1)-The user logins to some web application (which is client application) and tries to use or get some resources (for example upload a photo). The client application gives an option to upload the photos from google photos application, and that option the user chooses. Then the client application sends the request to the authorization server on behalf of the user. In that request client application sends very important information to the authorization server. It is a **scope** - it means the type of access that is required by client application on behalf of the user. In that example it might ask for access to view and download the photos from google photos app. In other words, a client application with the scope will inform the authorization server what it needs. Then authorization server requests for authorization from a user (username : password), after the authorization is succeeded server asks the user to provide permission for the client application, while doing so the server will show to user what exactly the client application wants to do (kind of permission that is requested). And also the authorization server will ask the user whether the user trusts the client application. If the user approves, then authorization server sends to the client application auth code. The auth code is a short lived token that can be used by the client application. NOTICE: all that flow happening on a front end side (channel), that is browser.

Once the client application gets the auth code, the back end channel of the client application (the back end server), it will send the auth code back to the authorization server. The authorization server will validate the auth code and if it is valid only then it will send back **access token** (might be refresh token) back to the client application (the back end server of it). Access token is short time lived token and it will have expiry time (15 -30min). If it is refresh token it remains valid until the user is logged in on the browser. Once the access token is expired the client application can use refresh token, to renew the access token as long as the refresh token is valid.

If the user logged out of google photo application, all the process will be repeated. Once the backend server gets the access token from an authorized server it can send a token to google photo application to request for the photo (resource). And the backend server will send the token as part of the API call. The google photo app. will validate the access token if it is valid it will send back requested resources (in our example - photos). So with this the client application has got access to the protected resources that are the photo. And it gets access on behalf of the resource owner, because the user (resource owner) authorized the client application to access the photos.

Important note: auth code, access token, refresh token ← all this information is transferred on the back end, this is not exposed to everyone - it is secure. That is why the authorization grant flow is the most secure way of delegated authorization. Usually this workflow is used for the web server application which has frontend and backend.