Handover

About

This document serves as a formal handover report of the project to another development team to take over and continue if needed.

- Deployment Guideline
- Final Presentation
- Final Release ZIP File
- Development Change Log

Background

Coaching-Mate

The **CoachingMate** is a system for the coach to deliver training sessions to their athletes. And it's now we are specifically required to enable Athletes to review all their Activities and Activity Details through Coaching Mate Web Application

Garmin

The **Garmin Connect** is the tool for Tracking, Analyzing and Sharing health and fitness activities from **Wearable Devices**. So Athletes with Devices like **Garmin Watch** will have their activity data sync to the **Garmin Connect**. And they provide an API that enables us to access those activity data.

- Garmin API
- Garmin Activity API
- Flexible and Interoperable Data Transfer (FIT) SDK

Project Goal

the **GOAL** of our project is about **data integration** with **Garmin API** and syncing the Activity data to the **Coaching-Mate Dashboard**. Our **Coaching-Mate** project is NOT built from scratch, and we are working on the code from the Previous **Garmin API Project**. So our main focus is to maintain and improve the existing platform. And we need to guarantee the data integration from GARMIN and ensure the dashboard display and retrieve the list of activity and activity details from **Garmin Connect**.

System Implementation

Our Scope

The Garmin API is what our project will be focusing on. Garmin is a commercialized product that targets clubs and coaches. Garmin API provides an easy way to integrate a platform directly into their content management website. This project needs to improve the APIs from the existing system, to increase the breadth of activities that are currently pulled by the CoachingMate backend.

Garmin should also be able to connect to external websites so that the workout data can be viewed by other people. Our team is also recommended to develop APIs for Garmin to connect to other external websites.

1. coaching-mate account management
    a. Account register
    b. Account log-in
    c. and log-out
2. Connect **CoachingMate Web Application** with the **Garmin Connect** API
3. Show data from **Garmin Connect** to the Dashboard of the **CoachingMate Web Application**
    a. List of Activities
    b. Activity Details

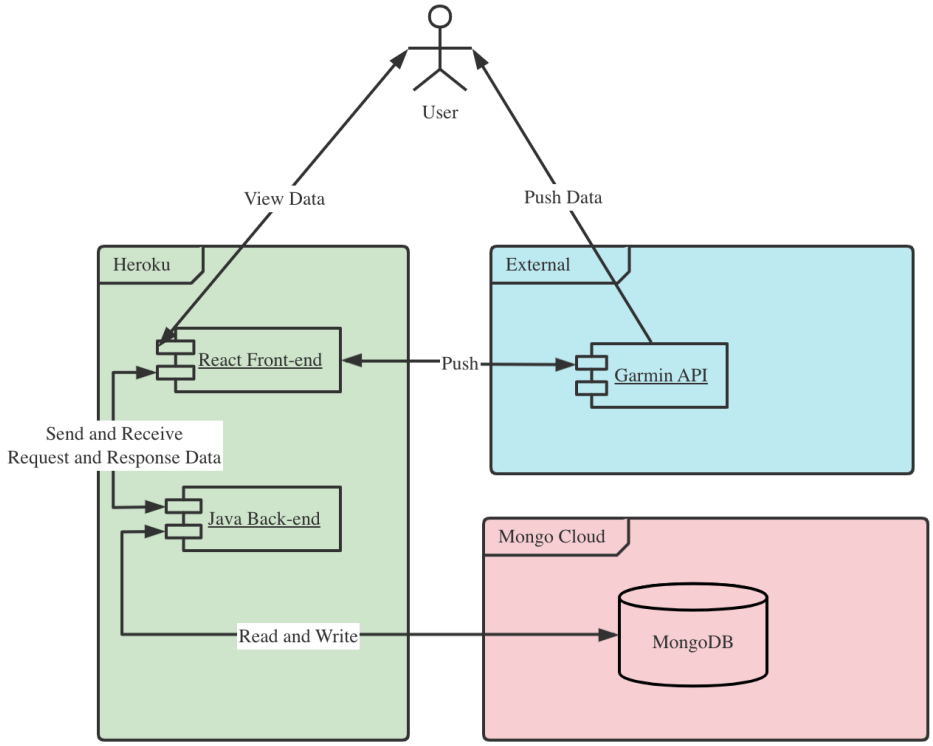
Out of Scope

Our team will not be responsible for data analytics. There are already a lot of functions on the Garmin dashboard. The GA teams will not be involved with the website design. This project will not be involved with how to visualize the data either.

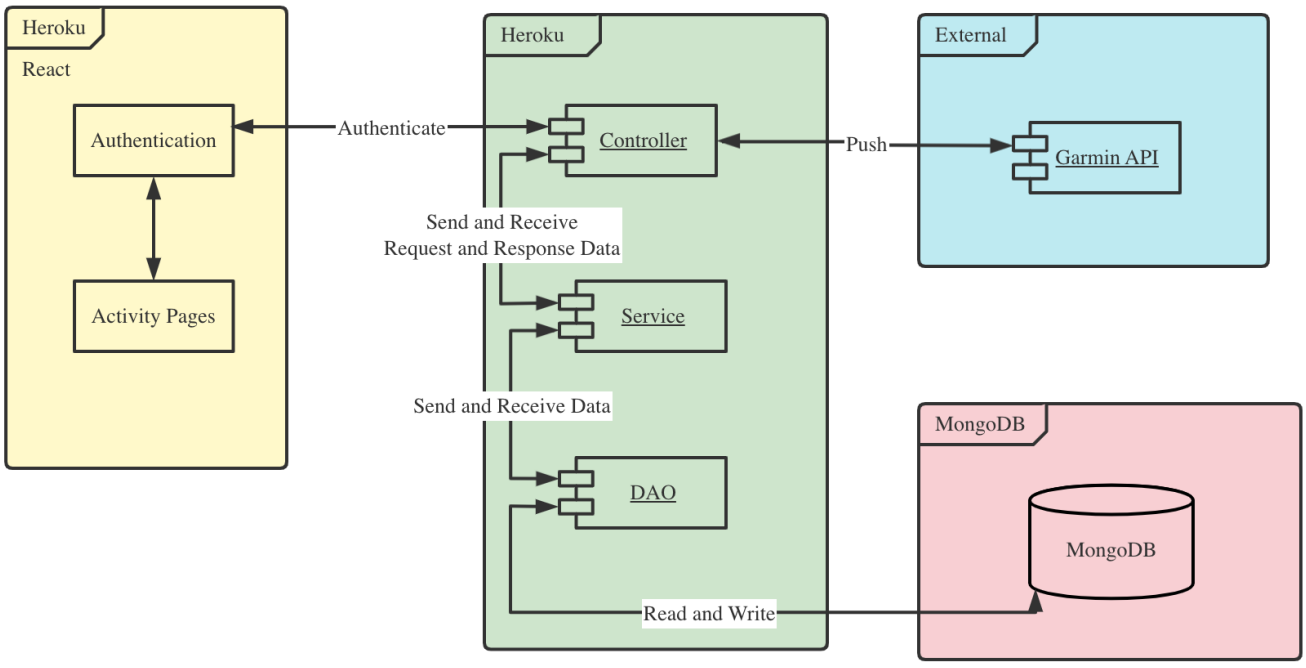
1. Frontend UI Design
2. Data Analysis

System Architecture

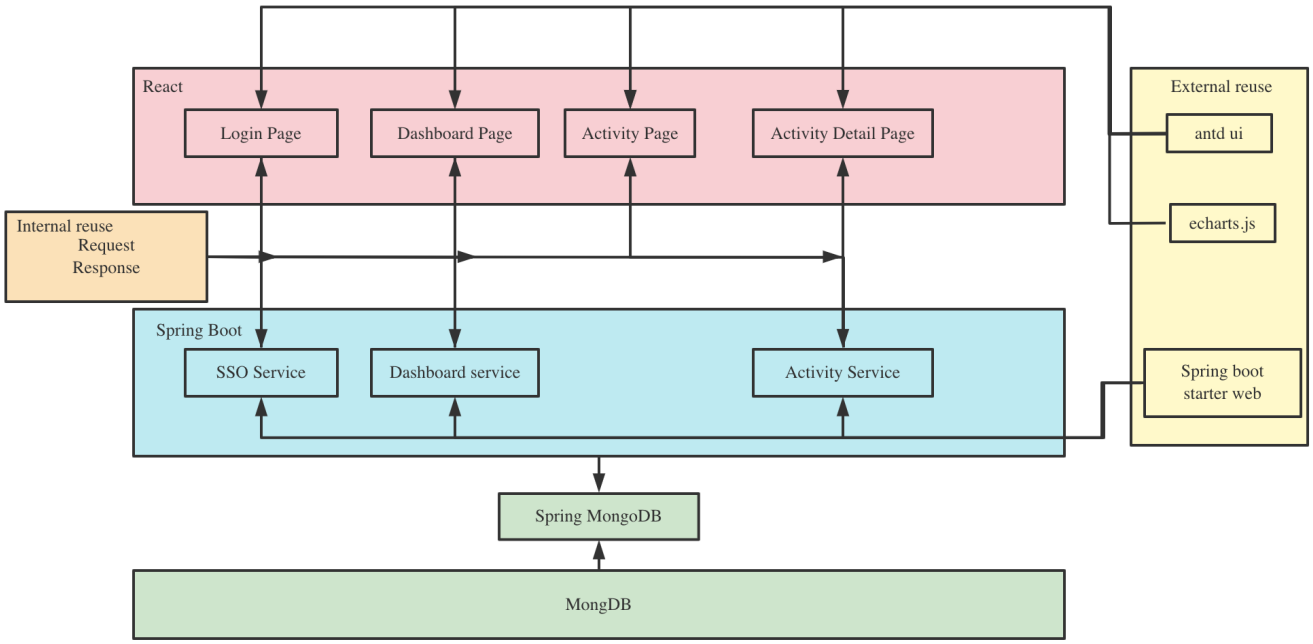
Components diagram



Deployment



System Logical Architecture



Deployment

- Deployment
- Development Change Log

Development Details

API

We made a major update to the front-end and back-end interactions. In the existing system, there is no unified front-end and back-end interaction specification, and error handling must be written every time. In the new system, the front-end and back-end API interactions will take the following format:

```
{
  "statusCode": 200,
  "message": "Success",
  "data": null
}
```

Field	Description
statusCode	The status code of this request, except 200, is a request error. The front end only needs to check this field to determine whether the request is successful.
message	The field for error requests. When the request is wrong, the front end only needs to display this message.
data	This field backend can hold any Java object. such as entity classes. Or the backend can return List<~>. If the backend wants to return different objects, it can use Map<String, Class>.

Frontend Code

GitHub Access

https://github.com/COMP90082SM12022/GA-Boxjelly/tree/master/src/cm-frontend-react

Code Structure

```
public                    # static sources
   favicon.ico            # Website LOGO
   index.html             # html Template
src                       # Frontend source code
   api                    # All requests
   assets                 # images, fronts
   components             # public componets
   config                 # global config
      menuConfig.js       # menu config
      routeMap.js         # router config
   lib                    # others resources
   mock                   # mock data for testing
   store                  # global store
   styles                 # global styles
   utils                  # global tool methods
   views                  # views all pages
   App.js                 # entry page
   defaultSettings.js     # default setting
   index.js               # source entry
.env.development          # development config file
.env.production           # production config file
config-overrides.js       # custom config for webpack of CRA
deploy.sh                 # CI deployment script
.travis.yml               # auto CI config
package.json              # package.json
```

Important files or dirs

- src/api/*.js: All backend APIs are declared here, and other files only need to call the declared functions.
- src/config/menuConfig.js&routeMap.js: Menu and permissions configuration files, menus and permissions need to be configured here before they can be accessed.
- src/store: user information and cookies
- src/views: Pages. After creating a new page file, you need to configure it in the routing before you can access it.
- src/utils/request.js: Advanced encapsulation of Axios, introduced in api/*.js. Contains global exception handling.

Backend Code Structure

GitHub Access: https://github.com/COMP90082SM12022/GA-Boxjelly/tree/master/src/coaching-mate

```
docs                                              # Some documents
lib                                               # other libraries
backup                                            # Backup files
src                                               # Source code
   main                                          # Product source code
      java/coachingmateanalytics/coachingmate
         common                                  # Public components
            config                               # Config files (Including http header config, swagger config, etc.)
            intercepter                          # Session and web intercepter
            utils                                # Request and Response classes
         controller                               # Interact with front-end
         dao                                      # Interact with database
         entity                                   # Entity classes
         service                                  # Handle bussiness logic
         CoachingmateApplication.java             # Application entry
      resources                                    # Config files
   test                                            # Test code
pom.xml                                           # Java library
```

Important files or dirs

- src/main/java/coachingmateanalytics/coachingmate/common/utils: public components. Contains Response, ErrorResponse. Response /ErrorResponse: Follow the Restful API specification.
- src/main/java/coachingmateanalytics/coachingmate/common/utils/ResponseHandler.java: Global return processing. All return values will be encapsulated as Response or ErrorResponse through this method.
- src/main/java/coachingmateanalytics/coachingmate/common/enums: Response enumeration class. Response contains status code and message, which are set here.
- src/main/java/coachingmateanalytics/coachingmate/common/annotation: Custom annotations. Existing annotation: AuthCheck, used to check whether the token is valid.