# Web Programming

## Project Report

Group:

Noman Ali (17k-3652)

Mubashir Hashmi (17k-2394)

Date of Submission:

21.December.2020

Live Hosting URL:

## Table of Contents

**Assignment #2**
**Instructor: Engr. Abdul Rahman**
**Web Programming, 08-10-2020**

Students have to create **4 new web pages** for your personal portfolio, utilizing only HTML and CSS and JavaScript skills.

### (A) cv.html [New page 1]

Your HTML5 microdata enabled CV (Resume – 1 page but detailed and should be very cool looking and nicely ordered under different heading that you can think of.)
https://schema.org/docs/gs.html

### (B) intro.html [New page 2]

A brief introduction and biography of the candidate. This should be real and not fictitious. It shall include the following headings: (1) Introduction, (2) Why I'm Different, and (3) What I can contribute.

### (C) project_proposal.html [New page 3]

Description of your term project proposal, document its features and development tasks and time line (Gantt chart), and estimate project development cost in terms of Rupees. Document name of group members (not more than 3 – cross section groups not allowed). Project should be based on Management Information System such as hospital management system, inventory management, bank management system, transport management etc. Avoid creating online book store or other shopping carts. Project should contain 15 core domain features (Note: login / logout/ dashboard is not core domain features). For example, online flight booking is a core domain feature of Travel and Tour planning management system.

### (D) index.html [New page 4]

Main home page. This page is accessed by http://yourdomain.com/index.html. You may use any free PHP/HTML /MySQL based hosting provider (but without adds). This page shall contain: Famous / known inspirational quotation, Your name, your photograph, links to cv.html, intro.html, validation.html, readme.html and project_proposal.html.

**(E) readme.html: [Assignment 1 old page]** Create a README page that contains the step by step information about deployment, installation and configuration of your home page. Explain in detail how can someone install and configure your web site on his own web server. Document details about your hosting account, CPanel, FTP details etc. For this page use different styles for normal text, and monospaces font style for commands / code illustrations.

**(F) validation.html:** Create a Validation page that contains the output results from validator.w3.org and jigsaw.w3.org/css-validator of your web site.

**(G) Do's:** You need to follow following standards:

All HTML pages must pass HTML5 specifications via validator.w3.org. Use at least 1 CSS file. CSS must pass the official CSS validation via jigsaw.w3.org/css-validator. File names must be in lowercase and only [a-z0-9_.] characters can be used for file names. Use file names without any whitespaces. Proper indentation of the HTML code is required. Variable names should be in camel case. All content must be stored in your hosting site. All HTML pages must contain search engine friendly HTML meta-name keyword tags. Exclude your CV page from search engines by using ROBOTS.TXT. (Instructions for ROBOTS.TXT: www.robotstxt.org/robotstxt.html). CV page must be password protected (use htaccess or hosting provider's method). ***Use attractive styles and create good looking web pages.*** You can utilize the US web design system (USWDS) guidelines. But if you are confident in website design then you can go ahead without it. USWDS is not mandatory for this task.

**(H) Don't:** Don't use HTML Frames. Don't use any styling attribute in elements (e.g., <h3 align="center">). Don't use any styling element in html file (e.g., <strong>, <em>, <font>, <center>). All styling should go into CSS. Don't' absolute URLs for any resources and images. Don't use any Front-end frameworks such as Bootstrap. Don't use any publicly available CSS files / or other people CSS files. Don't use JavaScript / jQuery. Don't use any web site generator.

## Description:

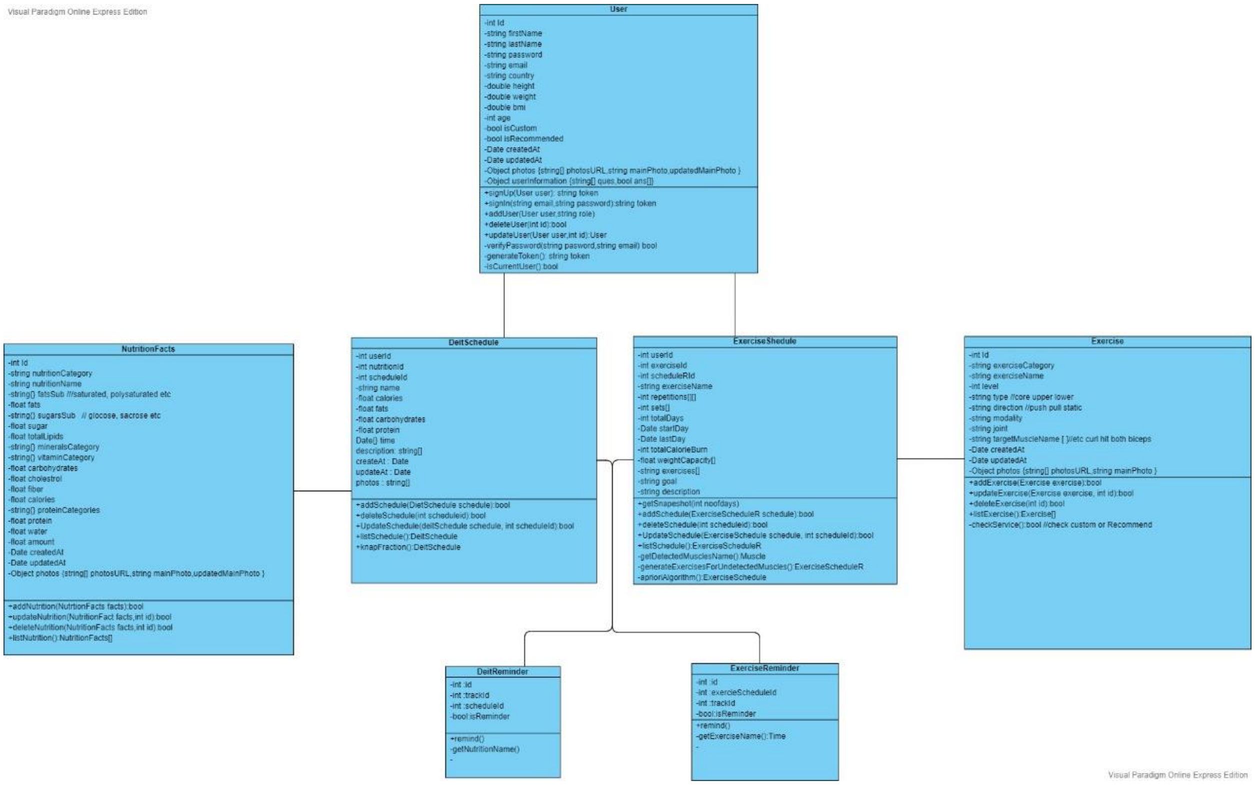
The project of Web Programming is developed to manage the Final Year Project. The project name is Diet And Exercise Recommendation System. It is a mobile application in our FYP. But in the Web Programming semester project, we develop the web application using React JS. In our project, we recommend users a diet and exercise plan by taking some information from the user. We take information from users in two ways. Either he can take photos of his physique or he can answer some questions.

The management of the system involves the users,diets, exercises, diet schedule and exercise schedule lists management. We are using mongoDB and microservices architecture on backend and REACT JS on frontend.

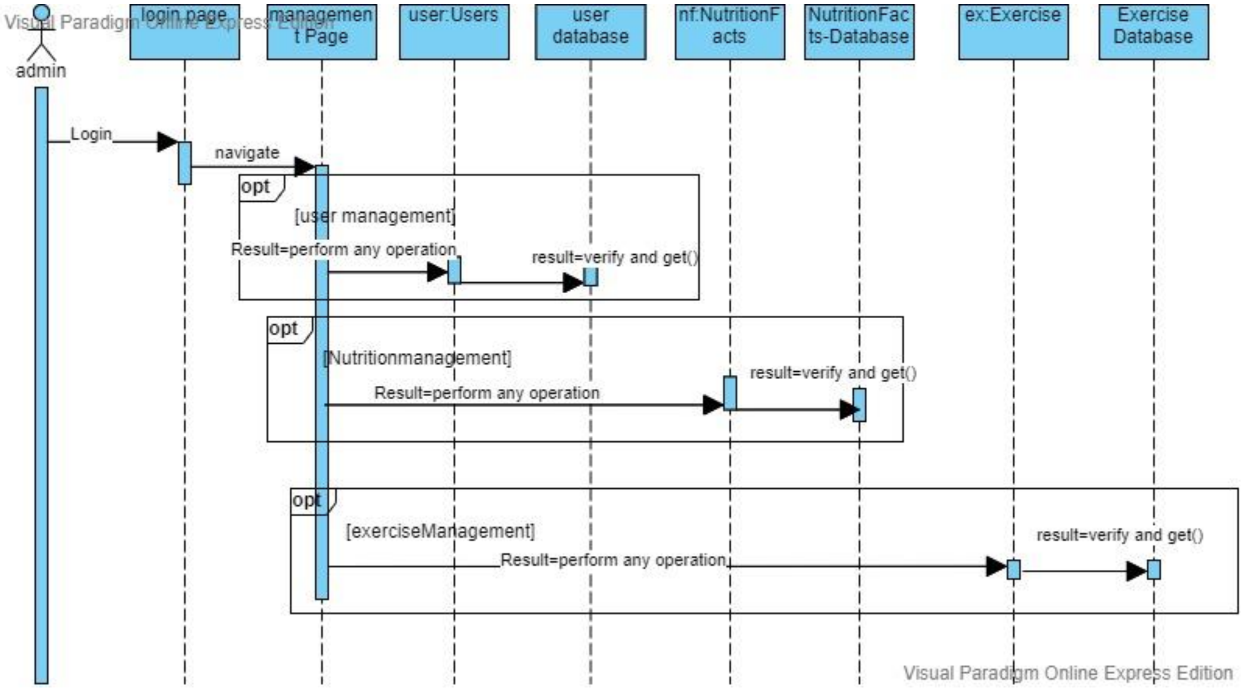
## Features:

1. Add Exercise
2. Delete Exercise
3. Update Exercise
4. Retrieve Exercise
5. Add Diet
6. Delete Diet
7. Update Diet
8. Retrieve Diet
9. Add User
10. Delete User
11. Update User
12. Retrieve User
13. Add Custom Diet Schedule
14. Delete Custom Diet Schedule
15. Add Custom Exercise Schedule
16. Delete Custom Exercise Schedule

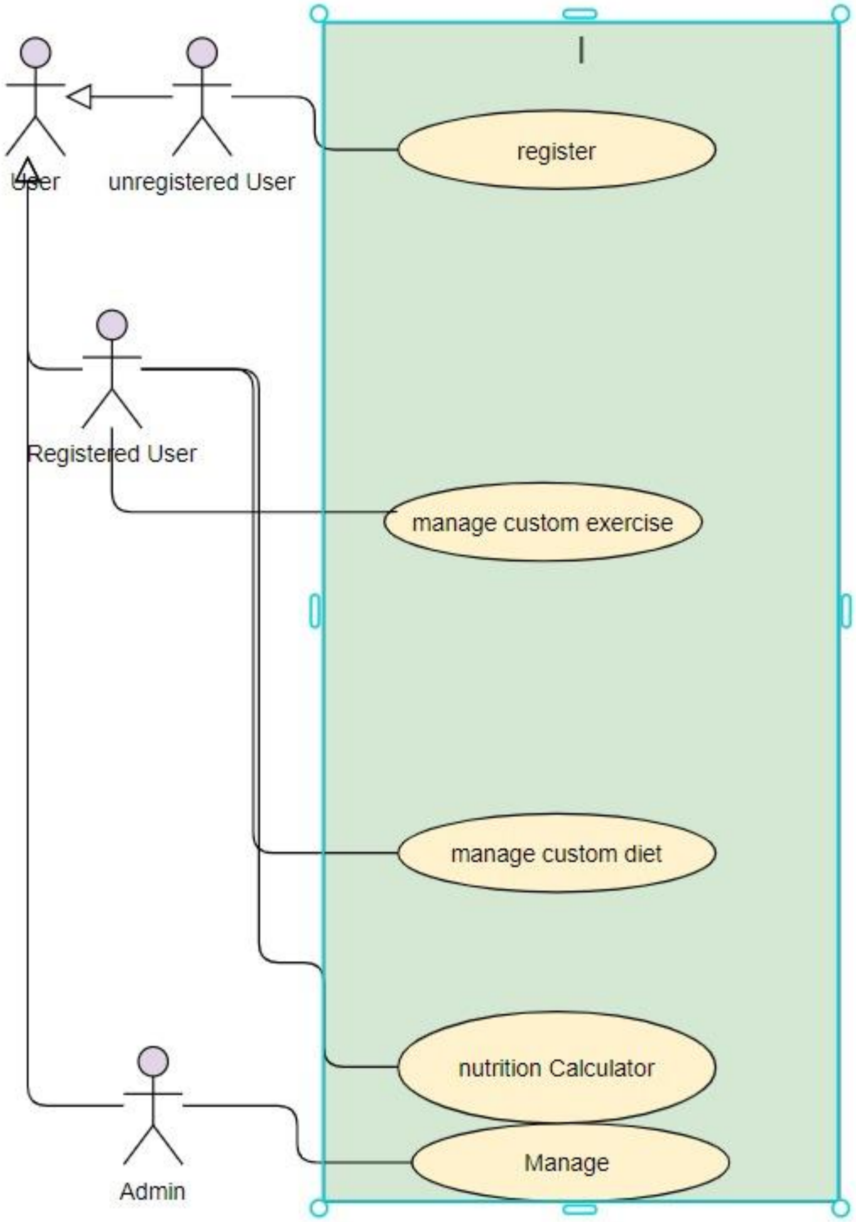
## Class Diagram


Visual Paradigm Online Express Edition
User
-int Id
-string firstName
-string lastName
-string password
-string email
-string country
-double height
-double weight
-double bmi
-int age
-bool isCustom
-bool isRecommended
-Date createdAt
-Date updatedAt
-Object photos {string[] photosURL,string mainPhoto,updatedMainPhoto }
-Object userInformation {string[] ques,bool ans[]}
+signUp(User user): string token
+signIn(string email,string password):string token
+addUser(User user,string role)
+deleteUser(int id):bool
+updateUser(User user,int id):User
-verifyPassword(string pasword,string email) bool
-generateToken(): string token
-isCurrentUser():bool
NutritionFacts
-int Id
-string nutritionCategory
-string nutritionName
-string[] fatsSub ///saturated, polysaturated etc
-float fats
-string[] sugarsSub // glocose, sacrose etc
-float sugar
-float totalLipids
-string[] mineralsCategory
-string[] vitaminCategory
-float carbohydrates
-float cholestrol
-float fiber
-float calories
-string[] proteinCategories
-float protein
-float water
-float amount
-Date createdAt
-Date updatedAt
-Object photos {string[] photosURL,string mainPhoto,updatedMainPhoto }
+addNutrition(NutrtionFacts facts):bool
+updateNutrition(NutritionFact facts,int id):bool
+deleteNutrition(NutritionFacts facts,int id):bool
+listNutrition():NutritionFacts[]
DeitSchedule
-int userId
-int nutritionId
-int scheduleId
-string name
-float calories
-float fats
-float carbohydrates
-float protein
Date[] time
description: string[]
createAt : Date
updateAt : Date
photos : string[]
+addSchedule(DietSchedule schedule):bool
+deleteSchedule(int scheduleid):bool
+UpdateSchedule(deitSchedule schedule, int scheduleId):bool
+listSchedule():DeitSchedule
+knapFraction():DeitSchedule
ExerciseShedule
-int userId
-int exerciseId
-int scheduleRId
-string exerciseName
-int repetitions[][]
-int sets[]
-int totalDays
-Date startDay
-Date lastDay
-int totalCalorieBurn
-float weightCapacity[]
-string exercises[]
-string goal
-string description
+getSnapeshot(int noofdays)
+addSchedule(ExerciseScheduleR schedule):bool
+deleteSchedule(int scheduleid):bool
+UpdateSchedule(ExerciseSchedule schedule, int scheduleId):bool
+listSchedule():ExerciseScheduleR
-getDetectedMusclesName():Muscle
-generateExercisesForUndetectedMuscles():ExerciseScheduleR
-aprioriAlgorithm():ExerciseSchedule
Exercise
-int Id
-string exerciseCategory
-string exerciseName
-int level
-string type //core upper lower
-string direction //push pull static
-string modality
-string joint
-string targetMuscleName [ ]//etc curl hit both biceps
-Date createdAt
-Date updatedAt
-Object photos {string[] photosURL,string mainPhoto }
+addExercise(Exercise exercise):bool
+updateExercise(Exercise exercise, int id):bool
+deleteExercise(int id):bool
+listExercise():Exercise[]
-checkService():bool //check custom or Recommend
DeitReminder
-int :id
-int :trackId
-int :scheduleId
-bool:isReminder
+remind()
-getNutritionName()
-
ExerciseReminder
-int :id
-int :exercieScheduleId
-int :trackId
-bool:isReminder
+remind()
-getExerciseName():Time
-
Visual Paradigm Online Express Edition

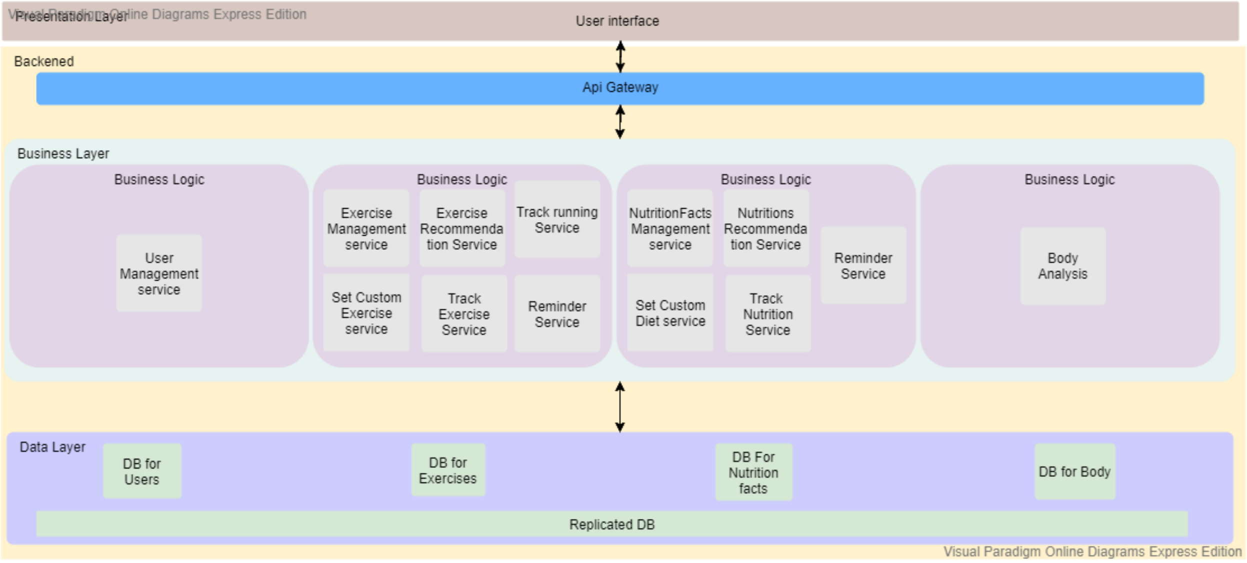
## Sequence Diagram



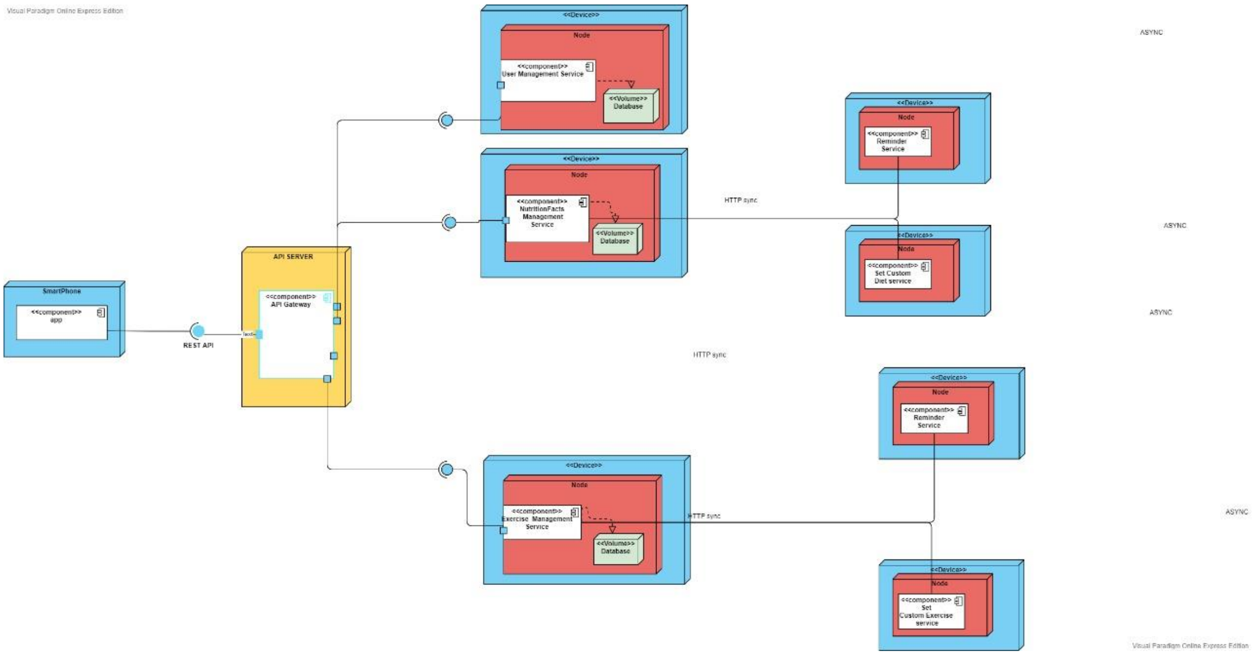
Use case Diagram



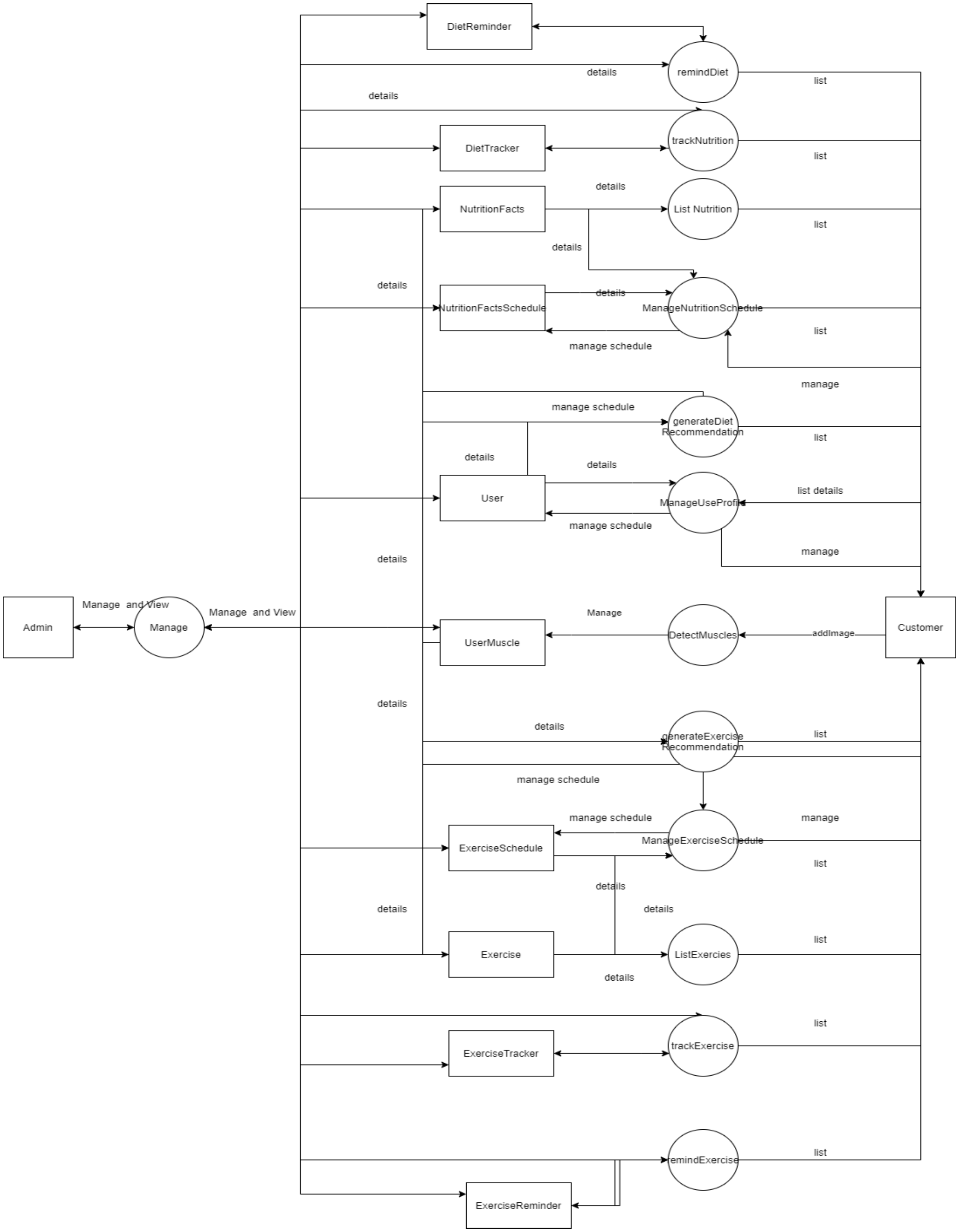
## Architecture Diagram



## Deployment Diagram


Visual Paradigm Online Express Edition
SmartPhone
<<component>> app
REST API
API SERVER
<<component>> API Gateway
<<Device>>
Node
<<component>> User Management Service
<<Volume>> Database
<<Device>>
Node
<<component>> NutritionFacts Management Service
<<Volume>> Database
HTTP sync
<<Device>>
Node
<<component>> Reminder Service
<<Device>>
Node
<<component>> Set Custom Diet service
ASYNC
ASYNC
ASYNC
HTTP sync
<<Device>>
Node
<<component>> Reminder Service
<<Device>>
Node
<<component>> Exercise Management Service
<<Volume>> Database
HTTP sync
<<Device>>
Node
<<component>> Set Custom Exercise service
ASYNC
Visual Paradigm Online Express Edition

# DFD


DietReminder
remindDiet
details
details
list
DietTracker
trackNutrition
list
NutritionFacts
details
List Nutrition
list
details
details
NutritionFactsSchedule
details
ManageNutritionSchedule
list
manage schedule
manage
manage schedule
generateDiet Recommendation
list
details
details
User
ManageUseProfile
list details
manage schedule
manage
details
Admin
Manage and View
Manage
Manage and View
UserMuscle
Manage
DetectMuscles
addImage
Customer
details
details
generateExercise Recommendation
list
manage schedule
manage schedule
manage
ExerciseSchedule
ManageExerciseSchedule
list
details
details
details
Exercise
ListExercies
list
details
list
ExerciseTracker
trackExercise
remindExercise
list
ExerciseReminder

## Data Dictionary

### Data 1

| | |
|---|---|
| **< Data 1>** | |
| **Name** | User |
| **Alias** | Write other names used for the first entry. |
| **Where-used/how-used** | during authorization, and using services  managing profile,recommending and custom |
| **Content description** | Notation for representing content. |

| **Column Name** | **Description** | **Type** | **Length** | **Null able** | **Default Value** | **Key Type** |
|---|---|---|---|---|---|---|
| id | id of user | string | [24] | False | | PK |
| firstName | first name of user | string | [24] | False | | |
| lastName | last name of user | string | [24] | False | | |
| password | password of user for login | string | [200] | False | | |
| email | email of user | string | [30] | False | | |
| country | country of user | string | [20] | True | *NULL* | |
| height | height of user | float | 1 | False | | |
| weight | weight of user | float | *1* | False | | |
| bmi | bmi of user | float | *1* | False | | |
| age | age of user | int | *1* | False | | |
| isCustom | true if user customized schedule | bool | *1* | True | NULL | |
| isRecommend | true if user use recommendation schedule | bool | *1* | True | NULL | |
| createAt | date of account creation | Date | *1* | False | | |
| updateAt | date of account update | Date | *1* | False | | |
| photos | photos of user | Array | [*] | True | NULL | |

| | | | | | | |
|---|---|---|---|---|---|---|
| userInformation | information of user | Array | [*] | True | NULL | |

### Data 2

<table>
<tr><th colspan="7">< Data 2></th></tr>
<tr><td colspan="2"><b>Name</b></td><td colspan="5">Exercise</td></tr>
<tr><td colspan="2"><b>Alias</b></td><td colspan="5">Write other names used for the first entry.</td></tr>
<tr><td colspan="2"><b>Where-used/how-used</b></td><td colspan="5">recommending and making custom  schedules</td></tr>
<tr><td colspan="2"><b>Content description</b></td><td colspan="5">Notation for representing content.</td></tr>
<tr><td colspan="7"></td></tr>
<tr><td><b>Column Name</b></td><td><b>Description</b></td><td><b>Type</b></td><td><b>Length</b></td><td><b>Null able</b></td><td><b>Default Value</b></td><td><b>Key Type</b></td></tr>
<tr><td>id</td><td>id of exercise</td><td>string</td><td>[24]</td><td>False</td><td></td><td>PK</td></tr>
<tr><td>exerciseCategory</td><td>to categories exercise into biceps, triceps, lats, etc</td><td>string</td><td>[20]</td><td>False</td><td></td><td></td></tr>
<tr><td>exerciseName</td><td>name of exercise</td><td>string</td><td>[20]</td><td>False</td><td></td><td></td></tr>
<tr><td>level</td><td>difficulty level of exercise</td><td>string</td><td>[20]</td><td>False</td><td></td><td></td></tr>
<tr><td>type</td><td>to check that exercise is upper, lower or core</td><td>string</td><td>[20]</td><td>False</td><td></td><td></td></tr>
<tr><td>direction</td><td>to check that exercise is pulling or pushing</td><td>string</td><td>[20]</td><td>False</td><td></td><td></td></tr>
<tr><td>modality</td><td>to check that exercise is free weight or cable or machine</td><td>string</td><td>[20]</td><td>False</td><td></td><td></td></tr>
<tr><td>joint</td><td>to check that exercise is multi joint or single joint</td><td>string</td><td>[20]</td><td>False</td><td></td><td></td></tr>
<tr><td>targetMusclesName</td><td>muscles affected by exercise</td><td>string</td><td>[*]</td><td>True</td><td>NULL</td><td></td></tr>
<tr><td>createdAt</td><td>date of creation</td><td>Date</td><td>1</td><td>False</td><td></td><td></td></tr>
</table>

| | | | | | | |
|---|---|---|---|---|---|---|
| updatedAt | date of updation | Date | *1* | False | | |
| photos | photos of exercise | Array | [*] | True | NULL | |

**Data 3**

| **< Data 3>** | | | | | | |
|---|---|---|---|---|---|---|
| **Name** | NutritionFacts | | | | | |
| **Alias** | Write other names used for the first entry. | | | | | |
| **Where-used/how-used** | recommending and making custom  schedules | | | | | |
| **Content description** | Notation for representing content. | | | | | |
| | | | | | | |
| **Column Name** | **Description** | **Type** | **Length** | **Null able** | **Default Value** | **Key Type** |
| id | id of nutrition | int | [24] | False | | PK |
| nutritionCategory | to categories in different in vegetables, diaries fruits etc | string | [24] | False | | |
| nutritionName | name of that nutrition | string | [24] | False | | |
| fatSub | to categories in different fat like saturated, polyunsaturated, and monoUnsaturated | string | [3] | False | | |
| sugar | sugar quantity in food | string | [20] | True | NULL | |
| totalLipids | lipids in nutrition | float | 1 | True | NULL | |
| mineralCategory | to categories in different minerals | float | [*] | True | NULL | |
| vitaminsCategory | to categories in different Vitamins | float | [*] | True | NULL | |
| proteinCategory | to categories in different protein | int | [*] | True | NULL | |

| | | | | | | |
|---|---|---|---|---|---|---|
| protein | protein in nutrition | bool | *1* | True | NULL | |
| carbohydrates | carbohydrates in nutrition | bool | *1* | True | NULL | |
| cholesterol | cholesterol in nutrition | Date | *1* | True | NULL | |
| fiber | fiber in nutrition | | *1* | True | NULL | |
| calories | calories in nutrition | Date | *1* | False | | |
| water | water needed in nutrition | | *1* | True | NULL | |
| amount | nutrition amount | | *1* | True | NULL | |
| createAt | nutrition creation date | Date | *1* | False | | |
| updateAt | nutrition last updated date | Date | *1* | False | | |
| photos | photos of nutrition | Array | [*] | True | *NULL* | |

**Data 4**

<table>
<tr><th colspan="7">&lt; Data 4&gt;</th></tr>
<tr><td colspan="2"><b>Name</b></td><td colspan="5">DietSchedule</td></tr>
<tr><td colspan="2"><b>Alias</b></td><td colspan="5">Write other names used for the first entry.</td></tr>
<tr><td colspan="2"><b>Where-used/how-used</b></td><td colspan="5">during diet remind anding diet tracking</td></tr>
<tr><td colspan="2"><b>Content description</b></td><td colspan="5">Notation for representing content.</td></tr>
<tr><td colspan="7"></td></tr>
<tr><td><b>Column Name</b></td><td><b>Description</b></td><td><b>Type</b></td><td><b>Length</b></td><td><b>Nullable</b></td><td><b>Default Value</b></td><td><b>Key Type</b></td></tr>
<tr><td>id</td><td>id of diet schedule</td><td>int</td><td>[24]</td><td>False</td><td></td><td>PK</td></tr>
<tr><td>nutritionId</td><td>id of nutrition</td><td>string</td><td>[24]</td><td>False</td><td></td><td>FK</td></tr>
<tr><td>userId</td><td>id of user</td><td>string</td><td>[24]</td><td>False</td><td></td><td>FK</td></tr>
<tr><td>nutritionName</td><td>name of nutrition</td><td>string</td><td>[20]</td><td>False</td><td></td><td></td></tr>
<tr><td>calories</td><td>calories on nutrition</td><td>float</td><td>1</td><td>False</td><td></td><td></td></tr>
<tr><td>fats</td><td>fats in nutrition</td><td>float</td><td>1</td><td>False</td><td></td><td></td></tr>
<tr><td>carbohydrates</td><td>carbohydrates in nutrition</td><td>float</td><td>1</td><td>False</td><td></td><td></td></tr>
</table>

| | | | | | | |
|---|---|---|---|---|---|---|
| protein | protein in nutrition | float | *1* | False | | |
| description | description of nutrition | string | [*] | True | NULL | |
| createAt | date of creation | Date | *1* | False | | |
| updateAt | last updation date | Date | *1* | False | | |
| photos | photos of nutrition | Array | [*] | True | NULL | |

Data 5

<table>
<tr><th colspan="7">&lt; Data 5&gt;</th></tr>
<tr><td colspan="2"><b>Name</b></td><td colspan="5">ExerciseSchedule</td></tr>
<tr><td colspan="2"><b>Alias</b></td><td colspan="5">Write other names used for the first entry.</td></tr>
<tr><td colspan="2"><b>Where-used/how-used</b></td><td colspan="5">during exercise reminding and exercise tracking</td></tr>
<tr><td colspan="2"><b>Content description</b></td><td colspan="5">Notation for representing content.</td></tr>
<tr><td colspan="7"></td></tr>
<tr><td><b>Column Name</b></td><td><b>Description</b></td><td><b>Type</b></td><td><b>Length</b></td><td><b>Nullable</b></td><td><b>Default Value</b></td><td><b>Key Type</b></td></tr>
<tr><td>id</td><td>id of schedule</td><td>int</td><td>[24]</td><td>False</td><td></td><td>PK</td></tr>
<tr><td>ExerciseId</td><td>id of exercise</td><td>string</td><td>[24]</td><td>False</td><td></td><td>FK</td></tr>
<tr><td>userId</td><td>id of user</td><td>string</td><td>[24]</td><td>False</td><td></td><td>FK</td></tr>
<tr><td>exerciseName</td><td>name of exercise</td><td>string</td><td>[20]</td><td>False</td><td></td><td></td></tr>
<tr><td>repetitions</td><td>reps of exercise</td><td>float</td><td>[10]</td><td>False</td><td></td><td></td></tr>
<tr><td>sets</td><td>sets of exercise</td><td>float</td><td>1</td><td>False</td><td></td><td></td></tr>
<tr><td>description</td><td>description of exercise</td><td>string</td><td>[*]</td><td>True</td><td>NULL</td><td></td></tr>
<tr><td>createAt</td><td>date of creation</td><td>Date</td><td><i>1</i></td><td>False</td><td></td><td></td></tr>
<tr><td>updateAt</td><td>date of updation</td><td>Date</td><td><i>1</i></td><td>False</td><td></td><td></td></tr>
<tr><td>photos</td><td>photos of exercise</td><td>Array</td><td>[*]</td><td>True</td><td>NULL</td><td></td></tr>
</table>

Data 6

| < Data 6> |
|---|

| | |
|---|---|
| **Name** | ExerciseTrack |
| **Alias** | Write other names used for the first entry. |
| **Where-used/how-used** | during exercise reminding |
| **Content description** | Notation for representing content. |
| | |

| **Column Name** | **Description** | **Type** | **Length** | **Null able** | **Default Value** | **Key Type** |
|---|---|---|---|---|---|---|
| id | id of tracking | string | [24] | False | | PK |
| ExerciseIdScheduleId | id of schedule | string | [24] | False | | FK |
| userId | id of user | string | [24] | False | | FK |
| weightCapacity | weight used by user | string | [*] | True | NULL | |
| totalRunning | Running of user | float | [*] | True | NULL | |
| totalCaloriesBurn | Calories burn by user | float | [*] | True | NULL | |
| repetitions | reps of user | float | [*] | True | NULL | |
| totalDays | number of days user do exercise | int | 1 | False | | |
| createAt | date of creation | Date | *1* | False | | |
| updateAt | date of updation | Date | *1* | False | | |

Data 7

| **< Data 7>** | |
|---|---|
| **Name** | DietTrack |
| **Alias** | Write other names used for the first entry. |
| **Where-used/how-used** | during diet remind |
| **Content description** | Notation for representing content. |
| | |

| Column Name | Description | Type | Length | Nullable | Default Value | Key Type |
|---|---|---|---|---|---|---|
| id | id of track | string | [24] | False | | PK |
| NutritionIdScheduleId | id of nutrition schedule | string | [24] | False | | FK |
| userId | id of user | string | [24] | False | | FK |
| totalCaloriesIntake | calories intake by user | float | 1 | False | | |
| totalProteinIntake | protein intake by user | float | 1 | False | | |
| total Carbohydrate Intake | carbohydrate intake by user | float | 1 | False | | |
| totalFatsIntake | fats intake by user | Float | 1 | False | | |
| totalDays | number of days user take diet | int | 1 | False | | |
| createAt | date of creation | Date | *1* | False | | |
| updateAt | date of updation | Date | *1* | False | | |

Data 8

<table>
<tr><th colspan="7">&lt; Data 8&gt;</th></tr>
<tr><td colspan="2"><b>Name</b></td><td colspan="5">ExerciseReminder</td></tr>
<tr><td colspan="2"><b>Alias</b></td><td colspan="5">Write other names used for the first entry.</td></tr>
<tr><td colspan="2"><b>Where-used/how-used</b></td><td colspan="5">during exercise tracking</td></tr>
<tr><td colspan="2"><b>Content description</b></td><td colspan="5">Notation for representing content.</td></tr>
<tr><td colspan="7"></td></tr>
<tr><td><b>Column Name</b></td><td><b>Description</b></td><td><b>Type</b></td><td><b>Length</b></td><td><b>Nullable</b></td><td><b>Default Value</b></td><td><b>Key Type</b></td></tr>
<tr><td>id</td><td>id of reminder</td><td>string</td><td>[24]</td><td>False</td><td></td><td>PK</td></tr>
<tr><td>NutritionIdScheduleId</td><td>id of nutrition schedule</td><td>string</td><td>[24]</td><td>False</td><td></td><td>FK</td></tr>
<tr><td>userId</td><td>id of user</td><td>string</td><td>[24]</td><td>False</td><td></td><td>FK</td></tr>
<tr><td>trackId</td><td>id of track</td><td>string</td><td>[24]</td><td>False</td><td></td><td>FK</td></tr>
</table>

| | | | | | | |
|---|---|---|---|---|---|---|
| upcoming days | days to remind | Date | [*] | True | NULL | |
| isReminder | To check that reminder will be off or on | bool | 1 | False | | |
| createAt | date of creation | Date | *1* | False | | |
| updateAt | date of updation | Date | *1* | False | | |

Data 9

| | | | | | | |
|---|---|---|---|---|---|---|
| **< Data 9>** | | | | | | |
| **Name** | DietReminder | | | | | |
| **Alias** | Write other names used for the first entry. | | | | | |
| **Where-used/how-used** | during diet tracking | | | | | |
| **Content description** | Notation for representing content. | | | | | |
| | | | | | | |
| **Column Name** | **Description** | **Type** | **Length** | **Null able** | **Default Value** | **Key Type** |
| id | id of reminder | string | [24] | False | | PK |
| NutritionIdScheduleId | id of nutrition | string | [24] | False | | FK |
| userId | last name of user | string | [24] | False | | FK |
| trackId | password of user for login | string | [24] | False | | *FK* |
| upcoming days | | Date | [*] | True | NULL | |
| isReminder | to check that reminder is set or not | bool | 1 | False | | |
| createAt | date of creation | Date | *1* | False | | |
| updateAt | date of updation | Date | *1* | False | | |

Data 10

| | |
|---|---|
| **< Data 10>** | |
| **Name** | Muscle |
| **Alias** | Write other names used for the first entry. |

| | |
|---|---|
| **Where-used/how-used** | during recommending |
| **Content description** | Notation for representing content. |
| | |

| **Column Name** | **Description** | **Type** | **Length** | **Nullable** | **Default Value** | **Key Type** |
|---|---|---|---|---|---|---|
| id | id of muscle | string | [24] | False | | PK |
| userId | id of user | string | [24] | False | | FK |
| detectedMuscles | muscles of user | string | [24] | False | | |
| userImages | images of user | string | [20] | False | | |
| createAt | date of creation | Date | *1* | False | | |
| updateAt | date of updation | Date | *1* | False | | |

## Work BreakDown Structure

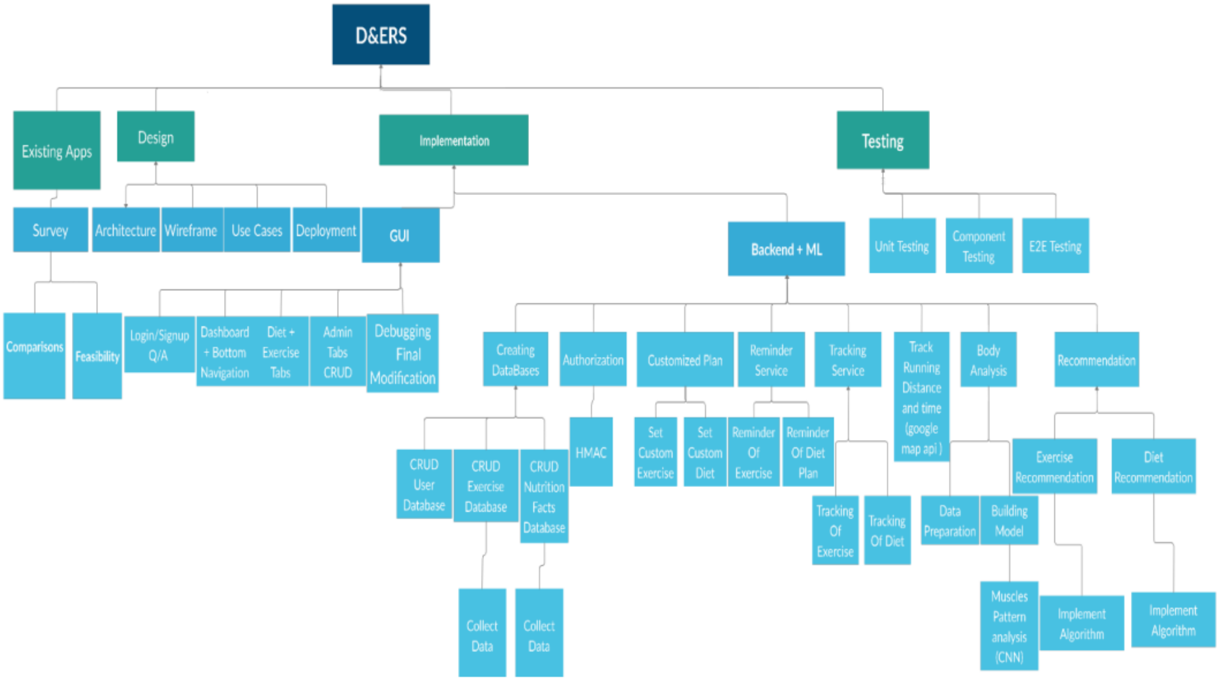
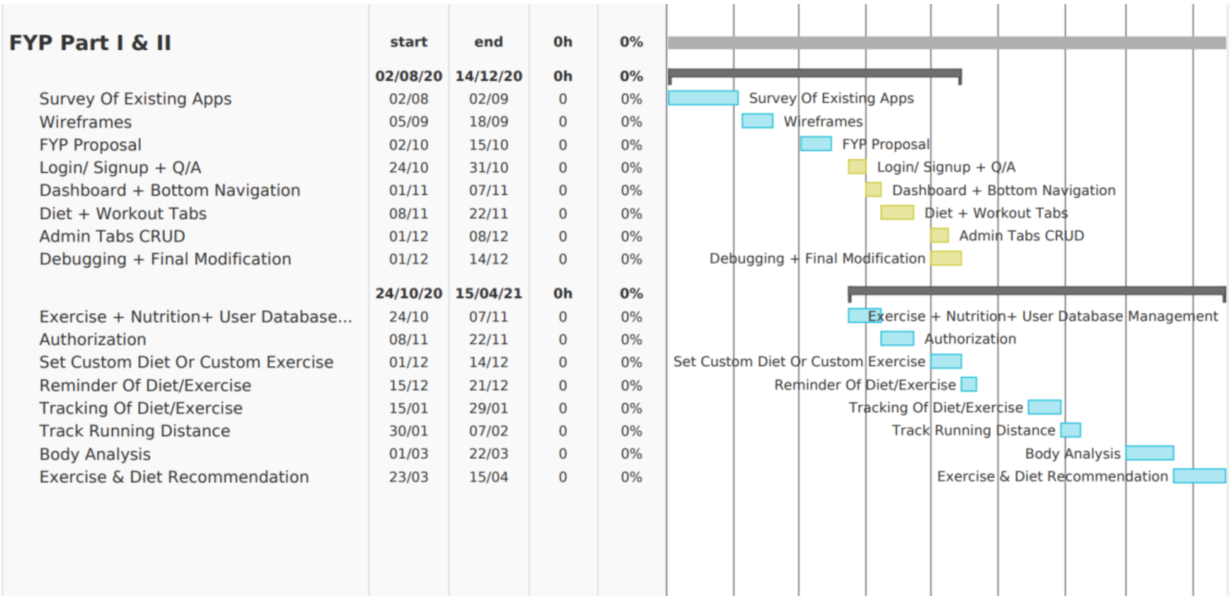


Figure 01. Work Breakdown Structure

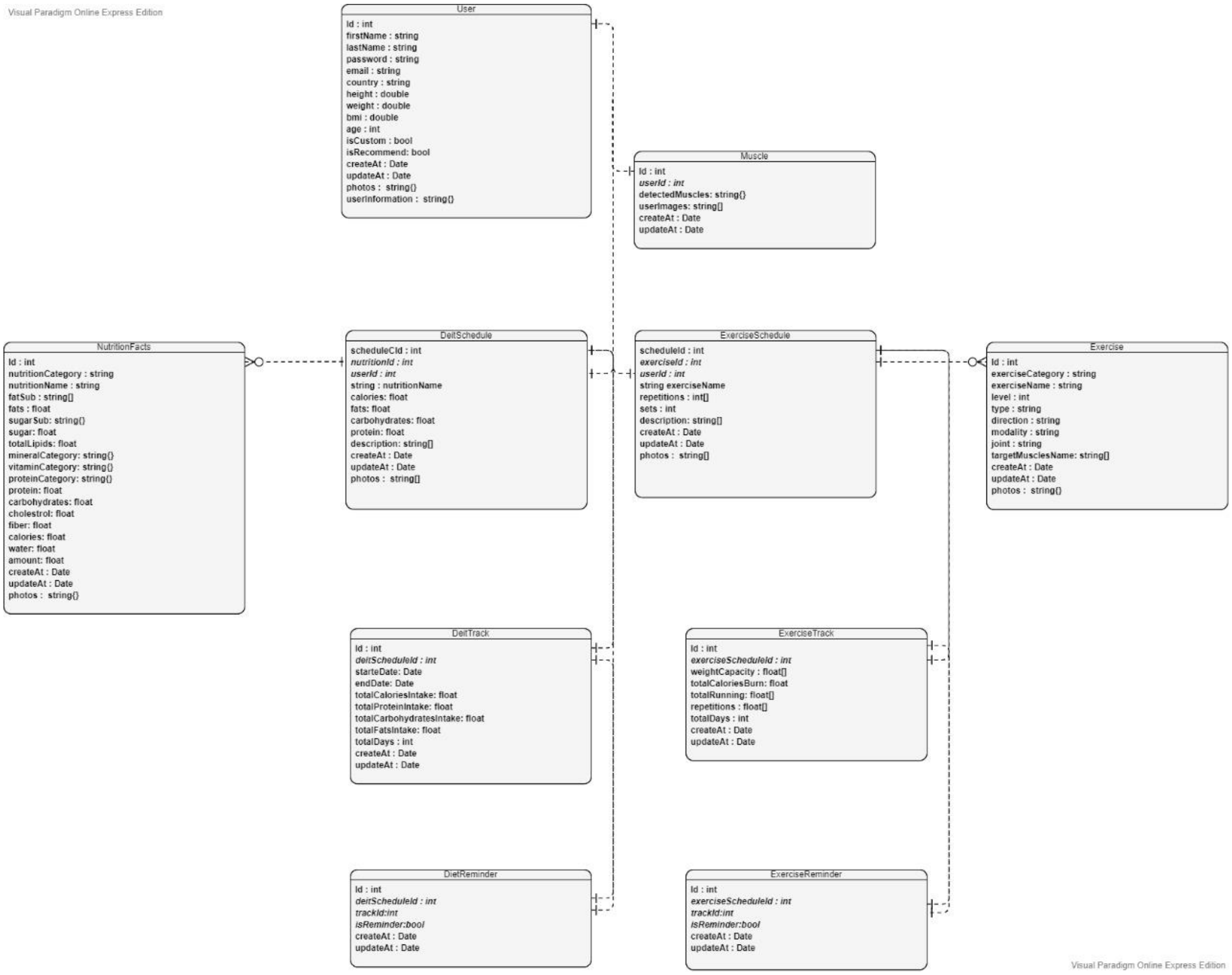
## Project Gantt Chart



| FYP Part I & II | start | end | 0h | 0% |
|---|---|---|---|---|
| | 02/08/20 | 14/12/20 | 0h | 0% |
| Survey Of Existing Apps | 02/08 | 02/09 | 0 | 0% |
| Wireframes | 05/09 | 18/09 | 0 | 0% |
| FYP Proposal | 02/10 | 15/10 | 0 | 0% |
| Login/ Signup + Q/A | 24/10 | 31/10 | 0 | 0% |
| Dashboard + Bottom Navigation | 01/11 | 07/11 | 0 | 0% |
| Diet + Workout Tabs | 08/11 | 22/11 | 0 | 0% |
| Admin Tabs CRUD | 01/12 | 08/12 | 0 | 0% |
| Debugging + Final Modification | 01/12 | 14/12 | 0 | 0% |
| | 24/10/20 | 15/04/21 | 0h | 0% |
| Exercise + Nutrition+ User Database... | 24/10 | 07/11 | 0 | 0% |
| Authorization | 08/11 | 22/11 | 0 | 0% |
| Set Custom Diet Or Custom Exercise | 01/12 | 14/12 | 0 | 0% |
| Reminder Of Diet/Exercise | 15/12 | 21/12 | 0 | 0% |
| Tracking Of Diet/Exercise | 15/01 | 29/01 | 0 | 0% |
| Track Running Distance | 30/01 | 07/02 | 0 | 0% |
| Body Analysis | 01/03 | 22/03 | 0 | 0% |
| Exercise & Diet Recommendation | 23/03 | 15/04 | 0 | 0% |

Survey Of Existing Apps
Wireframes
FYP Proposal
Login/ Signup + Q/A
Dashboard + Bottom Navigation
Diet + Workout Tabs
Admin Tabs CRUD
Debugging + Final Modification
Exercise + Nutrition+ User Database Management
Authorization
Set Custom Diet Or Custom Exercise
Reminder Of Diet/Exercise
Tracking Of Diet/Exercise
Track Running Distance
Body Analysis
Exercise & Diet Recommendation

## ER Diagram



Setup Instructions:

Pre-requisite include mongodb and node js installed.

1. Download the project file.
2. Extract it.
3. Open command prompt and go to the folder ‘DAERS-Project-main’.
4. Now run command ‘cd user’.
5. Run ‘npm install’ and then ‘npm start’.
6. Open a new command prompt in ‘DAERS-Project-main’ and run ‘cd exercise’.
7. Run ‘npm install’ and then ‘npm start’.
8. Open a new command prompt in ‘DAERS-Project-main’ and run ‘cd exerciseschedule’.
9. Run ‘npm install’ and then ‘npm start’.
10. Open a new command prompt in ‘DAERS-Project-main’ and run ‘cd nutritionfacts’.
11. Run ‘npm install’ and then ‘npm start’.
12. Open a new command prompt in ‘DAERS-Project-main’ and run ‘cd nutritionschedule’.
13. Run ‘npm install’ and then ‘npm start’.
14. Open a new command prompt in ‘DAERS-Project-main’ and run ‘cd client’.
15. Run ‘npm install’ and then ‘npm run dev’.