**1. def home(request)**   
  
- This function provides the home page screen.  
  
- It does make sure that the user is on an authorization process.  
  
- If authenticated:  
  
- It creates an unknown Meep Form object which could be POST-ed.  
  
- If the request method is POST and the form is valid:- If the request method is POST and the form is valid:  
  
- It is followed to save form data which is used later on to construct a new Meep object yet remains associated with the current user.  
  
- It does so and sends "Success!" as the message.  
  
- It directs the user to the home page once again.  
  
- It will pull all Meep objects from the database and organize them according to their creation time.  
  
- If not authenticated:  
  
- Hence, it recursively sorts all Meep items according to their creation timestamp.  
  
- It calls the home.html template, delivering Meeps(tm) objects and forms into the template.  
  
**2. def profile\_list(request):**  
  
- This infrastructure displays the profile page that includes all the profiles.  
  
- If the user is authenticated:- If the user is authenticated:  
  
- It runs a search for all Profile objects of a user other than the currently authenticated user's profile.  
  
- It proposes the 'profile\_list.html' template with the profile going to the template as a variable.  
  
- If not authenticated:  
  
- This action contacts the user to return to the home page and displays a message of logging in.

**3. def unfollow(request, pk):**  
  
- This function deals with the action of removing a profile.  
  
undefined  
  
- It gets the profile to unfollow using the primary key.  
  
- It deletes the profile from the follows of the logged-on user's profile and saves it.  
  
- It gives you a success message.  
  
- It directs the user to the previous page once again.  
  
- If not authenticated:  
  
- It leads to the home page with a message stating that the user must log in.  
  
**4. def follow(request, pk):**  
  
- This function allows for the action of following a profile.  
  
undefined  
  
- It finds the profile that should be followed through the primary key.  
  
- It adds that to the following of the currently authenticated user's profile and saves it.  
  
- It displays the success message.  
  
- It takes the user to the previous page.  
  
- If not authenticated:  
  
- It redirects the user to the main page with a message about the necessity to log in.

**5. def profile(request, pk):**  
  
- This function has the responsibility of showcasing a user's profile page instead.  
  
- Initially, it sees whether the user is authorized.  
  
- If authenticated:  
  
- It yields the Profile object tied to the aforementioned user ID (pk).  
  
- It searches for the user's Meep objects and sorts them by their creation time.  
  
undefined  
  
- It retrieves the current user's profile.  
  
- It gets the form data from the request.  
  
- It checks if such a command conforms to the form data.  
  
- It edits the friendships registry in the current user profile and also saves it.  
  
- It renders the template jhhhhhhhhhh and the profile and Meepss are passed into it.  
  
- If not authenticated:  
  
- It posts there a link to log in.  
  
- It gets the user to the home page.  
  
**6. def followers(request, pk)**  
  
- This role enables the showing of the followers of a single user.  
  
undefined  
  
- It gets the Profile object using the specified user ID (pk).  
  
- It is templating the 'followers.html', and passing profiles to the template.  
  
- This page will tell you that it is not your profile page if that's not the case.  
  
- It switches to the home page.  
  
- If not authenticated:  
  
- It puts in a note that says to log in to continue.  
  
- It moves you to the homepage.

**7. def follows(request, pk):**  
  
- This function exhibits the following of other users that the user itself does.  
  
- It is a kind of 'following function' but, shows profiles that the user follows in a similar structure.  
  
**8. def login\_user(request):**  
  
- This function covers user logic.  
  
undefined  
  
- It acquires the login data from the POST request.  
  
- It verifies the user identity with the entered credentials.  
  
- Successful authentication allows the user to log in and outputs a success message.  
  
- If the authentication fails it retoces an error message.  
  
- When the request is another method other than POST, it returns the 'login.html' template.  
  
**9. def logout\_user(request):**  
  
- This module takes out the user.  
  
- It invokes the `logout()` function from Django's built-in authentication system, sending an arbitrary request.  
  
- It follows with a success message saying that the user has logged out successfully.  
  
- It takes the user to the home page, instead.

**10. def register\_user(request):**  
  
- For this task, the function of user registration is taken over.  
  
- It retrieves the SignUpForm instance.  
  
undefined  
  
- It is for SignUpForm class creation with POST data.  
  
undefined  
  
- It serves the purpose of storing the data for new user creation.  
  
- It gets the credentials from the form.  
  
- But it sure does authenticate the user.  
  
- it logs the user in.  
  
- It finishes the registration as a success message to indicate it.  
  
- It brings about redirection to the home page.  
  
- It gives the render of '/register.html', the form is in the template as a parameter.  
  
**11. def update\_user(request):**  
  
- Here, the function of updating user info.  
  
undefined  
  
- This one gets the current user and profile information by their respective numbers.  
  
- It begins with an instantiation of SignUpForm and ProfilePicForm with POST data or None, using the current user info if available.  
  
undefined  
  
- It saves updated lately user and profile details  
  
- It insidiously reaches the user.  
  
- The application additionally has a success message that you can see when updating your profile.  
  
- It sends the user to the home page directly.  
  
- It loads 'update\_user.html', and sends the forms to the view.  
  
undefined  
  
- An information sign is added to request a login.  
  
- It redirects to the home page.

**12. def meep\_like(request, pk):**  
  
- This one is responsible for liking/unliking a Meep.  
  
undefined  
  
- It makes the query to the database by using the key (pk). The object is the Meep object.  
  
- If the user has already liked the Meep, then the session is updated to remove that like; otherwise, a new like is added.  
  
- This only takes the user back to the other page.  
  
undefined  
  
- In addition, it has a label informing a user to log in.  
  
- It is redirected to the home page.  
  
**13. def meep\_show(request, pk):**  
  
- On the other hand, this function allows us to know a Meep.  
  
- It retrieves the Meep object with a given ID(pk) or produces a 404 error if does not exist.  
  
- If the Meep object is accurate, it inactivates the 'show\_meep.html' template and passes the Meep object to the template.  
  
- The error message displayed would show that the Meep does not exist, which will force the user to go back to the home page.

**14. def delete\_meep(request, pk):**  
  
- This functionality takes care of removing the Meep.  
  
undefined  
  
- It seeks a Meep object with a given ID (pk) or returns a 404 error if it doesn’t exist.  
  
- It accesses the owner whether the Meep belongs to the current user.  
  
- If the player has the Meep, the Meep is removed from the player's inventory.  
  
- This gives a successful notification of the erasure of the Meep.  
  
- It goes to the previous page.  
  
- If the user has no Meep, then it displays an error message and redirects the user to the home page.  
  
- If the user is not logged in, it reassures him to do so and it ejects him back to the page he came from.  
  
**15. def edit\_meep(request, pk):**  
  
- Such a service involves editing a Meep.  
  
undefined  
  
- It fetches the Meep object with a specified ID (pk) or returns a 404 error if not found.  
  
- It informs whether the user who is currently logged in is Meep's owner.  
  
- If a user owns a Meep, it instantiates a MeepForm instance using POST data or None; by using the information of the Meep currently at hand.  
  
undefined  
  
- it updates the Meep object with the new data.  
  
- It displays a success message, notifying that the Meep has been successfully updated.  
  
- This is the redirect page.  
  
- On the other hand, if the request method is different than POST, it renders the 'edit\_meep.html' template with the form and a Meep object as a template.  
  
- If the user is not signed in, a message displaying the login form and redirecting to the home page is added.

**16. def search(request):**  
  
- This is the part, which is conducted through search Meeps respectively based on the input given by the user.  
  
- If the request method is POST:- In case, the request method is POST.  
  
- It does simple filtering of the search command from the input field.  
  
- It asks the Meep database about the Meeps which carry the words that are encompassed by the search term.  
  
- It provides the 'search.html' template so that the search query and possible search results (if any) can be displayed. It accepts and carries the query and possible search results parameters in the form of template parameters.  
  
- The '/'responses redirect to the ''search.html'' display template without any context data if the request method is not POST.