CCExtractor: Beacon (Backend)

Google Summer of Code - 2022

About Me

Name	Aadeesh Oswal
University	Indian Institute of Technology, Varanasi
Degree	Bachelor of Technology
Year	Sophomore
Timezone	Indian Standard Time (GMT +5:30)
Email	asoswal@gmail.com
Github	Aadeesh11
Resume	Aadeesh Oswal - resume
LinkedIn	https://www.linkedin.com/in/aadeesh-oswal-b422941b9/
Postal Address	Gulmohar Villa, 36, Shradhha Colony, Jalgaon, Maharashtra, India, 425002
Phone Number	(+91) 9422277144

Project

Synopsis

This project aims to add multiple features and optimisations to Beacon App’s backend, which is built with Apollo-GraphQL, Node.js and Express.js, and MongoDB. The app provides its user to share their location in real time and hence eases traveling in groups. Building upon the current app, users should be able to create groups, have multiple beacons within a group, preschedule beacons, deactivate beacons, share routes, and have a seamless app experience. Moreover the backend should be optimized enough to handle location sharing of multiple beacons in real time. Also writing a script that automates the deployment of backend to cloud, if a user wants to have their own server and database.

Motivation

The idea behind the app itself motivates me because of its myriad of real world applications. As an avid group hiker myself, I understand the pain to keep up with the group, and having an app that solves the problem motivates me. This also ensures that I commit my maximum time towards this project.

Why CCExtractor ?

CCExtractor has a lot of cool technologies that have sparked my interest in contributing to this community. Being a part of the community, I will have a far better opportunity of learning more about not just my project, but also other areas such as AI/ML, P2P etc. I've also witnessed interesting, high-quality debates on community channels, which has inspired me to learn more.

Also, one of the crucial aspects that helped me choose CCExtractor is the helpful atmosphere that the mentors create.

I am only applying for CCExtractor for GSoC’22 and have no plans to contribute to any other organization under GSoC’22.

Proposed Deliverables

- **N beacons** for **one group** feature (Major Update).

- A proper **installation script** for the backend.
- A **single subscription** for all beacon changes.
- **Relaying** beacon, i.e changing the leader of the beacon.
- **Pre Scheduling** and **deactivation/reactivation** of beacon.
- Adding **tests** for the backend.

Background Info

Since I have already started contributing to Beacon frontend and backend projects, and have my own backend instance up and running, I have a decent amount of understanding of both the codebases and thus can make the necessary changes for all the required features. I have reported issues and implemented some mutations on the backend. Details of all my contributions to CCExtractor can be found in the "**Contributions**" section. I have also started working on N beacons, 1 group feature and will be proposing the required changes in the next sections.

Plan of Action

I will start making required changes for N beacons, 1 group feature, such as model changes, adding mutations and queries so that the frontend dev/team don't have to wait to start developing the feature on their side.
After implementing the necessary changes, I will start writing tests on the backend.
After all the tests pass, the backend can be deployed to production, and the work on installation script will be started.

Timeline

Following is the breakout of the time I'll spend different aspects the Project:

- 50% - Adding major features.

- 25% - Installation Script and deployment.
- 15% - Writing Tests.
- 10% - Maintaining documentation.

Community Bonding Period (C.B.P) *May 20 - June 12*

- I'll discuss the proposed data-model and schemas with the front-end developers and the mentors, for the N beacon, 1 group and also about the implementation of subscription streams.
- I'll discuss the expected documentation and testing standards.
- Setup GSoC blog.

Coding Period - 1 *June 13 - July 24*

- I'll be making necessary model changes for the N beacon 1 group.
- I'll be updating / creating new mutations, queries and subscriptions to incorporate the new "group" model and features.
- I'll work on merging all the subscriptions into one.
- I'll fix any bugs that come up during the implementation of the said features.

Evaluation *July 25 - July 29*

- All necessary changes for N beacon, 1 group will be complete.
- All the queries, mutations and subscriptions will work.

Coding Period - 2 *July 30 - Sep 12*

- Writing tests for backend.
- Start working on an installation script.
- Work on deployment on AWS lambda using Serverless framework.
- I'll clean up the documentation and code to ensure that it is organized perfectly.
- Focus on reviews from mentors and fixing bugs that come up.

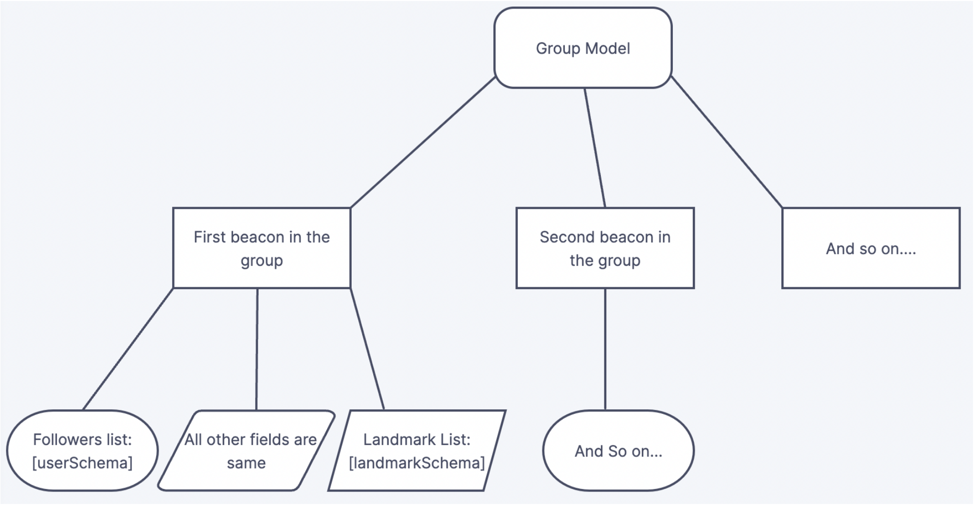
Final Evaluation *Sep 13 - Sep 19*

- All the proposed deliverables will be completed and would be ready for testing.
- All tests can be run and will pass.

Detailed Plan And Implementation

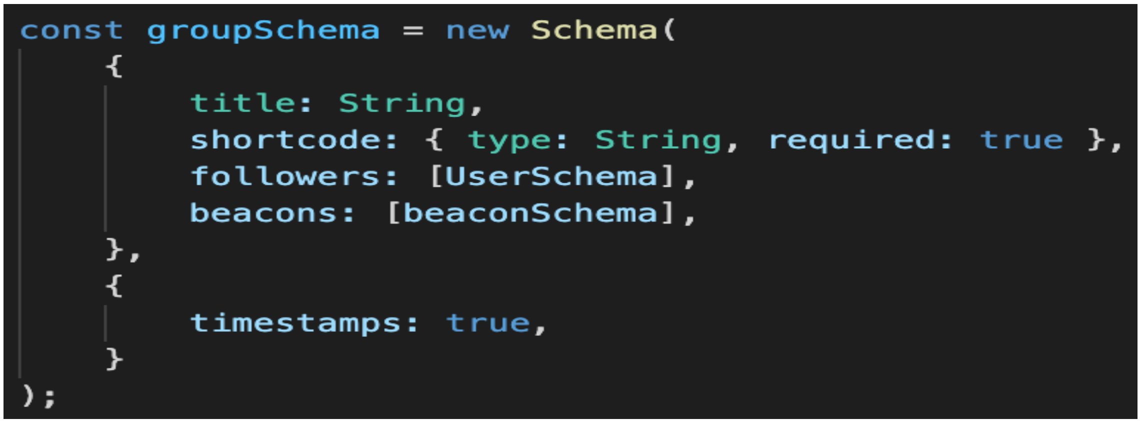
Proposed Data Model

In the data model, we'll be having beacon's field as a subcollection of the group model. (The thing that needs to be discussed is whether the follower's list must be the same for the entire group or a different list for each beacon). One thing that can


Group Model
First beacon in the group
Second beacon in the group
And so on....
Followers list: [userSchema]
All other fields are same
Landmark List: [landmarkSchema]
And So on...

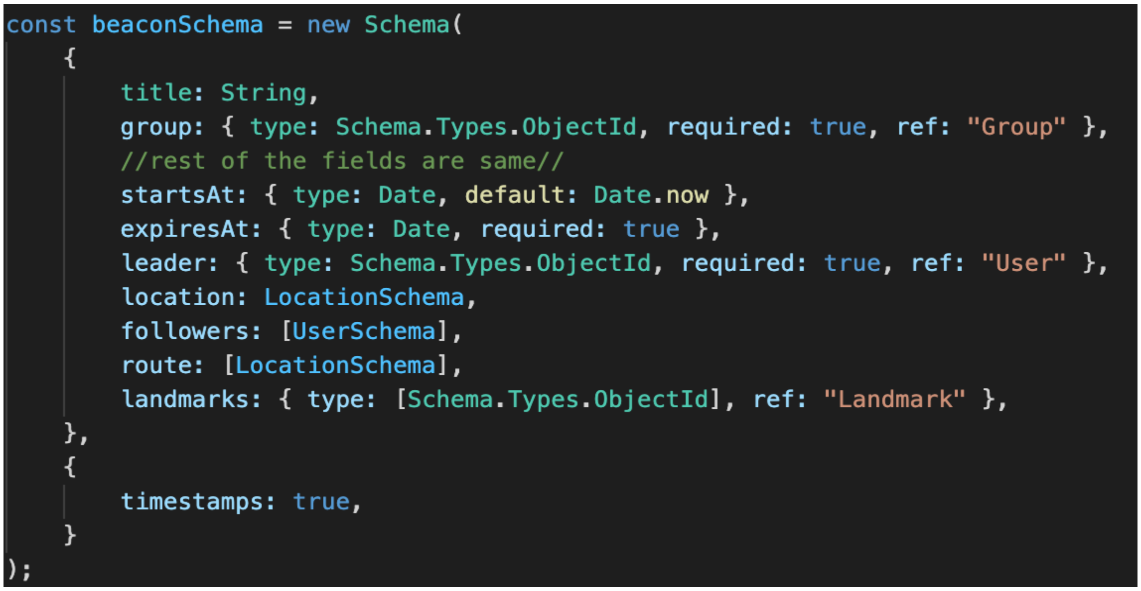
be done is that, we can have 2 lists, one for group members and other individual lists for each beacon.

Group Schema:



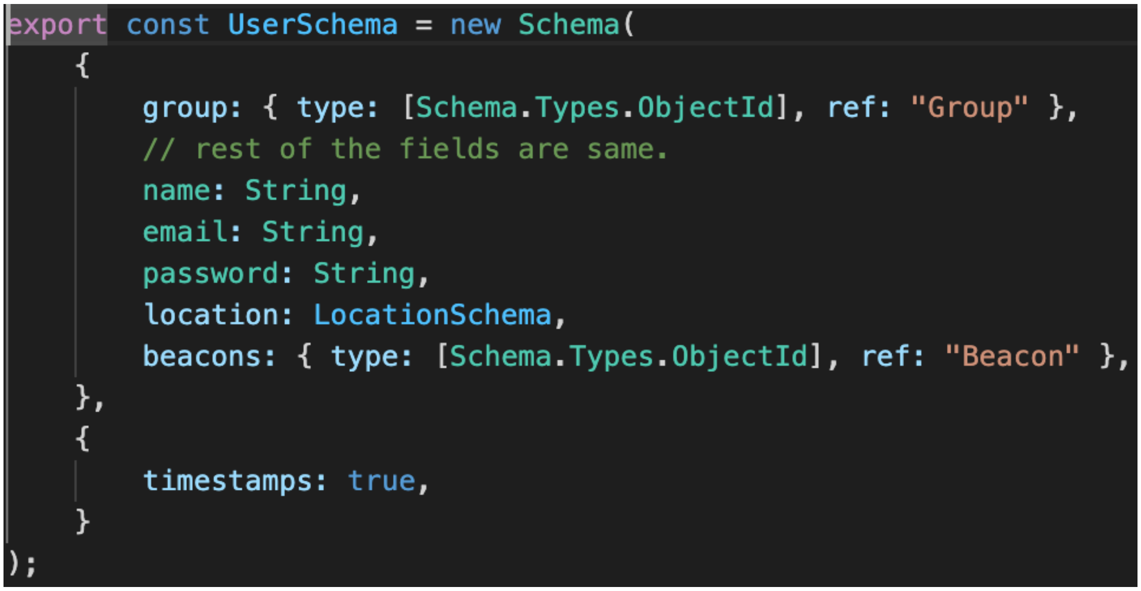
```
const groupSchema = new Schema(
    {
        title: String,
        shortcode: { type: String, required: true },
        followers: [UserSchema],
        beacons: [beaconSchema],
    },
    {
        timestamps: true,
    }
);
```

Beacon Schema:



```
const beaconSchema = new Schema(
    {
        title: String,
        group: { type: Schema.Types.ObjectId, required: true, ref: "Group" },
        //rest of the fields are same//
        startsAt: { type: Date, default: Date.now },
        expiresAt: { type: Date, required: true },
        leader: { type: Schema.Types.ObjectId, required: true, ref: "User" },
        location: LocationSchema,
        followers: [UserSchema],
        route: [LocationSchema],
        landmarks: { type: [Schema.Types.ObjectId], ref: "Landmark" },
    },
    {
        timestamps: true,
    }
);
```

User Schema:



```
export const UserSchema = new Schema(
    {
        group: { type: [Schema.Types.ObjectId], ref: "Group" },
        // rest of the fields are same.
        name: String,
        email: String,
        password: String,
        location: LocationSchema,
        beacons: { type: [Schema.Types.ObjectId], ref: "Beacon" },
    },
    {
        timestamps: true,
    }
);
```

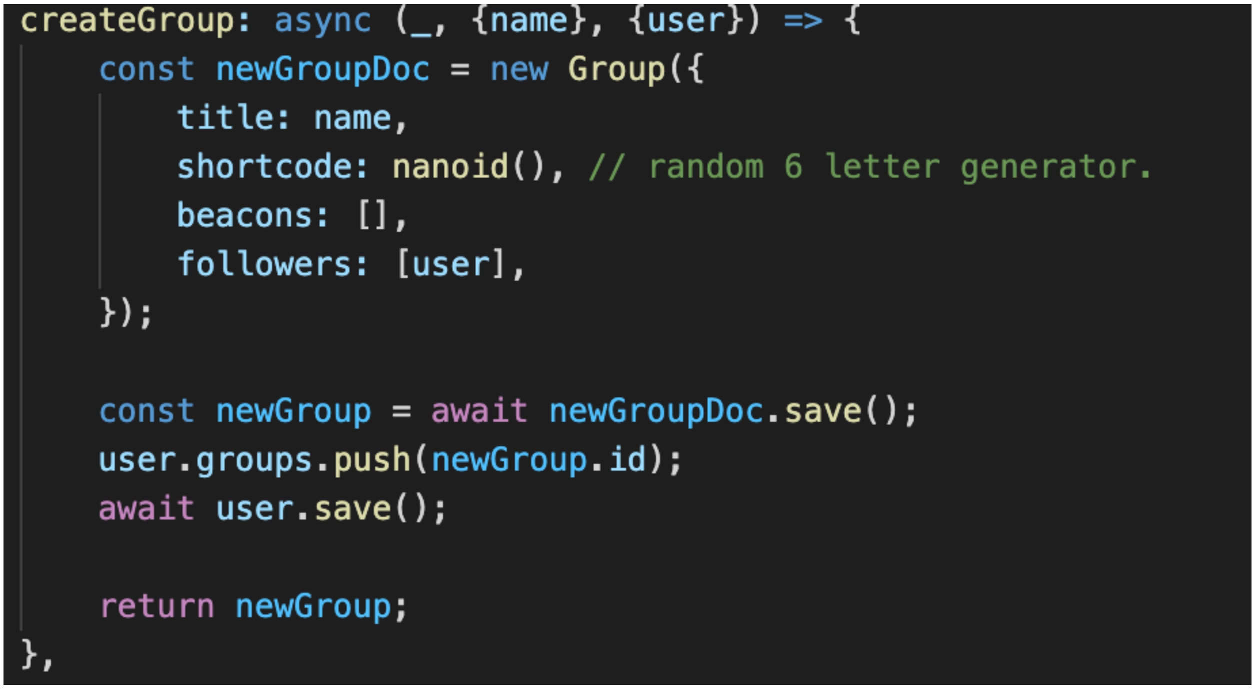
The Landmark model does not need any changes since it is a subcollection of the beacon model already.

Creating and Joining a group.

To achieve this following mutations can be defined:

While creating a group only its name is required, since the beacon list will be empty.

Pseudo Code:

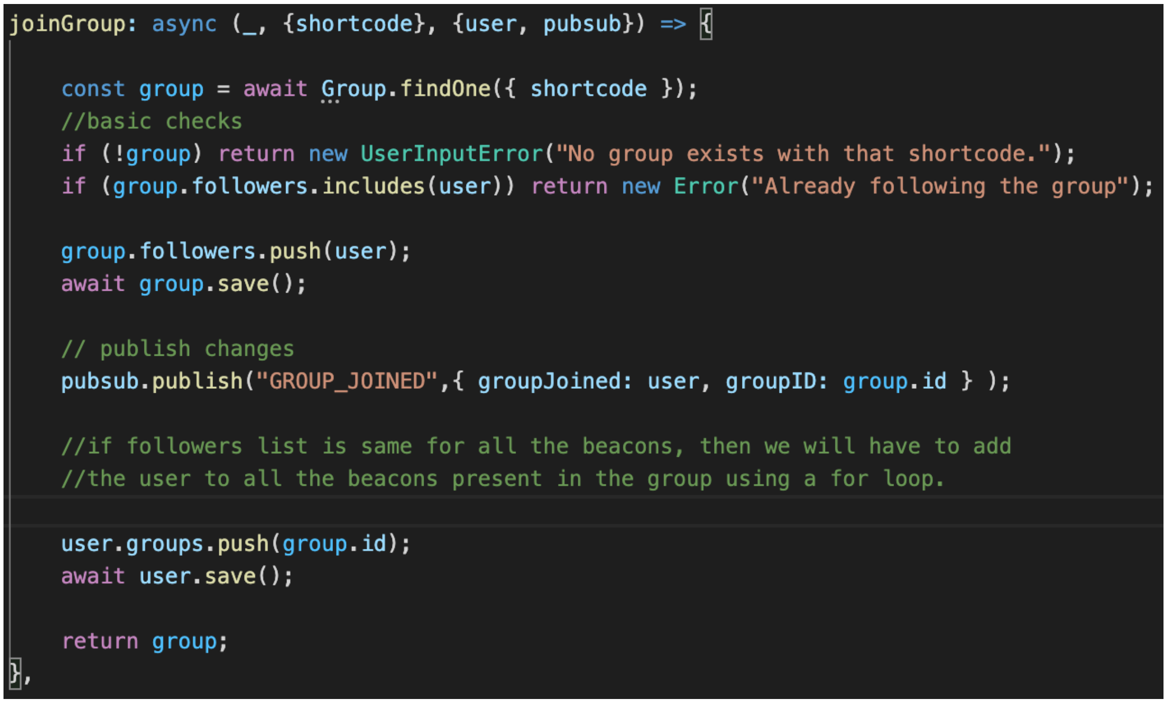


```
createGroup: async (_, {name}, {user}) => {
    const newGroupDoc = new Group({
        title: name,
        shortcode: nanoid(), // random 6 letter generator.
        beacons: [],
        followers: [user],
    });

    const newGroup = await newGroupDoc.save();
    user.groups.push(newGroup.id);
    await user.save();

    return newGroup;
},
```

While joining some basic validation needs to be performed and then users can be added to the group.

Also if the followers list is the same for all the beacons, then when a user joins a group, we will need to add the user to all the beacons present in the group. This can be achieved using a simple for-loop.

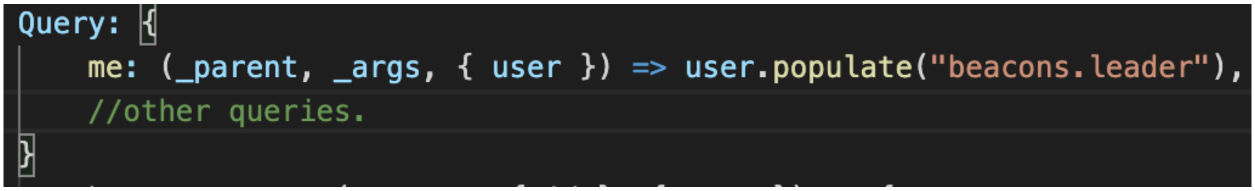
Pseudo Code:



```
joinGroup: async (_, {shortcode}, {user, pubsub}) => {

    const group = await Group.findOne({ shortcode });
    //basic checks
    if (!group) return new UserInputError("No group exists with that shortcode.");
    if (group.followers.includes(user)) return new Error("Already following the group");

    group.followers.push(user);
    await group.save();

    // publish changes
    pubsub.publish("GROUP_JOINED",{ groupJoined: user, groupID: group.id } );

    //if followers list is same for all the beacons, then we will have to add
    //the user to all the beacons present in the group using a for loop.

    user.groups.push(group.id);
    await user.save();

    return group;
},
```

Fetching all groups of a user.

To get all the groups of a user, the current “me” query can be modified to return all the groups the user has joined.



```
Query: {
    me: (_parent, _args, { user }) => user.populate("beacons.leader"),
    //other queries.
}
```

Writing tests for Backend.

To test the api, queries, mutations I will use these libraries:

- **apollo-server-testing**: to launch the apollo server api under test. It gives the ability to run graphql queries and mutations against it. It's the **official recommendation** from apollo.

- **@shelf/jest-mongodb** to create a **mongodb** in memory.

- **nock** and **graphql-query-test-mock** to mock external api calls and tell if you receive that http request on that url, please return this.
- **jest-each** to create test cases using a template. **jest-each** is an official package from facebook

I will also discuss the required testing standards with my mentors and will follow this article and the official docs to achieve the best possible implementation.

Deployment on AWS-Lambda.

I will use the Serverless framework to deploy on AWS-Lambda and will follow the Official docs for deployment. Serverless is a framework that makes deploying to services like AWS Lambda simpler. For the production deployment on AWS, org's credentials and keys can be used. The ec2 instance also needs to be updated.

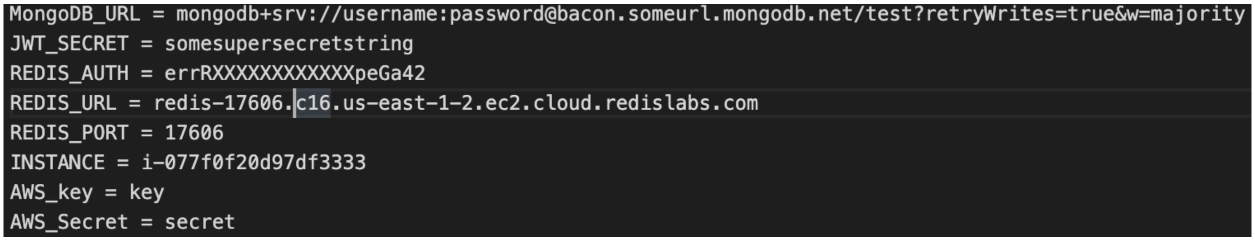
Installation Script.

The purpose of the script is that it takes connection Strings/URLs from a prepopulated file or maybe it's an interactive script that asks for it, and deploys the backend to cloud, so that if normal users want to use their own cloud server and database they will be able to deploy the backend easily.
The user needs to have accounts in:

1) MongoDB
2) Redis
3) AWS

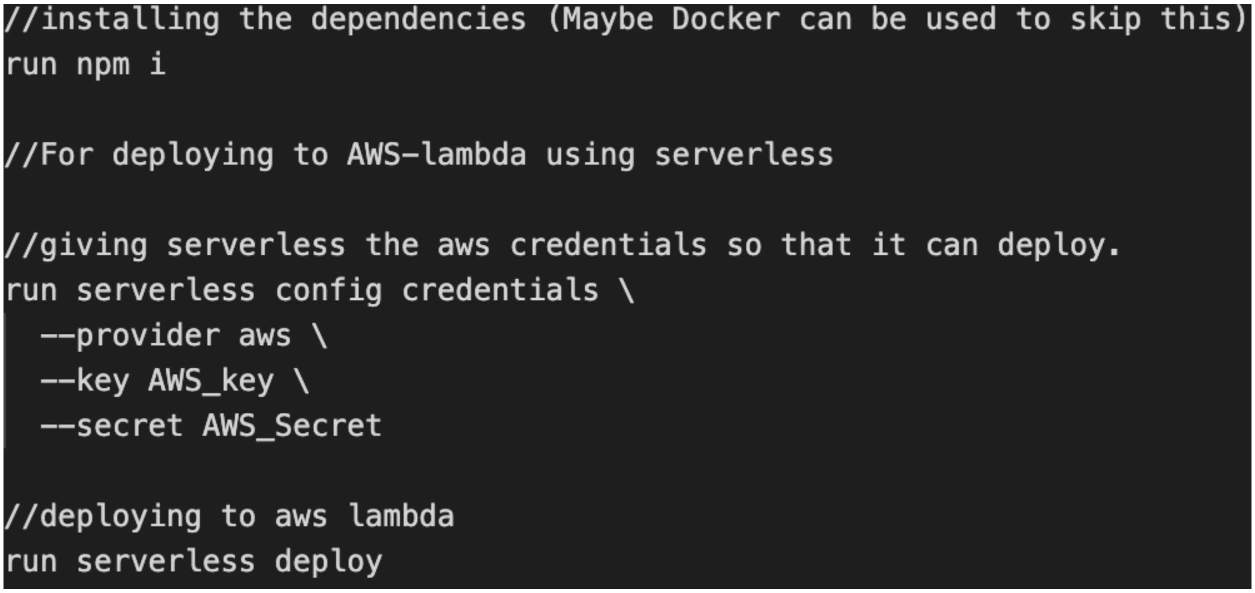
The script will deploy the aws-branch of the backend using "serverless deploy". Then the user will have to provision an ec2 instance from the AWS-console (A stretch goal is that the script can automate this too) and configure it using the readme.md (which will be updated to be a comprehensive guide using Carlo's notes on installation).

Configuration file may look like this (which will need to be pre populated):



```
MongoDB_URL = mongodb+srv://username:password@bacon.someurl.mongodb.net/test?retryWrites=true&w=majority
JWT_SECRET = somesupersecretstring
REDIS_AUTH = errRXXXXXXXXXXXXpeGa42
REDIS_URL = redis-17606.c16.us-east-1-2.ec2.cloud.redislabs.com
REDIS_PORT = 17606
INSTANCE = i-077f0f20d97df3333
AWS_key = key
AWS_Secret = secret
```

The pseudo code (what script aims to automate):



```
//installing the dependencies (Maybe Docker can be used to skip this)
run npm i

//For deploying to AWS-lambda using serverless

//giving serverless the aws credentials so that it can deploy.
run serverless config credentials \
  --provider aws \
  --key AWS_key \
  --secret AWS_Secret

//deploying to aws lambda
run serverless deploy
```

The part that can not be automated is creating accounts in the required places and getting the necessary connection URLs.

The part that remains to be automated is the EC2 stuff which user currently does using the AWS console or AWS-CLI.

Contributions to Beacon-Backend Project:

Contributions:

- #71: Feature Request: A Graphql mutation to allow changing the duration of beacon once it has been created.
- #73: User should not be able to join expired beacons.
- #76: User should not be able to join the beacon he/she is already part of.
- #107: One subscription for all beacon changes.
- #91: fix: leader name is being returned as anon.

Contributions to Beacon-Frontend Flutter Project:

Contributions:

- #11 Migrating Deprecated Widgets and Removing unused imports.
- #15 Overflow issue on homescreen.
- #16 Misaligned Buttons on dialogue box.
- #17 [Hike Screen] Unexpected behavior when leader tries to change duration.
- #20 [Beacons]: Users can see beacons even after they have expired.
- #27 [Auth Screen] Bug: If keyboard is open and we switch tabs, keyboard stays open and input is connected to prev tab.
- #41 CI failing due to flutter analyse command.
- #48 [Beacons]: User is able to join his own newly created beacon. (reported issue)
- #62: [Feat + rfrac]Hike screen updated to follow Stacked MVVM architecture + share route if beacon is active + camera auto zooms to necessary map surface.
- #102 Feature request: Display markers for each and every user of the beacon. (issue reported).
- #94: feat: Maintain State when network is lost
- #117 Feature: Animate markers position and transitions.
- #97: feat: Allow leader to change beacon duration, and deactivate the beacon
- #126 Feature Request: Reactivate Beacon after it has expired on leader's demand. (issue reported)
- #127 Feature Request: Custom Hike reminders.
- #136 Sync the updates for all and stop the subscriptions if the beacon has been deactivated.
- #114: Test: Model test (Beacon and User model

Personal Information

More about me

I am currently studying in Indian Institute of Technology (BHU), Varanasi. I am always on the lookout for new technologies and love to contribute to open source software. I am a full stack developer at my college's elite coding club, Club of Programmers. I enjoy playing games (especially Esports), traveling, working in a team in hackathons, squashing bugs, and working on tough projects, making bots to automate tasks. I like competitive programming, problem solving and software development.

Mode of Communication

Any of the communication channels are fine with me. On weekdays, I am able to work full time and am usually active between **11.30 AM IST (6 AM UTC) to 2.30 AM IST (9 PM UTC).** On weekends, I would love to discuss with my mentors and other teammates on the issues and bugs that occur during the week.

I plan to stick to the schedule and put in equal hours every week. There are currently no planned absences but I will let my mentor(s) know as soon as anything comes up.

Post GSOC

If any work is left unimplemented, for eg, documentation/testing or feature, then I would continue working post-gsoc to complete all the tasks proposed and will keep contributing and maintaining the project.

Thank You.