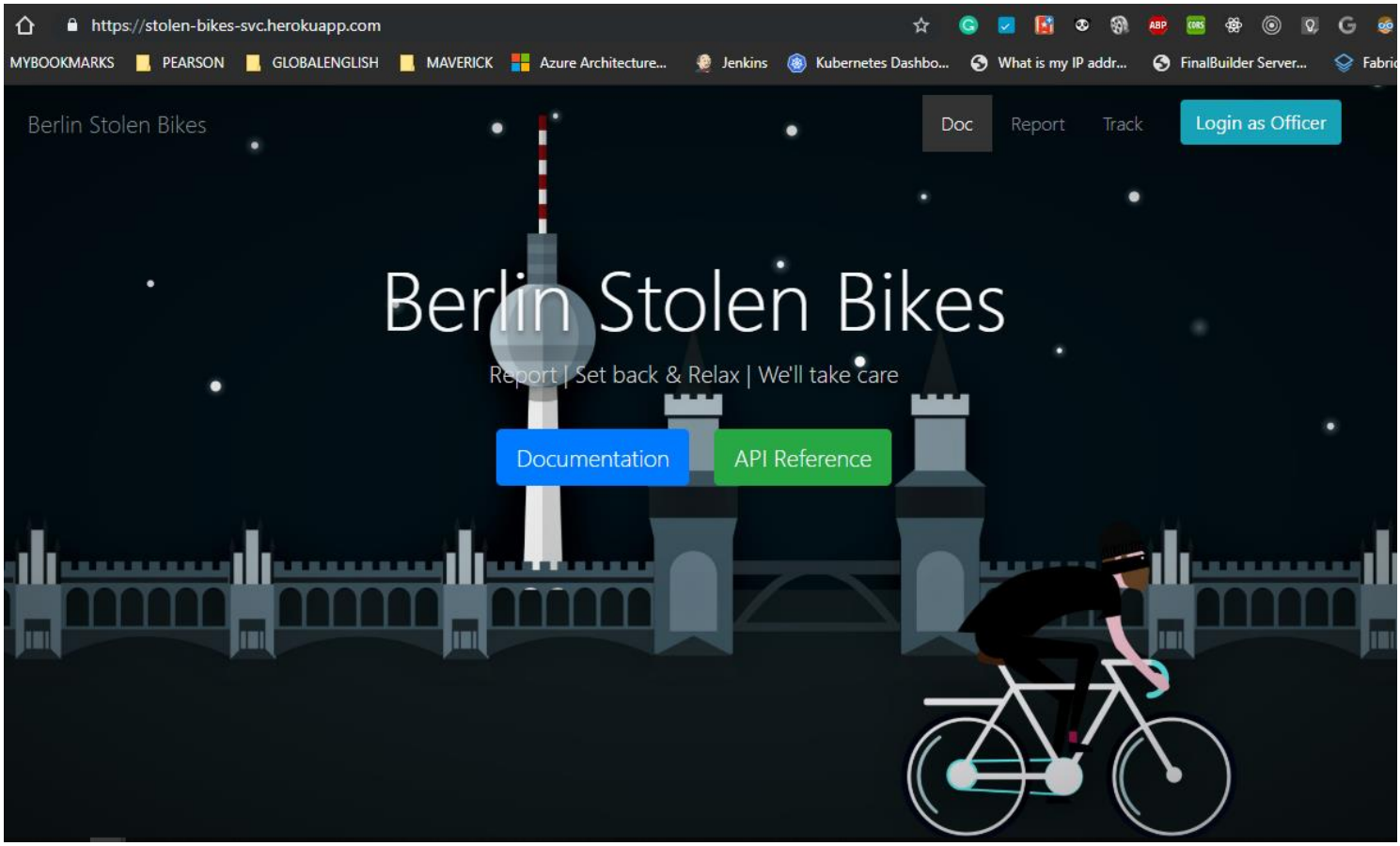


Berlin Stolen Bikes

JOIN.COM – ONLINE CODING CHALLENGE

Renish Bhaskaran | Senior Engineer | 12th May 2019

DOCUMENTATION

Requirement

Stolen bikes are a typical problem in Berlin. The Police want to be more efficient in resolving stolen bike cases.

1. Bike owners can report a stolen bike.
2. A bike can have multiple characteristics: license number, color, type, full name of the owner, date, and description of the theft.
3. The Police can increase or decrease the number of police officers.
4. Each police officer should be able to search bikes by different characteristics in a database and see which police officer is responsible for a stolen bike case.
5. New stolen bike cases should be automatically assigned to any free police officer.
6. A police officer can only handle one stolen bike case at a time.
7. When the Police find a bike, the case is marked as resolved and the responsible police officer becomes available to take a new stolen bike case.
8. The system should be able to assign unassigned stolen bike cases automatically when a police officer becomes available.

Clarification

1) Do we have two different roles here? Police and Police Officer?

I mean Police (Admin) who creates/maintains other police officers? If not peers can create/maintain peers? Could you please clarify this?

Anton: *no roles, all endpoints are publicly available. No need to have any authentication or authorization.*

2) As said the system automatically assigns/unassigns cases to police officers. I can find, two scenarios here.

1) When the bike owners create a case, we can automatically assign it to police officers

2) When the police officers are created, we can automatically assign him if any unassigned cases there in DB.

Here is my question/clarification, what happens if there is a shortage of police officers?

(or) Shall I assume we have enough officers?

(or) I can put the case status to unassigned & whenever the police officers are created, we can assign them with the case (in descending order, so the priorities are met)

Please do let me know.

Anton: *A case is not assigned if there is no available police officer. And automatically assigned when a new officer is added. So your scenarios both valid.*

3) As per the requirement, it seems bike owners don't have login/uniqueId to backtrack his filed cases. So my implementation (initial) would be serving him back with caseId, which he can use it to find the details later. Let me know if this is fine or I can improve it by getting additional details from bike owners?

Anton*: no need in any login, it's enough to return case id*

4) Since you've mentioned SQL database only, just wanted to clarify MongoDB cannot be used for this project right?

Anton*: we use Postgres so knowledge of SQL databases is important*

Note: I'm going to use Javascript (ECMAScript 6+) since I'm more familiar with this than Typescript. No problem for me to convert the whole code to Typescript later.

Anton*: No need to do in Typescript, please do in es6*

Planning

Actors:

1. Owner (One who reports the case)
2. Police (One who works and resolve a case, One who can add other police)

Police

Owner

Technology:

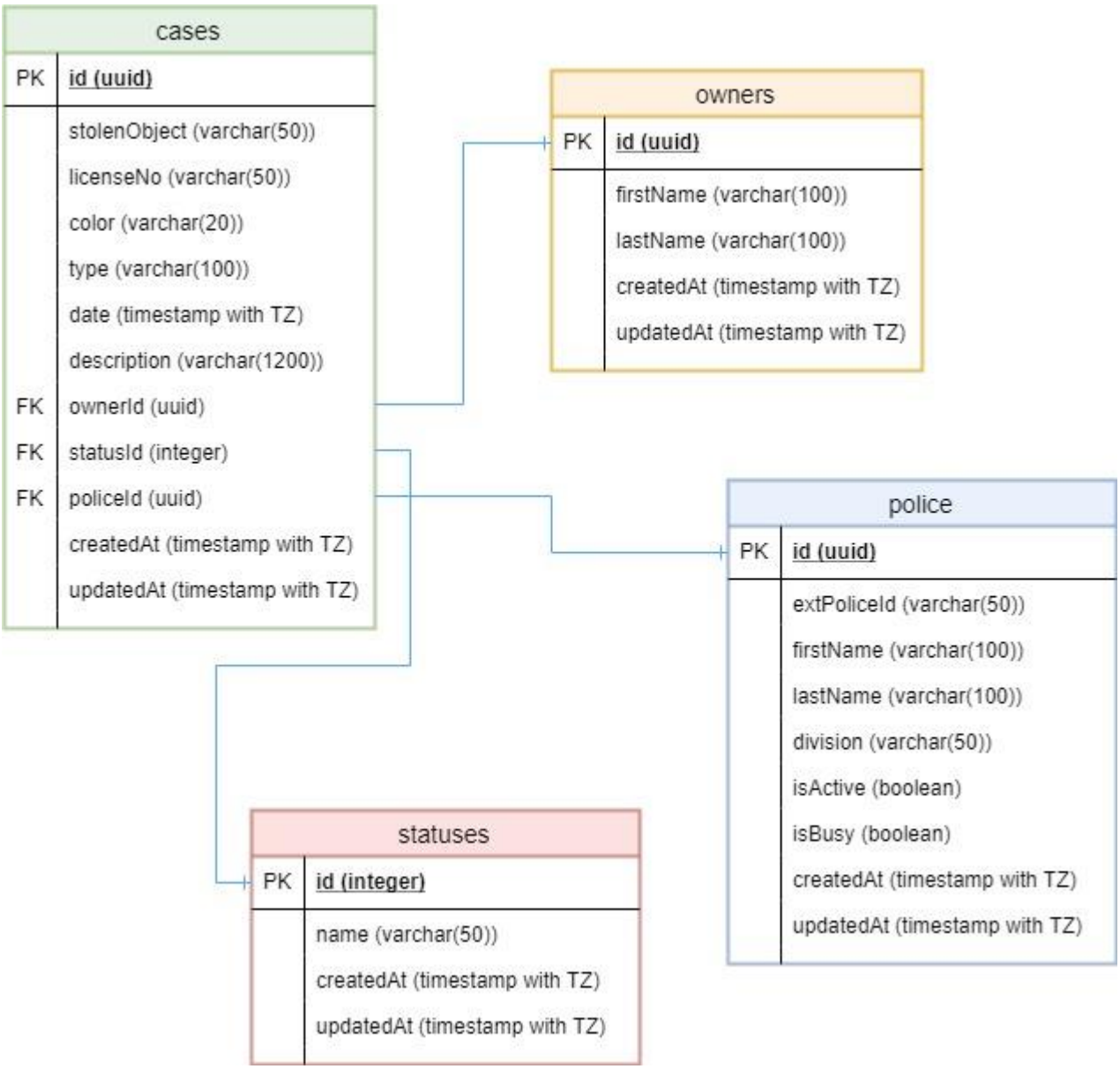
- Node.js
- EcmaScript 6+
- Express.js
- Postgres DB
- Sequelize ORM
- Sentry Log
- Swagger
- Heroku

Design

Core APIs (Inception idea):

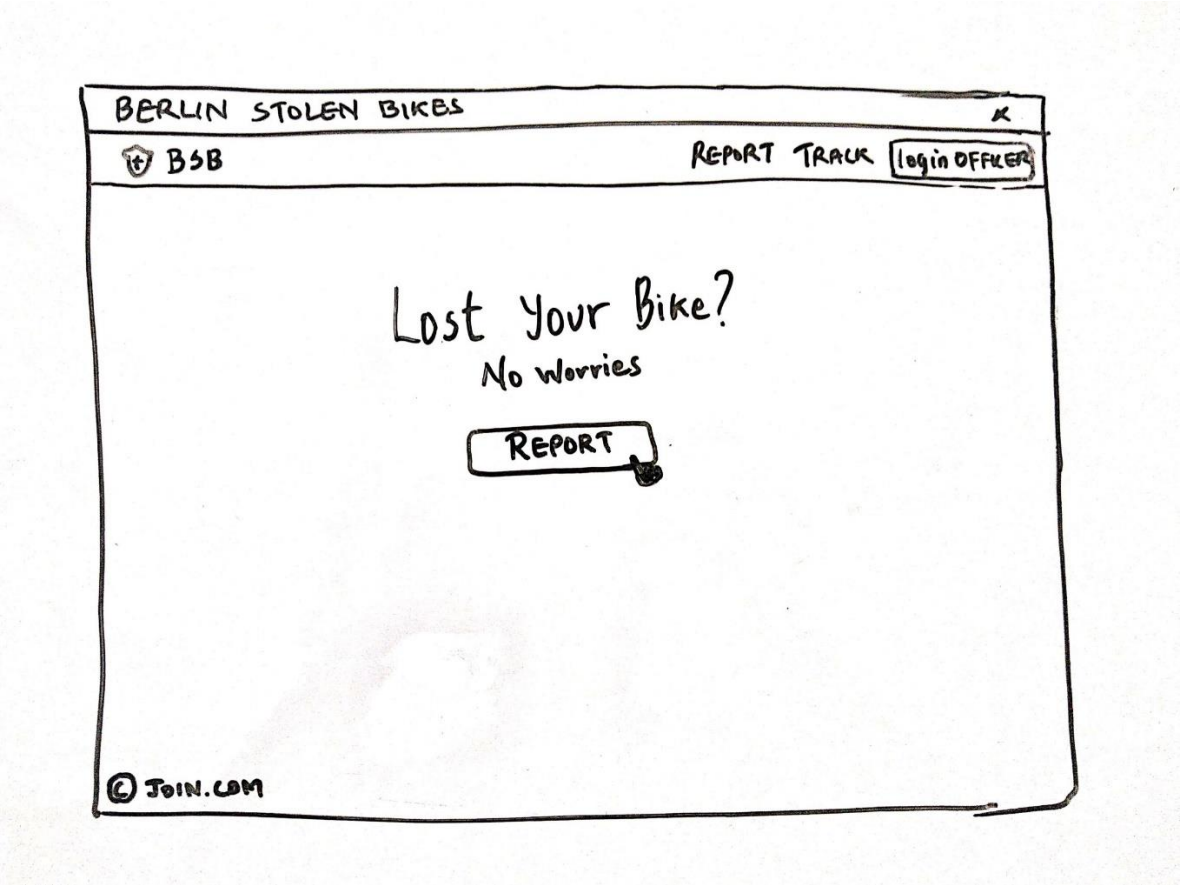
- POST Case API
- GET Case API
- PATCH Case Resolve API
- POST Police API
- PUT Police API
- DELETE Police API

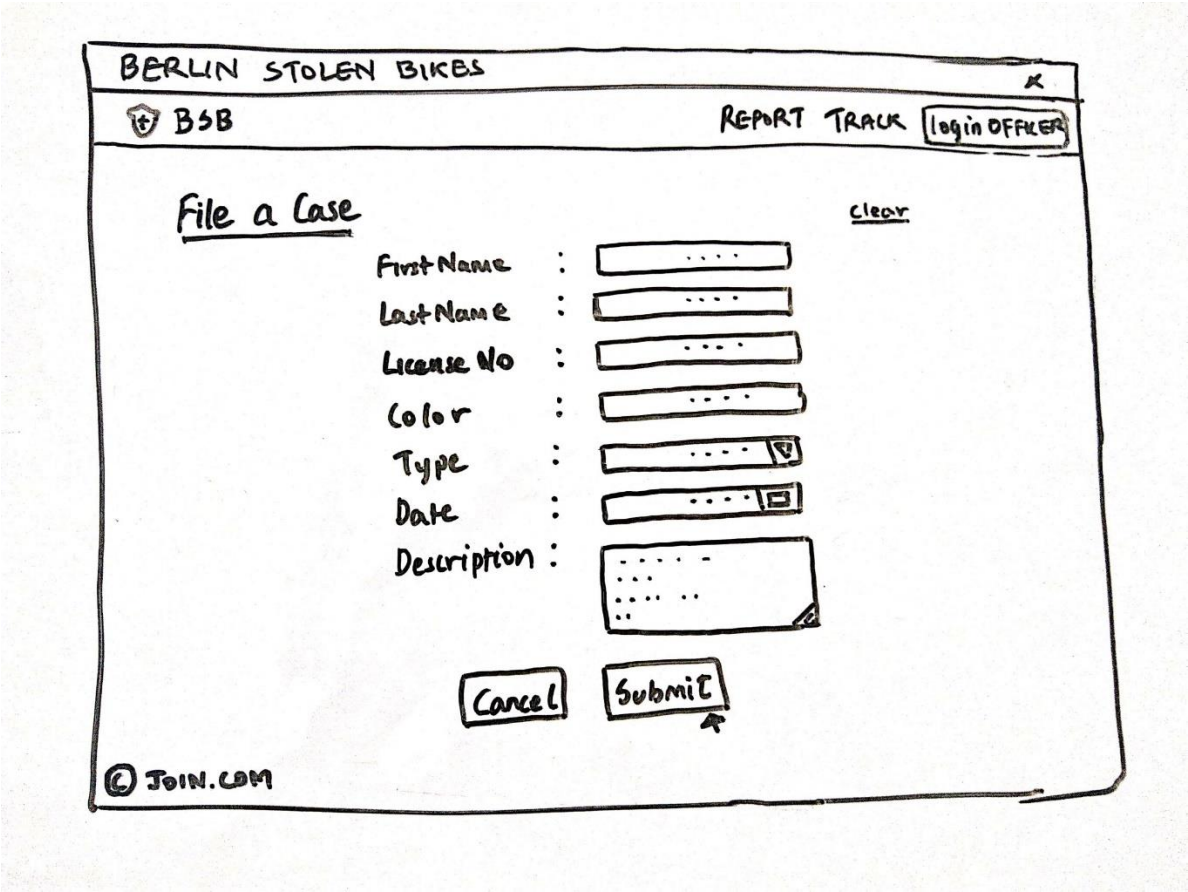
Entity Relationship Diagram:

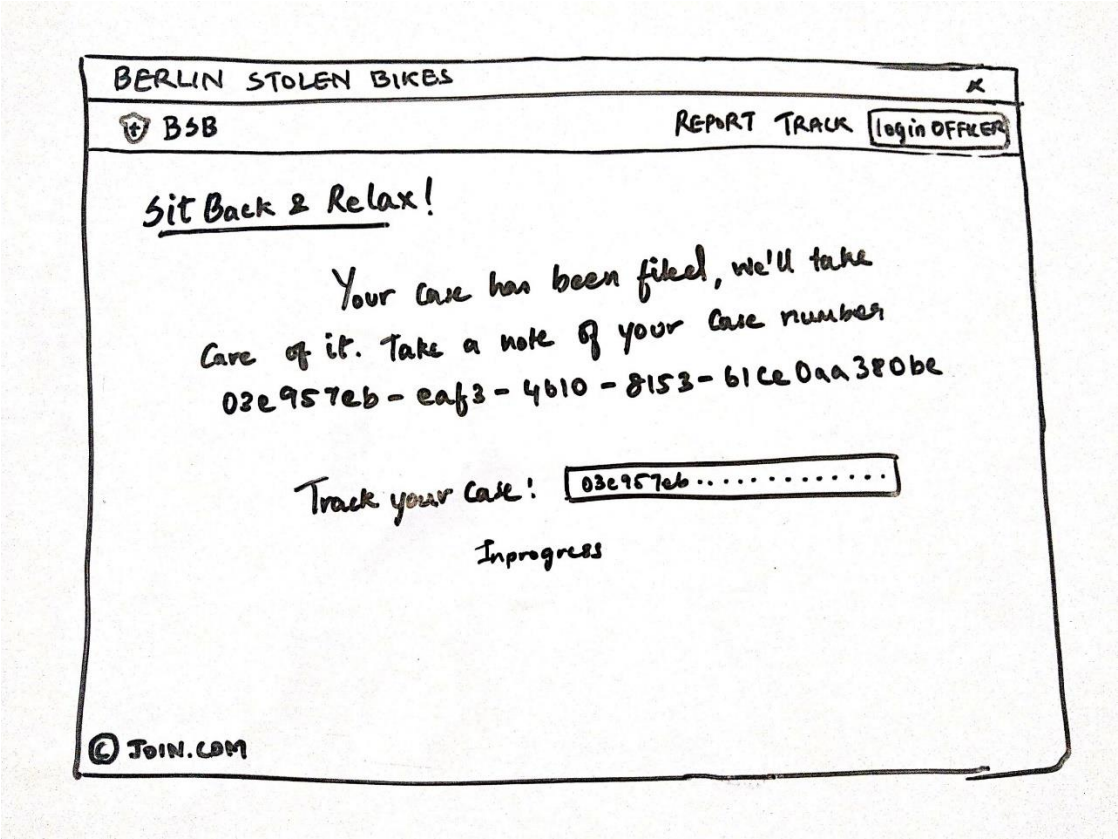


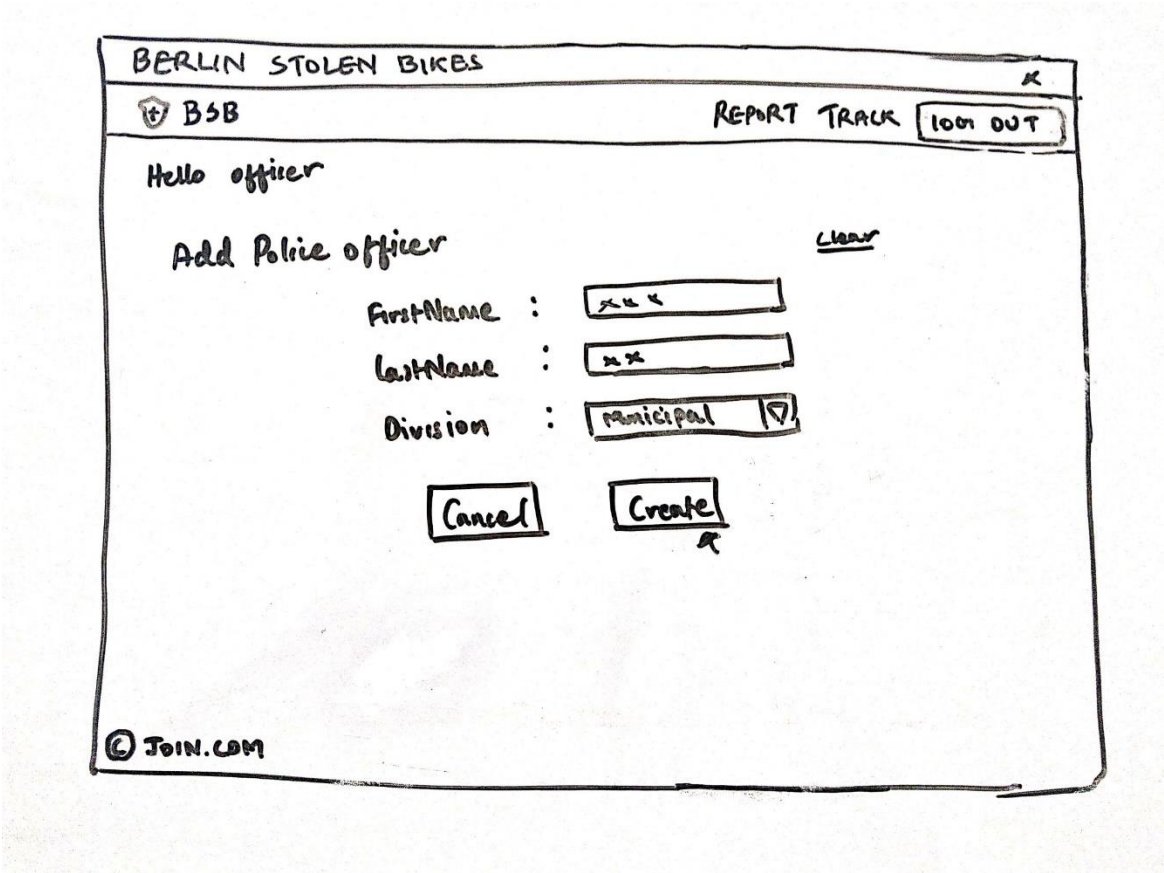
UX Wireframe

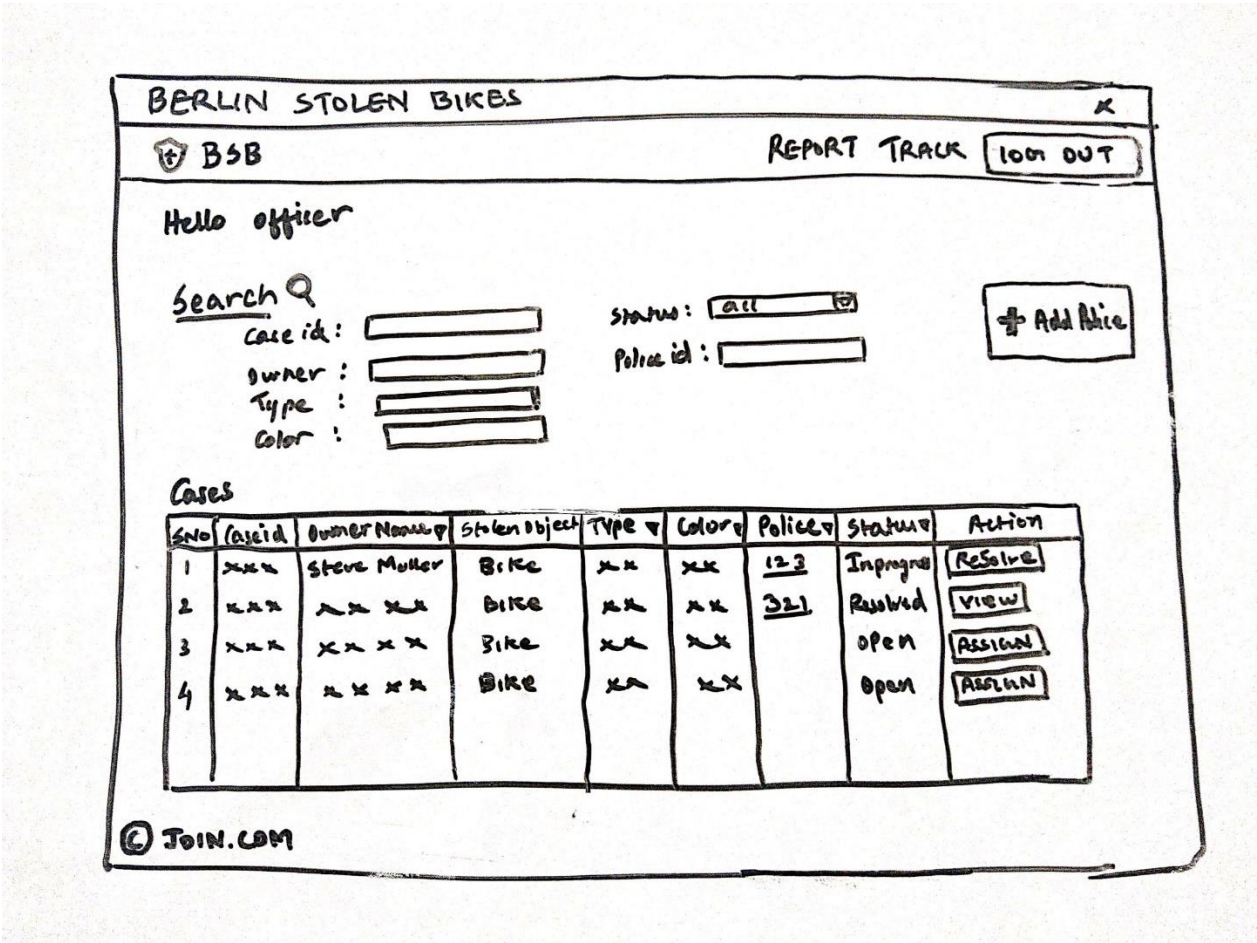
- This is just for my reference, I've drawn this on a whiteboard & captured using CamScanner App.




BERLIN STOLEN BIKES
BSB
REPORT TRACK
login OFFICER
File a Case
clear
First Name :
Last Name :
License No :
Color :
Type :
Date :
Description :
Cancel
Submit
© JOIN.COM

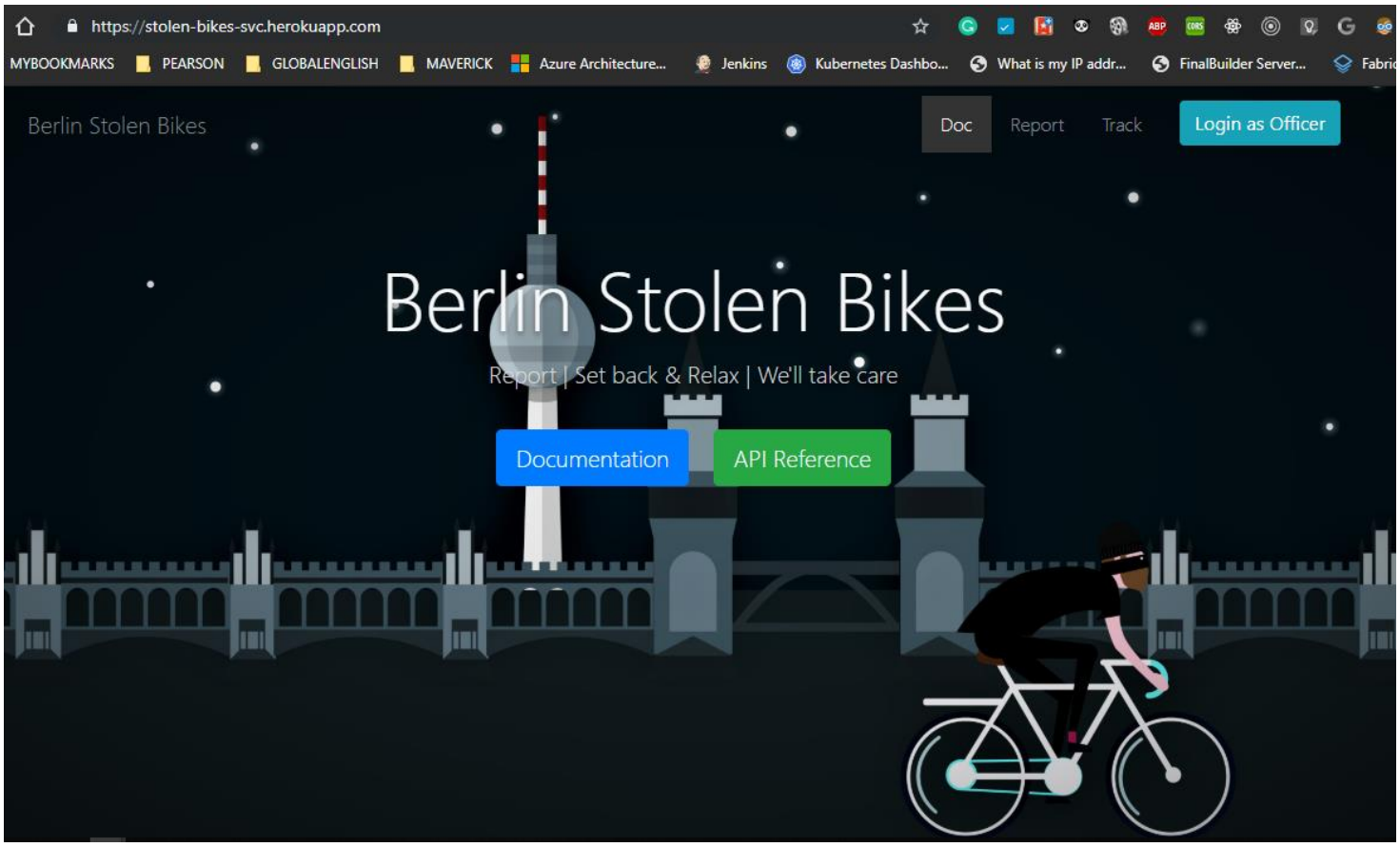

BERLIN STOLEN BIKES
BSB
REPORT TRACK
login OFFICER
Sit Back & Relax!
Your case has been filed, we'll take care of it. Take a note of your case number
03e957eb-eaf3-4610-8153-61ce0aa380be
Track your case!
03e957eb
Inprogress
© JOIN.COM


BERLIN STOLEN BIKES
BSB
REPORT TRACK
LOG OUT
Hello officer
Add Police officer
Clear
FirstName :
LastName :
Division :
Municipal
Cancel
Create
© JOIN.COM


BERLIN STOLEN BIKES
BSB
REPORT TRACK
LOG OUT
Hello officer
Search
Case id :
Owner :
Type :
Color :
Status :
all
Police id :
Add Police
Cases
SNo | Caseid | Owner Name | Stolen object | Type | Color | Police | Status | Action
1 | xxx | Steve Muller | Bike | xx | xx | 123 | Inprogress | Resolve
2 | xxx | xx xx | Bike | xx | xx | 321 | Resolved | View
3 | xxx | xx xx | Bike | xx | xx | | open | Assign
4 | xxx | xx xx | Bike | xx | xx | | Open | Assign
© JOIN.COM

Launch Page:

- Used bootstrap boilerplate code and express static site serving mechanism.

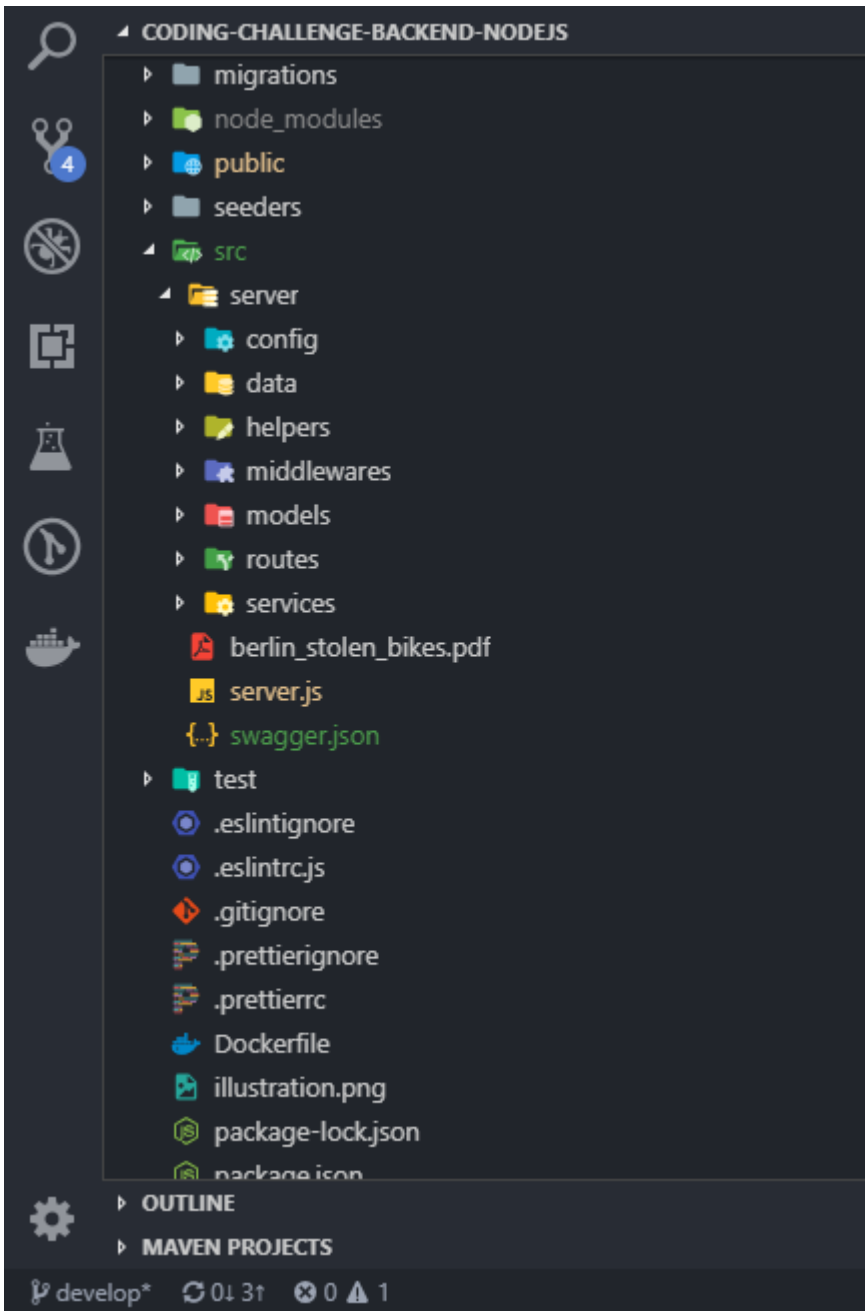


Code

Boilerplate:

- No pre-written boilerplate used.
- Started from scratch NPM init.
- Added required libraries to the package.json initially based on prior experience.

Folder Structure:



Code branching strategy

- Used Git flow for the branching.
- Just for demonstration purpose I've created develop branch from master and feature/xyz from develop branch.
- Raised PR for merging.

Visual Studio Code Setup

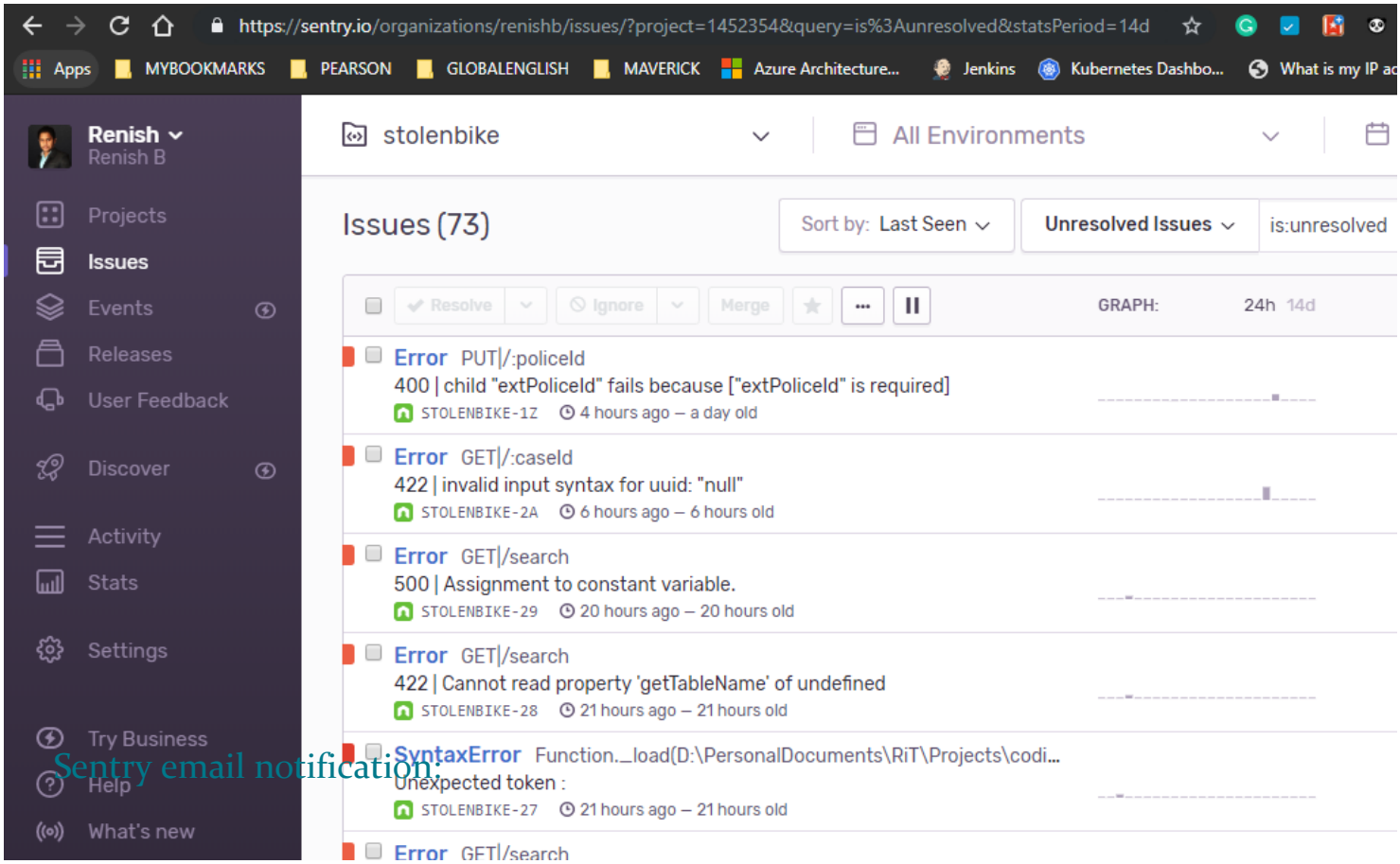
- Installed ESLint extension
- Installed Prettier extension

Express utility middlewares:

- Helmet: It helps to secure our Express apps by setting various HTTP headers. It's not a silver bullet, but it can help!
- Lodash: It makes JavaScript easier by taking the hassle out of working with arrays, numbers, objects, strings, etc.
- UUID for V4 uuid generation
- CORS

Logging:

- Subscribed a free account to Sentry.io and used for our project.



Sentry email notification:

STOLENBIKE-2A - Error: 422 | invalid input syntax for uuid: "null" Inbox ×

Sentry <noreply@md.getsentry.com> Unsubscribe 3:56 PM (5 hours ago)
to me

SENTRY

View on Sentry

New alert from stolenbike

ISSUE

Error **Error** *GET|/:caseId*
422 | invalid input syntax for uuid: "null"

May 13, 2019, 10:26:16 a.m. UTC ID: 102b1617dc1f4e7a8d5d7ac4679ece8e

Exception

```
Error: 422 | invalid input syntax for uuid: "null"
  File "D:\PersonalDocuments\RiT\Projects\coding-challenge-backend-
nodejs\node_modules\winston-sentry-log\dist\index.js", line 99, in
sentryClient.configureScope
    this.sentryClient.captureException(is_1.isError(info) ? info :
new Error(message));
  File "D:\PersonalDocuments\RiT\Projects\coding-challenge-backend-
nodejs\node_modules\winston-sentry-log\node_
modules\@sentry\hub\dist\hub.js", line 235, in Hub.configureScope
    callback(top.scope);
  File "D:\PersonalDocuments\RiT\Projects\coding-challenge-backend-
nodejs\node_modules\winston-sentry-log\node_
modules\@sentry\minimal\dist\index.js", line 18, in callOnHub
    return hub[method].apply(hub, tslib_1.__spread(args));
  File "D:\PersonalDocuments\RiT\Projects\coding-challenge-backend-
nodejs\node_modules\winston-sentry-log\node_
modules\@sentry\minimal\dist\index.js", line 90, in
Object.configureScope
    callOnHub('configureScope', callback);
  File "D:\PersonalDocuments\RiT\Projects\coding-challenge-backend-
nodejs\node_modules\winston-sentry-log\dist\index.js", line 83, in
Sentry.log
    this.sentryClient.configureScope((scope) => {
...
(29 additional frame(s) were not displayed)
```

SENTRY

Weekly Update for Renish
May 6th, 2019 – May 13th, 2019

Events Seen This Week

All: 196 Resolved: 0

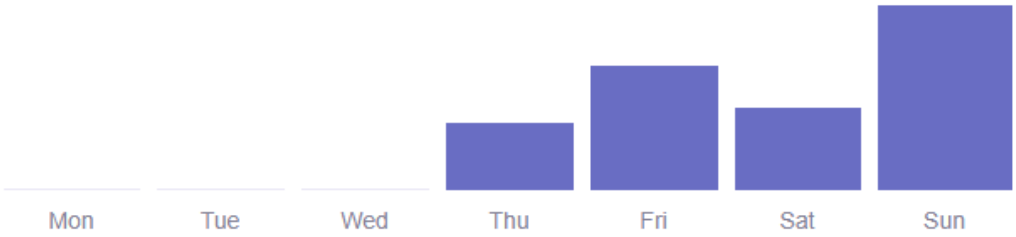
There is not enough data to compare to last week.

300%
more than four week average

Events by Issue Type

New: 98.0% Reopened: 0.0% Existing: 2.0%

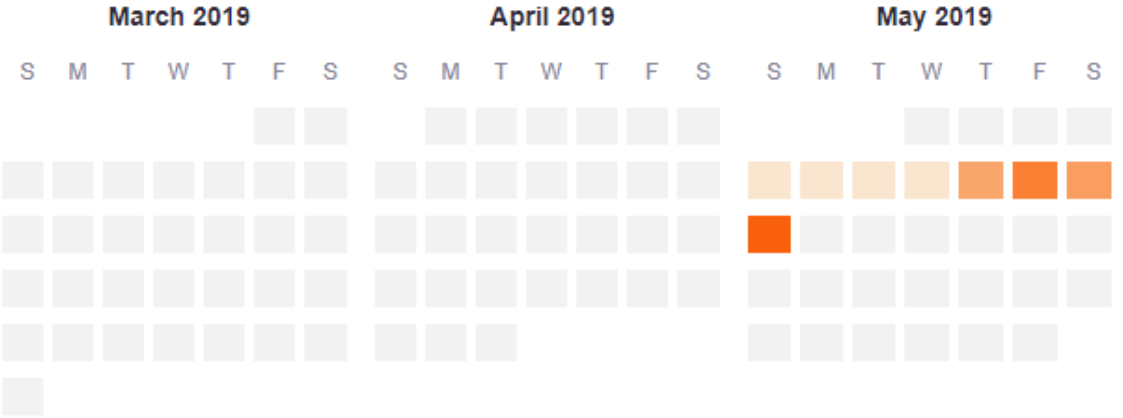
Events by Project



	EVENTS	PROJECT	FILTERED	RATE LIMITED
	196	stolenbike	0	0

3 Month History

Fewer Events More Events



Error Handling

- Used Winston logger with console and added Sentry.io via transport method.
- Used Volleyball for Http request logging in console.
- Made a custom wrapper around logging and made it globally available.

Validation

Joi:

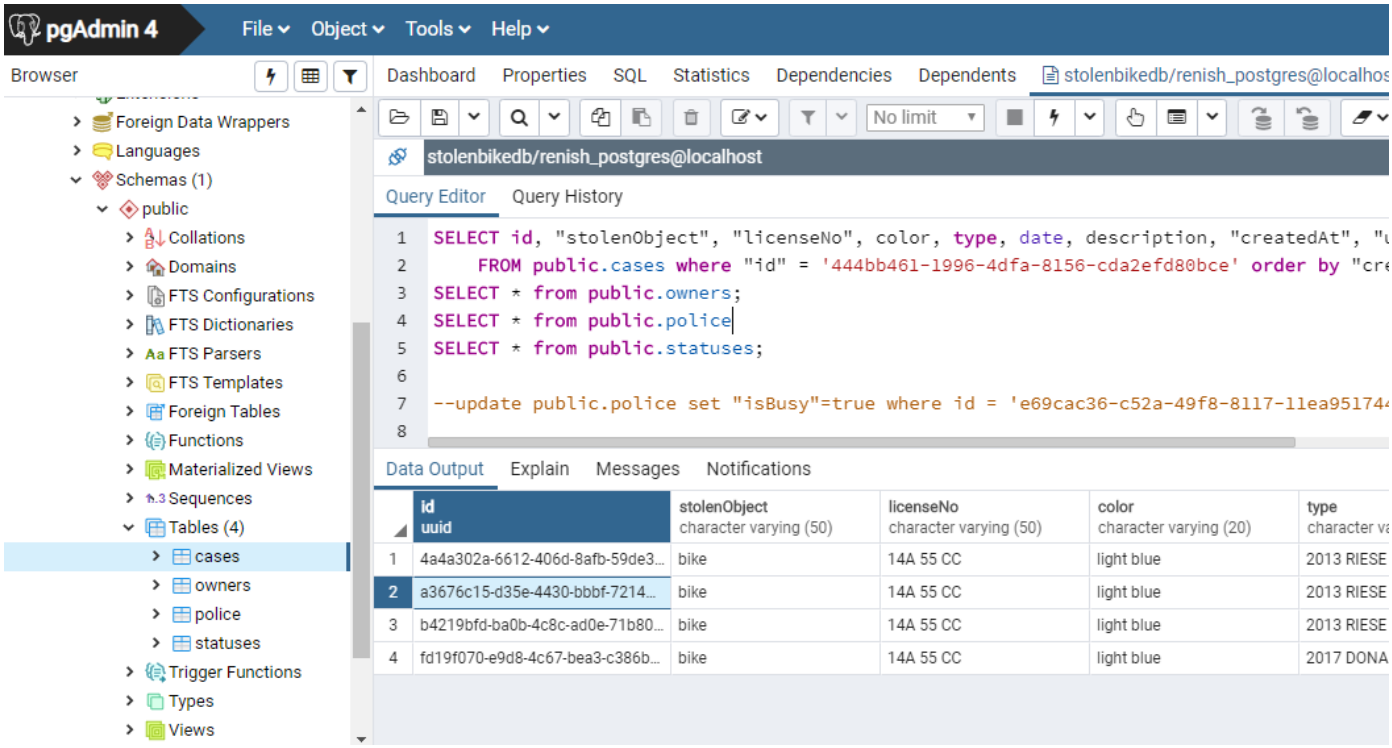
- Joi is an object schema description language and validator for JavaScript objects.
- Used it for validating the user's form input.

DB Model validation:

- In addition to Joi, I've added DB validation as well.
- Added via Sequelize feature.

Sequelize ORM & Postgres

- I've been using MongoDB for all my Node projects and because of this challenge I had gone through the Sequelize ORM & Postgres.
- My past experience with .NET Entity Framework (ORM) helped me to grasp the content quickly.



Indexing:

- Added indexing as part of performance enhancement through Sequelize modeling

Type	Name	Restriction
Index	public.police_ext_police_id	auto
Index	public.police_first_name	auto
Index	public.police_first_name_last_name	auto
Index	public.police_is_busy	auto
Index	public.police_last_name	auto
Primary Key	public.police_pkey	auto
Unique Constraint	public.police_extPoliceId_key	auto
Foreign Key	public.cases.cases_policeId_fkey	normal

Unit Testing:

- I've written a very basic test cases (only happy path scenarios).
- Used Mocha, Chai & Chai-http libraries
- Also implemented Istanbul test coverage tool. (having an issue with this).

```
  API endpoint /cases
StolenBike Log [2019-05-13 17:18:36]: info -> Database: Connection has
    √ should create a Case (234ms)
StolenBike Log [2019-05-13 17:18:36]: info -> Listening on port 8080
    √ should return all Cases
2013 undefined
    √ should return searched Cases
    √ should return a Case
    √ should update a Case
    √ should resolve a Case
    √ should delete a Case

  API endpoint /police
    √ should create a Police (73ms)
    √ should return all Police
    √ should return a Police
    √ should update a Police
```

API Documentation:

- Tried adapting Open API specification
- Used Swagger JSON for documentation

Swagger
Supported by SMARTBEAR

Explore

Berlin Stolen Bike API Service 1.0.0

[Base URL: stolen-bikes-svc.herokuapp.com/api/v1]

API documentation for Berlin Stolen Bike API Service (JOIN.COM - Coding Challenge)

Schemes
HTTP

case Handles case reporting, assigning/resolving, searching & CRUD operations

- GET /cases
- POST /cases
- GET /cases/search
- GET /cases/:caseId
- PUT /cases/:caseId
- DELETE /cases/:caseId
- PATCH /cases/:caseId/resolve

police Handles policemen, auto case assigning & CRUD operations

- GET /police
- POST /police
- GET /police/:policeId
- PUT /police/:policeId
- DELETE /police/:policeId

status Handles status CRUD operations

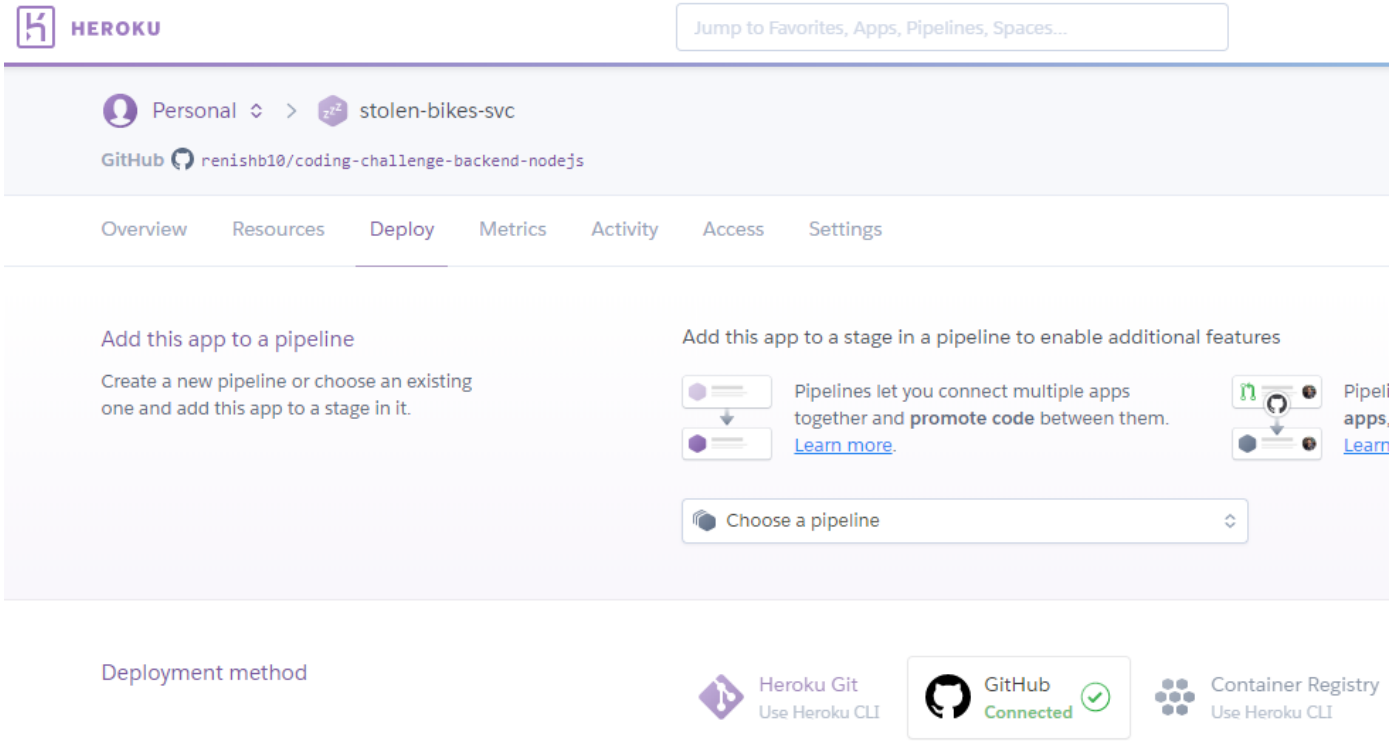
- GET /statuses
- POST /statuses
- PUT /statuses/:statusId
- DELETE /statuses/:statusId

Models

- Case
- Police
- Owner
- Status

Deployment

- Initially used Google Cloud, then later shifted to Heroku.
- Heroku's app -> db configuration and cluster-in access win-over GCM for small demo projects.



Scope for future enhancement

- Database tables can be expanded & normalized further if there is a scope for owner/police profile maintenance.
- Input data sanitization can be improved using Joi's feature.
- Case can hold additional status called 'reopened' can be brought in for history and tracking purpose.
- Case & police assignment can be archived so that cases table will not overloaded.
- Test cases should be improved, need to write negative and edge cases.
- Test coverage tool Istanbul can be replaced if not compatible with Chai-Http.
- Search functionality can be improved by bringing in more filters.
- Sorting can be implemented.
- Pagination support (Limit and Skip params).

Are some of the improvements that I think we could apply. Kindly let me know if anything needs to be implemented or enhanced. Ready to do it.

Thanks for the opportunity!

RENISH B