Outline

This project is a fitness tracking app that allows users to enter their fitness achievements and track their fitness progress. Registered users are able to access personalised features of the application. The main feature is logging in their physical activities which may include walking, running, swimming, gym etc. They are able to log additional details of their activity such as the duration, date and any additional note that they would like. These activities can be viewed for future reference in a list and can be searched through using keywords.

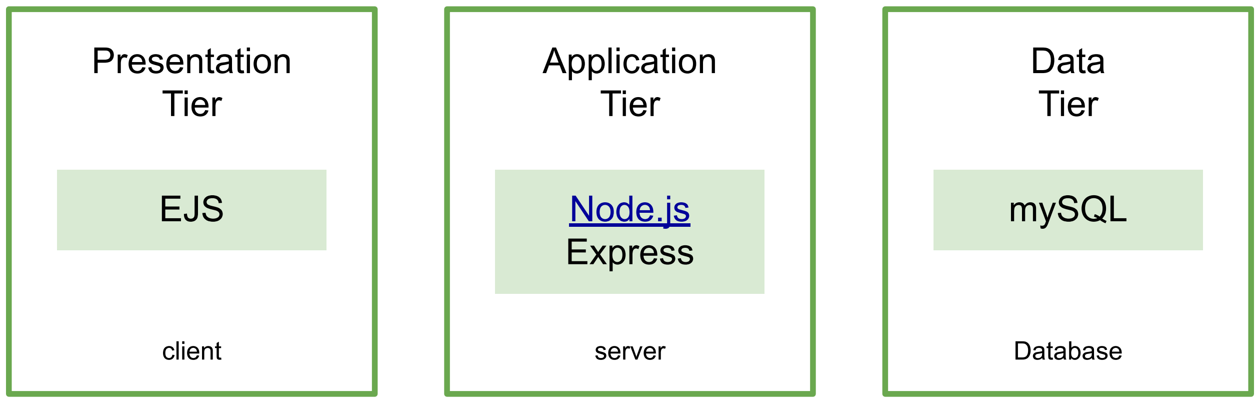
This application has been built using Node.js, Express, EJS and mySQL. Node.js and Express were used to handle the server side logic, EJS provides templates to develop dynamic HTML markup and mySQL was used for managing relational databases. This application includes user authentication which is handled using server side sessions which means that certain features of the application can only be accessed by registered users. It also uses Goldsmiths virtual machine where it is deployed.

FitApp is an application that uses user authentication, databases, servers and virtual machines.

Architecture

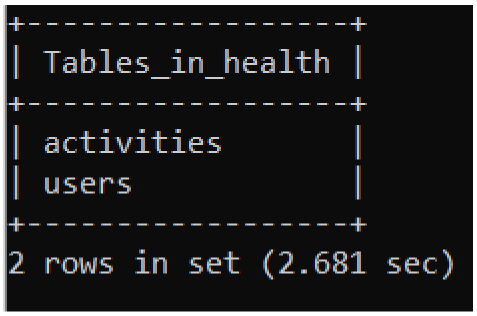
FitApp follows a three-tiered web architecture which consists of an application tier, presentation tier and data tier.

The application tier purpose is to manage processing and the business logic of the application using Node.js and Express. The presentation tier is the user interface of the application and is what is presenting and receiving input from the user. It is the layer the user directly interacts with. EJS templates are used in this tier. The data tier uses mySQL to handle all the relational databases, this includes storing user credentials and their personal activity records.



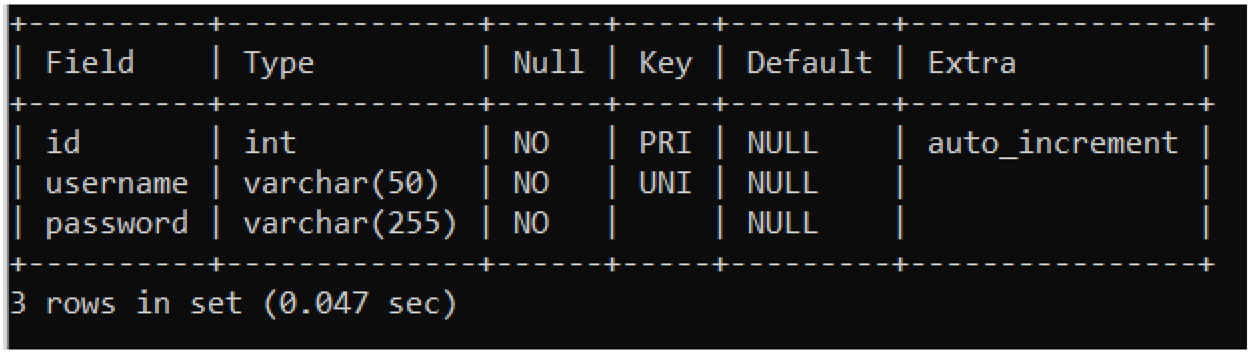
Data Model

In the Application FitApp, a database called health was created where it contained two relational tables, users and activities.



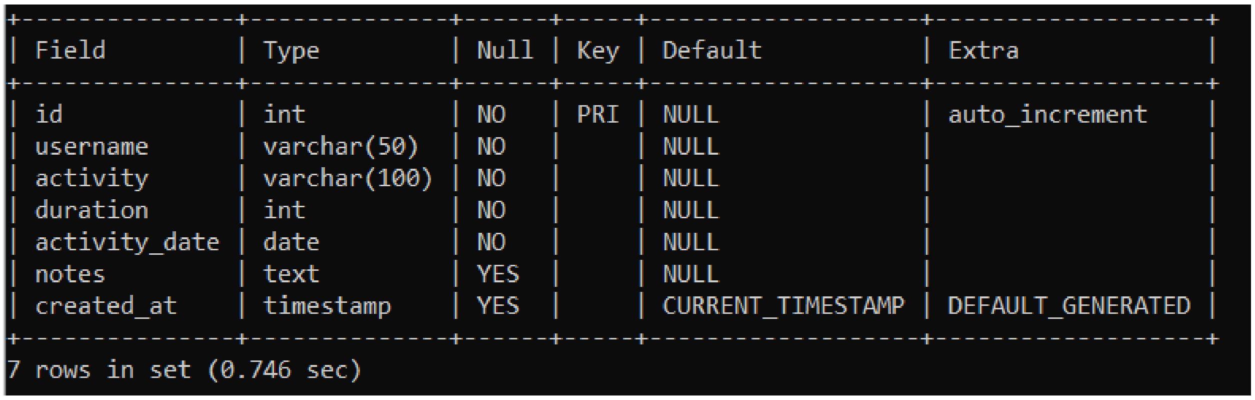
```
+------------------+
| Tables_in_health |
+------------------+
| activities       |
| users            |
+------------------+
2 rows in set (2.681 sec)
```

Table users: Handles all registered users credentials



```
+----------+--------------+------+-----+---------+----------------+
| Field    | Type         | Null | Key | Default | Extra          |
+----------+--------------+------+-----+---------+----------------+
| id       | int          | NO   | PRI | NULL    | auto_increment |
| username | varchar(50)  | NO   | UNI | NULL    |                |
| password | varchar(255) | NO   |     | NULL    |                |
+----------+--------------+------+-----+---------+----------------+
3 rows in set (0.047 sec)
```

Table activities: This is where the user's physical activity logs are stored.

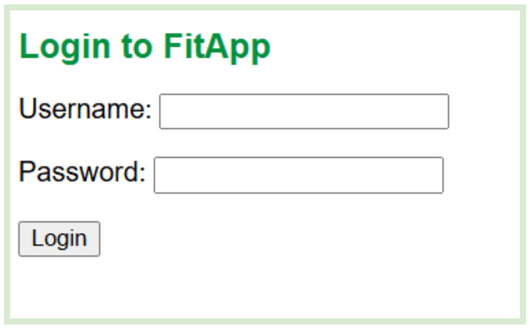


```
+---------------+--------------+------+-----+-------------------+-------------------+
| Field         | Type         | Null | Key | Default           | Extra             |
+---------------+--------------+------+-----+-------------------+-------------------+
| id            | int          | NO   | PRI | NULL              | auto_increment    |
| username      | varchar(50)  | NO   |     | NULL              |                   |
| activity      | varchar(100) | NO   |     | NULL              |                   |
| duration      | int          | NO   |     | NULL              |                   |
| activity_date | date         | NO   |     | NULL              |                   |
| notes         | text         | YES  |     | NULL              |                   |
| created_at    | timestamp    | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
+---------------+--------------+------+-----+-------------------+-------------------+
7 rows in set (0.746 sec)
```

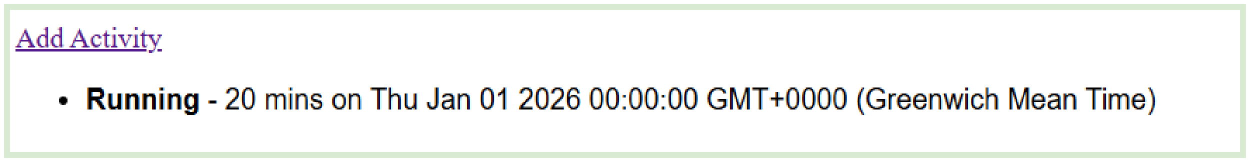
A single user can partake in many activities, therefore these two tables are linked with the username field. The table activities means that users can log the type of activity, duration, date, any additional notes and also record the timestamp of when the activity log has been created. Users have the option to record notes as it is an optional field in the form while everything else is required.

User Functionality

Login:
Users are able to access the login page and gain entry into their personal accounts using a username and password that is stored in the database. After successfully logging in, the user is directed to their activities page. If the wrong username or password is entered, an error message will be shown.

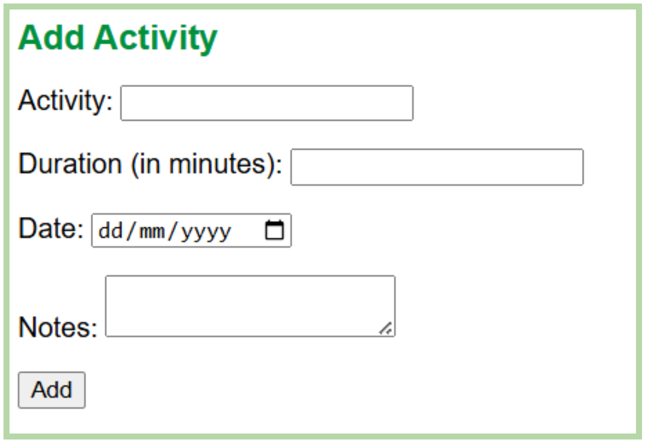


View Activities:
Like mentioned before, after logging in users are shown the activities page which shows a list of all of their previously recorded activities. This list is displayed in chronological order and will show the record of the signed in user. This page is only accessible to registered users. If the user is not signed in, it will redirect them to the login page.



- **Running** - 20 mins on Thu Jan 01 2026 00:00:00 GMT+0000 (Greenwich Mean Time)

Add Activity:
This page allows users to add a new activity through a form that collects the activity name, duration, date and optional notes. After submitting the form, the data is saved onto the database and linked to the signed in user. This page is also only accessible to registered users. If the user is not signed in, it will redirect them to the login page.

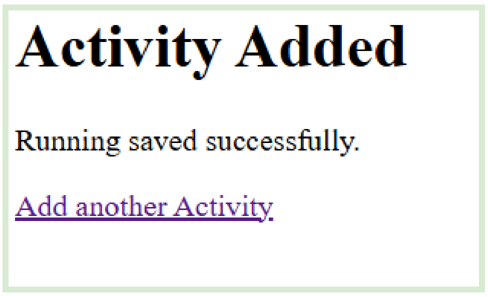

Add Activity

Activity:

Duration (in minutes):

Date: dd/mm/yyyy

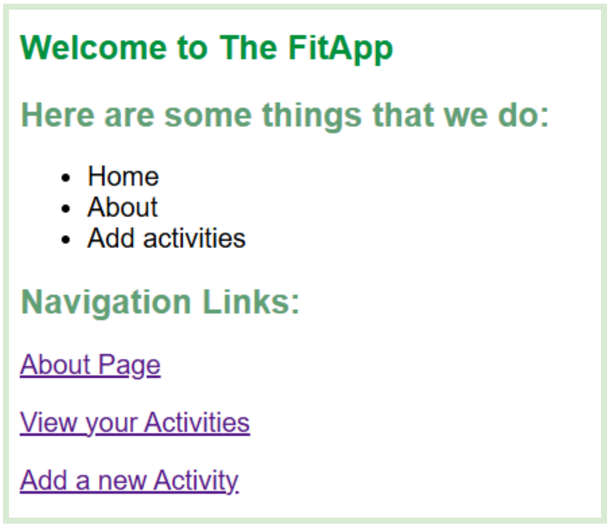
Notes:


Activity Added

Running saved successfully.

Add another Activity

Home Page:
The home page is the first page the user is introduced to and it is not required to be a signed in user to access this page. This page gives a quick summary on the application and provides navigation links to other pages of the application.


Welcome to The FitApp

Here are some things that we do:

- Home
- About
- Add activities

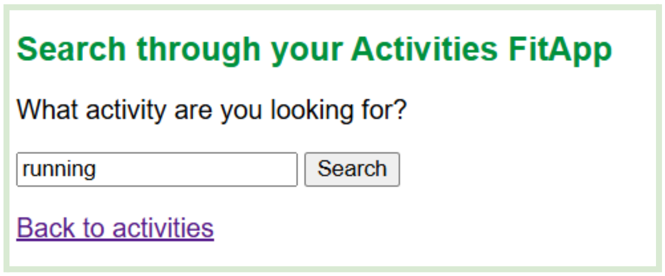
Navigation Links:

About Page

View your Activities

Add a new Activity

Search Activities:
Users are also able to search through their previously recorded activities through keywords. The search matches against the fields, activity name and notes field to find the results the user is looking for.


Search through your Activities FitApp

What activity are you looking for?

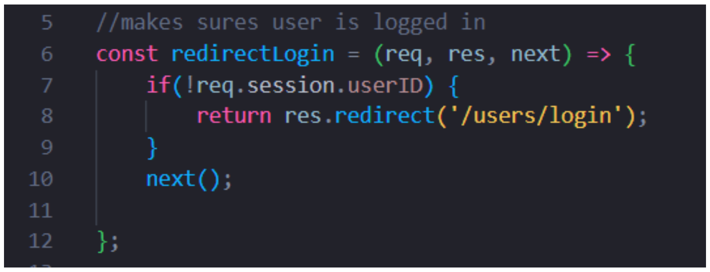
running

Back to activities

Logout:
Users can log out when they are finished with the session to prevent further access to protected pages.

Advanced Techniques

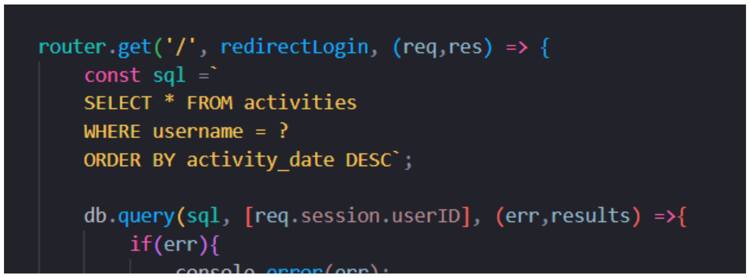
Session base Authentication:
This application uses session based authentication to secure restricted routes such as viewing previous physical activity records or recording new records. This was implemented into many routes in particular in the file activities.js. It uses express-session middleware to monitor authenticated users.



```
//makes sures user is logged in
const redirectLogin = (req, res, next) => {
    if(!req.session.userID) {
        return res.redirect('/users/login');
    }
    next();

};
```

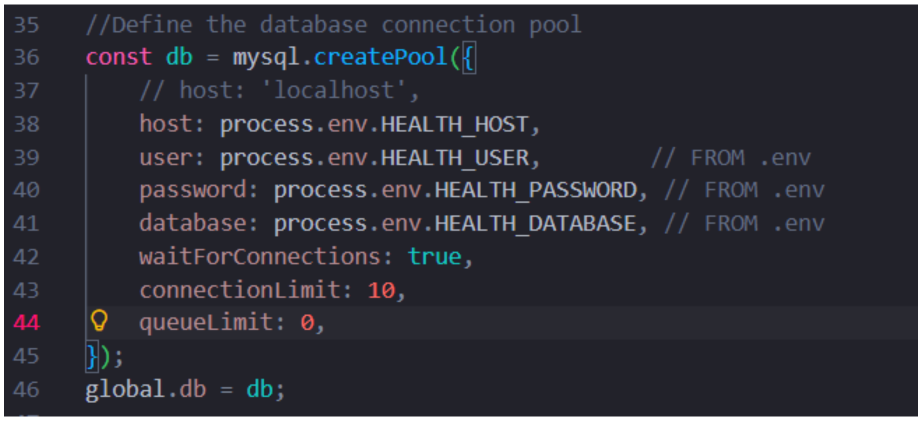
Secure Database Access:
All database interactions used prepared SQL statements which separate SQL logic and user input. This ensures that input cannot alter SQL queries.



```
router.get('/', redirectLogin, (req,res) => {
    const sql =`
    SELECT * FROM activities
    WHERE username = ?
    ORDER BY activity_date DESC`;

    db.query(sql, [req.session.userID], (err,results) =>{
        if(err){
```

Configuration:
Dotenv can be used to load environment variables that help manage database data and deployment. This allows it to be used both locally and on the Goldsmiths virtual machine without modifications.



```
//Define the database connection pool
const db = mysql.createPool({
    // host: 'localhost',
    host: process.env.HEALTH_HOST,
    user: process.env.HEALTH_USER,         // FROM .env
    password: process.env.HEALTH_PASSWORD, // FROM .env
    database: process.env.HEALTH_DATABASE, // FROM .env
    waitForConnections: true,
    connectionLimit: 10,
    queueLimit: 0,
});
global.db = db;
```

AI Declaration

AI tools were only used to assist with debugging errors and understanding key concepts.