BACKEND API REQUESTS & RESPONSES (`${apiVersion}` is by default 'v1')

Category	**Requests**	**Responses**

HOME	GET /api/${apiVersion}/home;
	POST /api/${apiVersion}/home {query: string};
		GET -> {success: bool, message: string, events: [ObjID]}
		POST -> {success: bool, message: string, events: [ObjID], users: [ObjID], generalEvents: [ObjID]}

LOGIN	POST /api/${apiVersion}/login {username: string, password: string};	POST -> {message: string, token: string}
SIGNUP	POST /api/${apiVersion}/register {username: string, email: string, password: string};	POST -> {message: string, accessToken: string}

RECOVER	POST /api/${apiVersion}/recover {email: string};
	GET /api/${apiVersion}/recover/:token;
	POST /api/${apiVersion}/recover/:token {newPassword: string};
		POST /recover -> {message: string};
		GET /recover/:token -> {message: string};
		POST /recover/:token -> {message: string};

TERMS	POST /api/${apiVersion}/accept-terms {username: string};	POST -> {message: string};
EVENTS	/api/${apiVersion}/events/:eventCode	// NOT IMPLEMENTED YET
ADDEVENT	/api/${apiVersion}/addevent	// NOT IMPLEMENTED YET
USERS	/api/${apiVersion}/users/:username	// NOT IMPLEMENTED YET
SETTINGS	/api/${apiVersion}/settings	// NOT IMPLEMENTED YET

NOTES:

- Response structure is to be considered only in success cases, while in other cases a proper error code and response will be returned
- Frontend does not need to worry about input validation, token management and terms acceptance for allowing User's requests, as this is managed by the Backend through a middleware internal system
- `events` in home GET and POST responses is the list of events whose title match with the provided query and are organised by users who are followed by the current User (number of returned items: 10)
- `generalEvents` in home POST response is the list of events whose title match with the provided query (even if their organized is not in the 'following' list of the current User) (number of returned items: 10)