ForgetNot Web Application

System requirements

- Python 3.6+
- Django 2.2+
- Node.js v14+

Back-end deployment

For local deployment, the line *axios.defaults.baseURL = "https://forgetnot.uk/"* in the *utils/Request.js* must be replaced with *axios.defaults.baseURL = "* http://localhost:3000/". Then follow below steps*:*

1. Open the terminal
2. Change directory to project directory *ForgetNotBack* from the location of the code files
3. Run command *pip install -r requirements.txt*
4. All the required packages will be installed now
5. Run command p*ython manage.py runserver* to start the development server.
6. The server will start running now

Front end deployment

1. Once the backend server started running, open another terminal
2. Change directory to project directory *ForgetNot* from the location of the code files
3. Run command *npm install*
4. All the required packages will be installed now
5. Run command *npm start* to load the webpage locally.
6. The landing page will be displayed on the browser

Frontend - Backend communication

The communication between the front end and the backend is implemented using RESTful webservices – *GET* and *POST*. HTTP client library called *Axios* is used for sending HTTP requests and receiving HTTP responses from the webserver asynchronously. Only textual data is used for transmission as the application do not require any multimedia transmission.

The landing page does not require any webservice calls. In the *SignUp* page, the user will be prompted to send a verification email by clicking a button. Upon clicking *Send verification code* button, *generate_verification_code* service will be called, and the actual email sending is handled by the backend. After entering the verification code and all the required details, clicking the *SIGN UP* button will post all user details to *register* webservice. Based on the response code from server, the user will be taken to

SignIn page, if successful. The user will be able to enter the registered email address and chosen password. Clicking *SIGN IN* button will call *login* webservice with required parameters. Based on the response code from the server, user will be able to enter the *Home* page, if successful.

While loading the home page, all the labels' lists will be fetched from the server by calling *get* webservice with required parameters. After it is received successfully, label list on the left side of the home page will be populated. At the same time, the events list of the first label will be fetched using the *get_list* webservice with required parameters. Then corresponding array list will be populated thereby updating events on the calendar. The same process will be followed every time a label on the left side list is clicked. The creation, updation and deletion on the events are synchronized with web server using *create*, *update* and *delete* webservices respectively.

Guest users will be able to view event details in two ways. First one, the registered users will be able to invite guests by typing their email address and clicking *INVITE* button. *get_vistor_list* webservice on the server will be called and the backend will send an email to the given email address with event details with an option to *Confirm* or *Decline* the invitation. Second way for guest to view event details is by typing the reference id (that is sent to the email by the host) on the landing page and submitting it. While loading the Guest page, the event details will be fetched from the server using *get* webservice of event and respective text fields will be populated based on that. Here also, guest users will be able to *confirm* or *reject* the invitation.

Registered user can view and update their profile details by clicking profile icon on the top right corner of the screen. While loading the profile page, user details will be retrieved from the webserver using *get* service related to users. Upon receiving successful response code, the user data will be populated into the respective text fields. The updation is handled by user *update* webservice. Users will be able to customize the labels by selecting a label and updating it using the text field. The label creation, updation and deletion is handled by label *create*, *update* and *delete* webservices.

The registered users or guest users will be able to contact us for any issues or concerns regarding the application through Contact us page. They will be able to enter their details, queries or messages in that page. Upon clicking on SEND button, all the entered details will be posted to the server using *contact* webservice. An acknowledgement will be sent to the given email address by the back end.