Parabol: checklist for a new mutation change

Server side:

DB types:

- [ ] Add the new field to database type definition in `packages/server/database/types/[TableName].ts`
- [ ] Remember to also update subclass. For example if you modify `Meeting`, you might also need to update `MeetingAction` and `MeetingRetrospective`
- [ ] Write a new DB migration script in `packages/server/database/migration/timestamp-newfieldDescription.ts`

GraphQL types:

- [ ] Update the corresponding GraphQL type based on the DB type, e.g., `NewMeeting` (a GraphQL type) vs. `Meeting` (a DB type). GraphQL types are in `packages/server/graphql/types/[GraphQLTypeName].ts`
- [ ] Create a new mutation payload in `packages/server/graphql/types/[GraphQLMutationTypeName].ts`
- [ ] Add your newly written `GraphQLMutationTypeName` into `packages/server/graphql/types/[Team|Meeting|Notification|Task|Organization]SubscriptionPayload.ts` according to which subscription the mutation should publish to

GraphQL mutations:

- [ ] Create a new mutation type and implementatio of its resolver in `packages/server/graphql/mutations/[MutationName].ts`
- [ ] Add your newly written mutation `MutationName` to `packages/server/graphql/rootMutation.ts`

Client side:

- [ ] Create the client side mutation fragment and operation in `packages/client/mutations/[MutationName]Mutation.ts` and implement its `updater` and `optimisticUpdater` callback functions for Relay store updates.
- [ ] Add your newly written mutation fragment `[MutationName]Mutation_[fieldName]` into `packages/client/subscriptions/[Team|Meeting|Notification|Task|Organization]Subscription.ts`
- [ ] Go to the corresponding subscription

`packages/client/subscriptions/[Team|Meeting|Notification|Task|Organization]Subscription.ts` and implement its custom `updateHandler` and `onNextHandler` for Relay store updates, if needed.