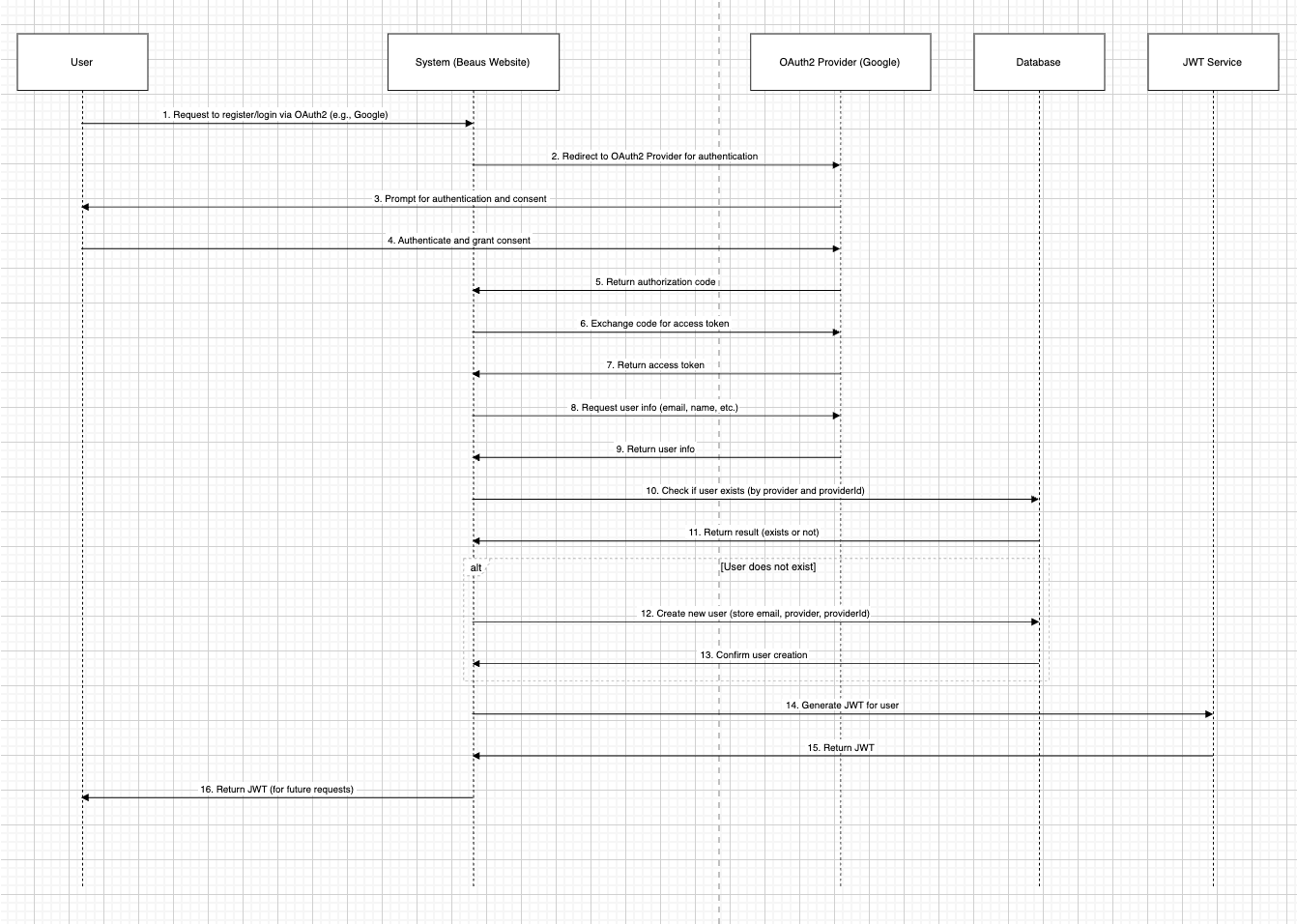
**Sequence Diagram for User Registration Flow with OAuth2 and JWT**



**Detailed Explanation (in English)**

This sequence diagram illustrates the user registration flow for the "Beaus Website" system, integrating OAuth2 for authentication and JWT for subsequent user authorization. The process involves the following key components:

* **User (U):** The individual attempting to register or log into the system.
* **System (S):** The "Beaus Website" application that handles registration and authentication requests.
* **OAuth2 Provider (O):** An external OAuth2 provider (e.g., Google) used for user authentication.
* **Database (D):** The database where user information is stored.
* **JWT Service (J):** A service responsible for generating JSON Web Tokens (JWTs) for user authentication in future requests.

The diagram captures the interaction between these components during the registration process, starting from the user initiating a login via OAuth2, through the authentication with the OAuth2 provider, to the creation of a new user in the system (if necessary), and finally issuing a JWT for the user to use in future API requests.

**Detailed Flow (in English)**

1. **User Initiates Registration/Login:** The user requests to register or log in using an OAuth2 provider (e.g., by clicking "Login with Google").
2. **System Redirects to OAuth2 Provider:** The system redirects the user to the OAuth2 provider's authentication page.
3. **OAuth2 Provider Prompts for Authentication:** The OAuth2 provider prompts the user to authenticate (e.g., by entering their Google credentials) and grant consent for the system to access their information (e.g., email, name).
4. **User Authenticates and Grants Consent:** The user successfully authenticates with the OAuth2 provider and grants the necessary permissions.
5. **OAuth2 Provider Returns Authorization Code:** The OAuth2 provider sends an authorization code back to the system as part of the redirect URI.
6. **System Exchanges Code for Access Token:** The system sends the authorization code to the OAuth2 provider to request an access token.
7. **OAuth2 Provider Returns Access Token:** The OAuth2 provider responds with an access token, which the system can use to access the user's information.
8. **System Requests User Info:** Using the access token, the system requests the user's information (e.g., email, name) from the OAuth2 provider.
9. **OAuth2 Provider Returns User Info:** The OAuth2 provider returns the requested user information to the system.
10. **System Checks for Existing User:** The system queries the database to check if a user with the same provider (e.g., Google) and providerId (unique identifier from the OAuth2 provider) already exists.
11. **Database Returns Result:** The database responds with whether the user exists or not.
12. **Create New User (if not exists):** If the user does not exist, the system creates a new user in the database, storing details such as email, provider, and providerId.
13. **Database Confirms Creation:** The database confirms that the new user has been successfully created.
14. **System Requests JWT Generation:** The system sends a request to the JWT Service to generate a JWT for the user.
15. **JWT Service Returns JWT:** The JWT Service generates and returns a JWT containing the user's information (e.g., user ID, roles).
16. **System Returns JWT to User:** The system sends the JWT back to the user, which the user can use in the Authorization header for future API requests (e.g., Authorization: Bearer <token>).