

Assignment 2



Assignment Concept

A Gym requires an app to periodically track assessments on individual members

Assessments capture essential measurements on the member at a given date

The app should generate analytics on the status of the member after a given assessment

The app should enable a Trainer to review assessments, and comment on any specific assessment

BMI Tracker

Fill in your assessments week by week and track your BMI. Trainers will comment on your progress and encourage you to keep going!



Register

Name

Gender

Email

Password

Address

Height

Starting Weight

Submit



Log-in

Email

homer@simpson.com

Password

•••••

Login





HOMER SIMPSON



IDEAL WEIGHT INDICATOR



42.91

BMI



VERY SEVERELY OBESE

Weight

00.00kg

Chest

00.00cm

Thigh

00.00cm

Upper Arm

00.00cm

Waist

00.00cm

Hips

00.00cm

[Add Assessment](#)

Date	Weight	Chest	Thigh	Upper Arm	Waist	Hips	Trend	Comment	
26-Mar-2017 11:34:49	124.0	45.0	12.0	23.0	33.0	38.0			
19-Mar-2017 08:31:00	123.3	45.0	12.0	23.3	31.0	36.0			
10-Mar-2017 08:31:00	123.3	45.0	12.5	23.3	32.0	38.0			



designed by freepik.com

Register

Name

Homer Simpson

Gender

Male

Email

homer@simpson.com

Password

•••••

Address

Springfield, Mass

Height

1.7

Starting Weight



101.0



Submit

Play Gym Trainer

Dashboard

Logout

 **Homer Simpson** 
3 assessments

 **Bart Simpson** 
0 assessments



HOMER SIMPSON



IDEAL WEIGHT INDICATOR



42.66

BMI



VERY SEVERELY OBESE

Weight	Chest	Thigh	Upper Arm	Waist	Hips	Comment
123.3	45.0	12.5	23.3	32.0	38.0	

Comments

good stuff!

Update

Weight	Chest	Thigh	Upper Arm	Waist	Hips	Comment
123.3	45.0	12.0	23.3	31.0	36.0	

Comments

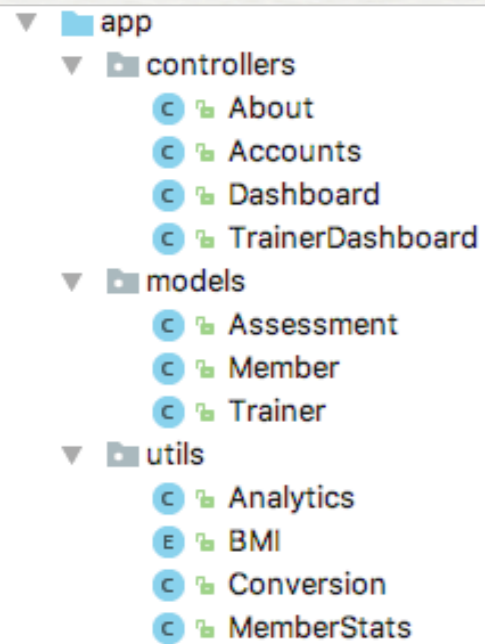
Update

Solution

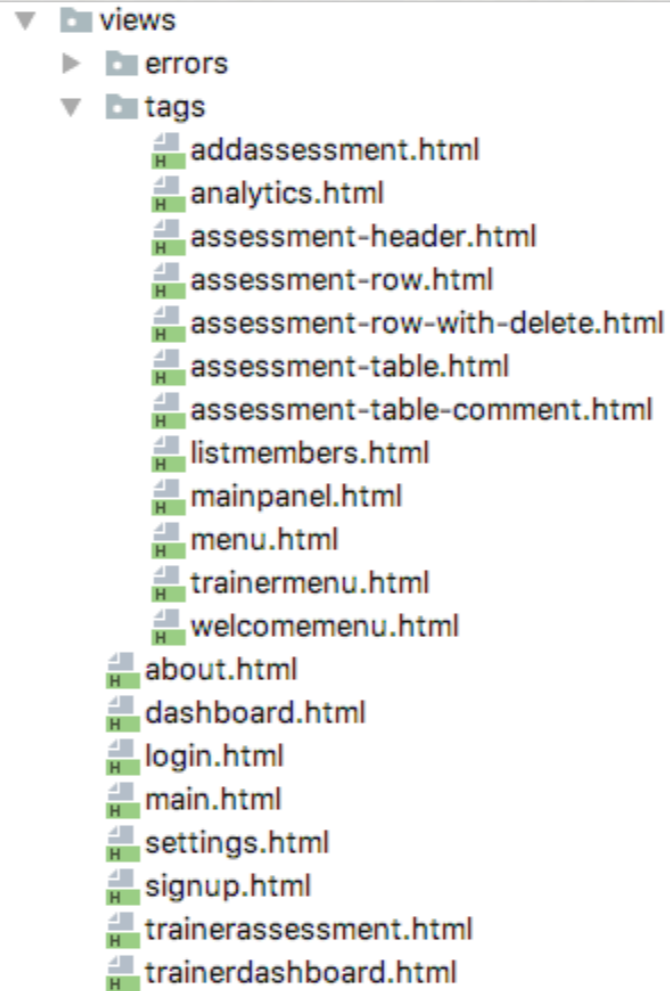
java

html

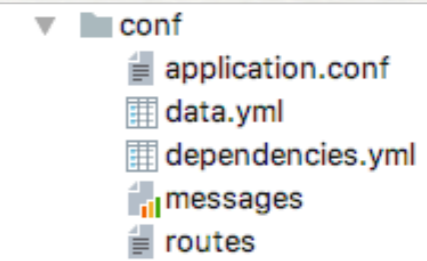
config



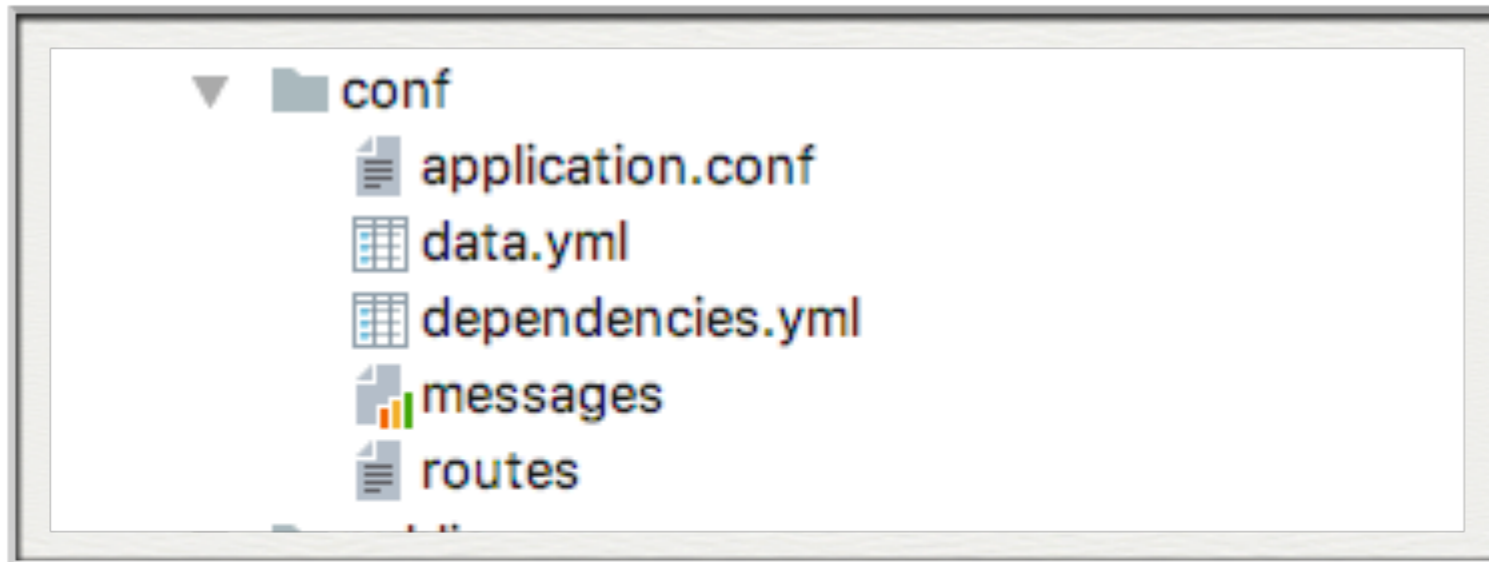
```
▼ app
  ▼ controllers
    About
    Accounts
    Dashboard
    TrainerDashboard
  ▼ models
    Assessment
    Member
    Trainer
  ▼ utils
    Analytics
    BMI
    Conversion
    MemberStats
```



```
▼ views
  errors
  tags
    addassessment.html
    analytics.html
    assessment-header.html
    assessment-row.html
    assessment-row-with-delete.html
    assessment-table.html
    assessment-table-comment.html
    listmembers.html
    mainpanel.html
    menu.html
    trainermenu.html
    welcomemenu.html
  about.html
  dashboard.html
  login.html
  main.html
  settings.html
  signup.html
  trainerassessment.html
  trainerdashboard.html
```



```
▼ conf
  application.conf
  data.yml
  dependencies.yml
  messages
  routes
```



config

Routes

```
# Routes
# This file defines all application routes (Higher priority routes first)

# Accounts
GET    /                Accounts.index
GET    /signup          Accounts.signup
GET    /login           Accounts.login
GET    /logout          Accounts.logout
POST   /authenticate    Accounts.authenticate
POST   /register        Accounts.register
GET    /settings        Accounts.settings
POST   /settings        Accounts.updateSettings

# Home page
GET    /dashboard       Dashboard.index
POST   /dashboard/addassessment Dashboard.addAssessment
GET    /dashboard/{memberid}/deleteassessment/{assessmentid} Dashboard.deleteAssessment
GET    /about           About.index

# Trainer page
GET    /trainerdashboard TrainerDashboard.index
GET    /trainerdashboard/deletemeber/{id} TrainerDashboard.deleteMember
GET    /trainerassessment/{id} TrainerDashboard.trainerAssessment
POST   /editcomment/{id} TrainerDashboard.editComment

# Ignore favicon requests
GET    /favicon.ico     404

# Map static resources from the /app/public folder to the /public path
GET    /public/         staticDir:public

# Catch all
*      /{controller}/{action} {controller}.{action}
```

```
Assessment(a1):  
  weight: 123.3  
  chest: 45.0  
  thigh: 12.5  
  upperarm: 23.3  
  waist: 32  
  hips: 38  
  trend: true  
  date: 2017-03-10 08:31:00
```

```
Assessment(a2):  
  weight: 123.3  
  chest: 45.0  
  thigh: 12  
  upperarm: 23.3  
  waist: 31  
  hips: 36  
  trend: true  
  date: 2017-03-19 08:31:00
```

```
Member(m1):  
  email: homer@simpson.com  
  password: secret  
  name: Homer Simpson  
  address: Springfield, Mass  
  gender: Male  
  height: 1.7  
  startingweight: 101  
  assessments:  
  - a1  
  - a2
```

```
Member(m2):  
  email: bart@simpson.com  
  password: secret  
  name: Bart Simpson  
  address: Springfield, Mass  
  gender: Male  
  height: 1.2  
  startingweight: 67
```

```
Trainer(t1):  
  email: marge@simpson.com  
  password: secret
```

```
@OnApplicationStart  
public class Bootstrap extends Job  
{  
  public void doJob()  
  {  
    if (Member.count() == 0)  
    {  
      Fixtures.loadModels("data.yaml");  
    }  
  }  
}
```

```
# Database configuration
# ~~~~~
# Enable a database engine if needed.
#
# To quickly set up a development database, use either:
#   - mem : for a transient in memory database (H2 in memory)
#   - fs  : for a simple file written database (H2 file stored)

db.default=mem
```

```
# Database configuration
# ~~~~~
# Enable a database engine if needed.
#
# To quickly set up a development database, use either:
#   - mem : for a transient in memory database (H2 in memory)
#   - fs  : for a simple file written database (H2 file stored)

# db.default=mem

db=${DATABASE_URL}
jpa.dialect=org.hibernate.dialect.PostgreSQLDialect
jpa.ddl=update
```

▼ play-gym-web ~/repos/modules/web/ict-2017/pr

▼ app

▼ controllers

Ⓢ About

Ⓢ Accounts

Ⓢ Dashboard

Ⓢ TrainerDashboard

▼ models

Ⓢ Assessment

Ⓢ Member

Ⓢ Trainer

▼ utils

Ⓢ Analytics

Ⓢ BMI

Ⓢ Conversion

Ⓢ MemberStats

Utils

BMI

```
public class MemberStats
{
    public double bmi;
    public String bmiCategory;
    public boolean isIdealBodyweight;
    public boolean trend;
}
```

MemberStats

```
public enum BMI
{
    VERY_SEVERELY_UNDERWEIGHT (0.0, 15.0) {
        @Override
        public String toString(){ return "Very Severely Underweight";}
    },
    SEVERELY_UNDERWEIGHT      (15.0, 16.0) {
        @Override
        public String toString(){ return "Severely Underweight";}
    },
    UNDERWEIGHT              (16, 18.5) {
        @Override
        public String toString(){ return "Underweight";}
    },
    NORMAL                    (18.5, 25) {
        @Override
        public String toString(){ return "Normal";}
    },
    OVERWEIGHT                (25, 30) {
        @Override
        public String toString(){ return "Overweight";}
    },
    MODERATELY_OBESE         (30, 35) {
        @Override
        public String toString(){ return "Moderately Obese";}
    },
    SEVERELY_OBESE           (35, 40) {
        @Override
        public String toString(){ return "Severely Obese";}
    },
    VERY_SEVERELY_OBESE      (40, 1000) {
        @Override
        public String toString(){ return "Very Severely Obese";}
    };

    private double rangeLow;
    private double rangeHigh;

    private BMI(double rangeLow, double rangeHigh) {
        this.rangeLow = rangeLow;
        this.rangeHigh = rangeHigh;
    }

    public boolean bmiCategory(double bmiValue){
        if ((bmiValue >= this.rangeLow) && (bmiValue < this.rangeHigh)){
            return true;
        }
        return false;
    }
}
```


Conversion

```
public class Conversion
{
    public static double round(double numberToConvert, double precision)
    {
        double p = Math.pow(10, precision);
        return (double) Math.round(numberToConvert * p) / p;
    }

    public static double convertKGtoPounds(double numberToConvert, double precision)
    {
        return round(numberToConvert * 2.2, precision);
    }

    public static double convertMetresToInches(double numberToConvert, double precision)
    {
        return round(numberToConvert * 39.37, precision);
    }
}
```

Analytics (1/2)

```
public class Analytics
{
    public static MemberStats generateMemberStats(Member member)
    {
        MemberStats stats = new MemberStats();

        double weight = member.startingweight;
        List<Assessment> assessments = member.assessments;
        if (assessments.size() > 0) {
            Assessment assessment = assessments.get(assessments.size() - 1);
            weight = assessment.weight;
        }
        stats.bmi = calculateBMI(member, weight);
        stats.bmiCategory = determineBMICategory(stats.bmi);
        stats.isIdealBodyweight = isIdealBodyWeight(member, weight);
        stats.trend = true;
        if (assessments.size() > 1) {
            stats.trend = assessments.get(assessments.size() - 2).weight > assessments.get(assessments.size() - 1).weight;
        }
        return stats;
    }

    public static double calculateBMI(Member member, double weight)
    {
        if (member.height <= 0)
            return 0;
        else
            return Conversion.round((weight / (member.height * member.height)), 2);
    }

    public static String determineBMICategory(double bmiValue)
    {
        for (BMI bmi : BMI.values()) {
            if (bmi.bmiCategory(bmiValue)) {
                return bmi.toString();
            }
        }
        return "No category available.";
    }

    //...
}
```

```
public class Analytics
{
    //...

    public static boolean isIdealBodyWeight(Member member, double weight)
    {
        double fiveFeet = 60.0;
        double idealBodyWeight;

        double inches = Conversion.convertMetresToInches(member.height, 2);

        if (inches <= fiveFeet) {
            if (member.gender.equals("M")) {
                idealBodyWeight = 50;
            } else {
                idealBodyWeight = 45.5;
            }
        } else {
            if (member.gender.equals("M")) {
                idealBodyWeight = 50 + ((inches - fiveFeet) * 2.3);
            } else {
                idealBodyWeight = 45.5 + ((inches - fiveFeet) * 2.3);
            }
        }

        Logger.info("Ideal Weigfht" + idealBodyWeight);
        return ((idealBodyWeight <= (weight + 2.0))
            && (idealBodyWeight >= (weight - 2.0))
        );
    }
}
```

```
▼ play-gym-web ~/repos/modules/web/ict-2017/pr
  ▼ app
    ▼ controllers
      C About
      C Accounts
      C Dashboard
      C TrainerDashboard
    ▼ models
      C Assessment
      C Member
      C Trainer
    ▼ utils
      C Analytics
      E BMI
      C Conversion
      C MemberStats
```

Models

Member

```
@Entity
public class Member extends Model
{
    public String email;
    public String name;
    public String password;
    public String address;
    public String gender;
    public double height;
    public double startingweight;

    @OneToMany(cascade = CascadeType.ALL)
    public List<Assessment> assessments = new ArrayList<Assessment>();

    public Member(String email, String name, String password, String address,
                  String gender, double height, double startingweight)
    {
        this.email = email;
        this.name = name;
        this.password = password;
        this.address = address;
        this.gender = gender;
        this.height = height;
        this.startingweight = startingweight;
    }

    public static Member findByEmail(String email)
    {
        return find("email", email).first();
    }

    public boolean checkPassword(String password)
    {
        return this.password.equals(password);
    }
}
```

```
@Entity
public class Trainer extends Model
{
    public String email;
    public String password;

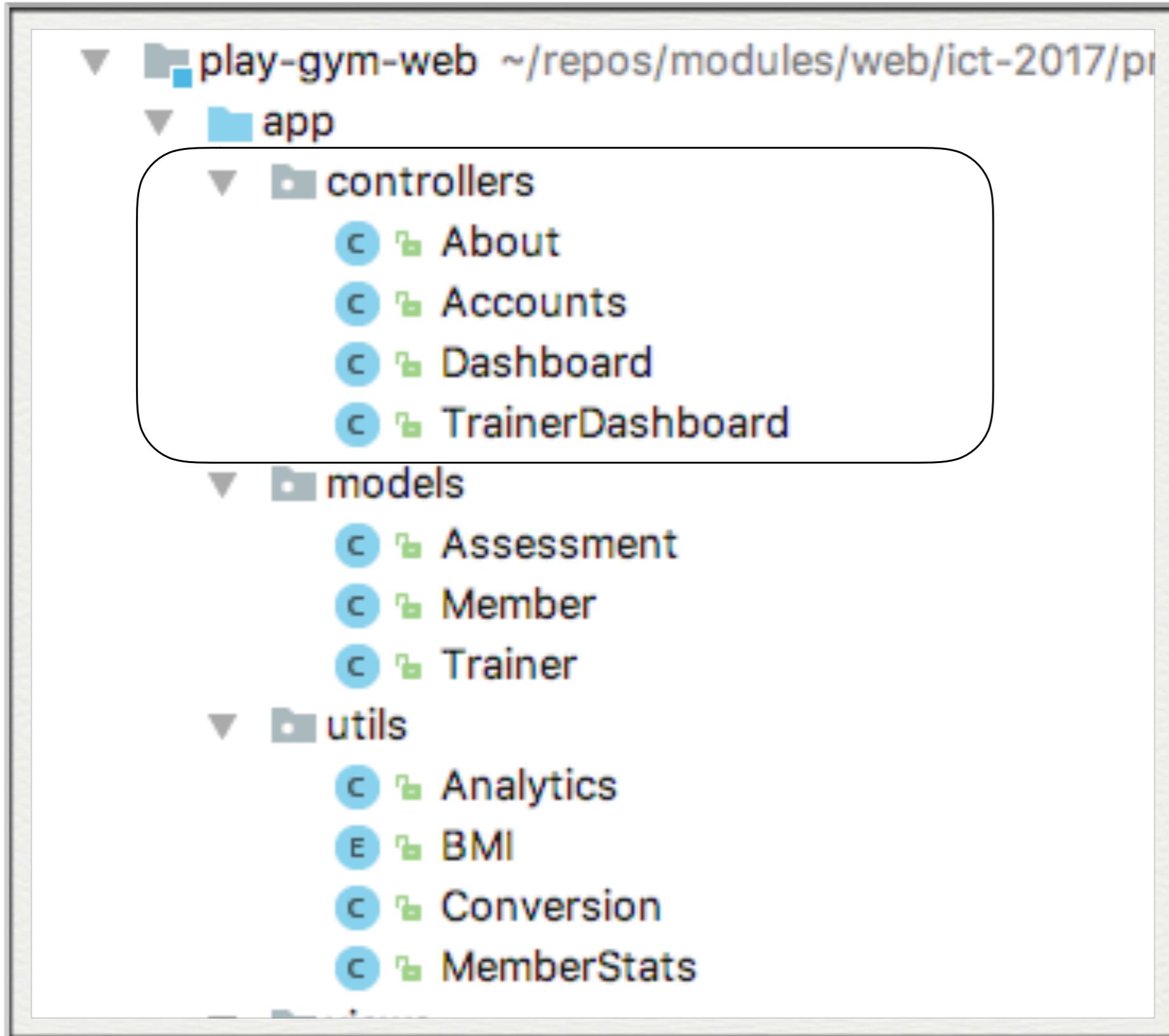
    public Trainer(String email, String password)
    {
        this.email = email;
        this.password = password;
    }

    public static Trainer findByEmail(String email)
    {
        return find("email", email).first();
    }
    public boolean checkPassword(String password)
    {
        return this.password.equals(password);
    }
}
```

```
@Entity
public class Assessment extends Model
{
    public double weight;
    public double chest;
    public double thigh;
    public double upperarm;
    public double waist;
    public double hips;
    public boolean trend;

    @Lob
    public String comment;
    public Date date;

    public Assessment(double weight, double chest, double thigh,
                      double upperarm, double waist, double hips)
    {
        this.weight = weight;
        this.chest = chest;
        this.thigh = thigh;
        this.upperarm = upperarm;
        this.waist = waist;
        this.hips = hips;
        this.date = new Date();
    }
}
```



Controllers

About

```
public class About extends Controller
{
    public static void index() {
        Logger.info("Rendering about");
        render ("about.html");
    }
}
```

```
public class Accounts extends Controller
{
    public static void index()
    {
        render("about.html");
    }

    public static void signup()
    {
        render("signup.html");
    }

    public static void login()
    {
        render("login.html");
    }

    public static void settings()
    {
        Member member = getLoggedInMember();
        render("settings.html", member);
    }

    public static void updateSettings(Member member)
    {
        Member loggedInMember = getLoggedInMember();

        loggedInMember.email = member.email;
        loggedInMember.name = member.name;
        loggedInMember.password = member.password;
        loggedInMember.address = member.address;
        loggedInMember.gender = member.gender;
        loggedInMember.height = member.height;
        loggedInMember.startingweight = member.startingweight;

        loggedInMember.save();
        redirect("/settings");
    }
}
```

Accounts (1/2)

Accounts (2/2)

```
public class Accounts extends Controller
{
    public static void logout()
    {
        session.clear();
        index();
    }

    public static Member getLoggedInMember()
    {
        Member member = null;
        if (session.contains("logged_in_Memberid")) {
            String memberId = session.get("logged_in_Memberid");
            member = Member.findById(Long.parseLong(memberId));
        } else {
            login();
        }
        return member;
    }

    public static void register(String email, String name, String password, String address,
                                String gender, double height, double startingweight)
    {
        Logger.info(name + " " + email);
        Member member = new Member(email, name, password, address, gender, height, startingweight);
        member.save();
        index();
    }

    public static void authenticate(String email, String password)
    {
        Logger.info("Attempting to authenticate with " + email + ":" + password);

        Member member = Member.findByEmail(email);
        if ((member != null) && (member.checkPassword(password) == true)) {
            Logger.info("Authentication successful");
            session.put("logged_in_Memberid", member.id);
            Dashboard.index();
        } else {
            Trainer trainer = Trainer.findByEmail(email);
            if ((trainer != null) && (trainer.checkPassword(password) == true)) {
                Logger.info("Authentication successful");
                session.put("logged_in_Trainerid", trainer.id);
                TrainerDashboard.index();
            } else {
                Logger.info("Authentication failed");
                login();
            }
        }
    }
}
```

Dashboard

```
public class Dashboard extends Controller
{
    public static void index()
    {
        Logger.info("Rendering Dashboard");
        Member member = Accounts.getLoggedInMember();
        List<Assessment> assessments = member.assessments;
        MemberStats memberStats = Analytics.generateMemberStats(member);
        Collections.reverse(assessments);
        render("dashboard.html", member, assessments, memberStats);
    }

    public static void addAssessment(double weight, double chest, double thigh,
                                     double upperarm, double waist, double hips)
    {
        Logger.info("Creating Assessment");
        Member member = Accounts.getLoggedInMember();
        Assessment assessment = new Assessment(weight, chest, thigh, upperarm, waist, hips);
        MemberStats memberStats = Analytics.generateMemberStats(member);
        assessment.trend = memberStats.trend;
        member.assessments.add(assessment);
        member.save();
        redirect("/dashboard");
    }

    public static void deleteAssessment(Long memberid, Long assessmentid)
    {
        Member member = Member.findById(memberid);
        Assessment assessment = Assessment.findById(assessmentid);
        member.assessments.remove(assessment);
        member.save();
        assessment.delete();
        redirect("/dashboard");
    }
}
```

TrainerDashboard

```
public class TrainerDashboard extends Controller
{
    public static void index()
    {
        List<Member> members = Member.findAll();
        Logger.info("Rendering Trainer Dashboard");
        render("trainerdashboard.html", members);
    }

    public static void trainerAssessment(Long id)
    {
        Member member = Member.findById(id);
        List<Assessment> assessments = member.assessments;
        MemberStats memberStats = Analytics.generateMemberStats(member);
        Collections.reverse(assessments);
        render("trainerassessment.html", member, assessments, memberStats);
    }

    public static void editComment(Long id, String comment)
    {
        Logger.info("Comment " + comment);
        Assessment assessment = Assessment.findById(id);
        assessment.comment = comment;
        assessment.save();
        redirect("/trainerdashboard");
    }

    public static void deleteMember(Long id)
    {
        Member member = Member.findById(id);
        if (member != null) {
            member.delete();
        }
        redirect("/trainerdashboard");
    }
}
```

- ▼ views
 - ▶ errors
 - ▼ tags
 - addassessment.html
 - analytics.html
 - assessment-header.html
 - assessment-row.html
 - assessment-row-with-delete.html
 - assessment-table.html
 - assessment-table-comment.html
 - listmembers.html
 - mainpanel.html
 - menu.html
 - trainermenu.html
 - welcomemenu.html
 - about.html
 - dashboard.html
 - login.html
 - main.html
 - settings.html
 - signup.html
 - trainerassessment.html
 - trainerdashboard.html

Views

main

```
<!DOCTYPE html>
<html>
  <head>
    <title>#{get 'title' /}</title>
    <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>
    <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/semantic-ui/2.2.6/semantic.min.js"></script>
    <link type="text/css" href="https://cdnjs.cloudflare.com/ajax/libs/semantic-ui/2.2.6/semantic.min.css" rel="stylesheet" >
  </head>
  <body>
    <section class="ui container">
      #{doLayout /}
    </section>
  </body>
  <script>
    $(' .ui.dropdown' ).dropdown();
  </script>
</html>
```

welcomemenu

```
<nav class="ui menu">
  <header class="ui header item"> <a href="/"> Play Gym </a></header>
  <div class="right menu">
    <a id="signup" class="item" href="/signup"> Signup </a>
    <a id="login" class="item" href="/login"> Login </a>
  </div>
</nav>

<script>
  $("##{_id}").addClass("active item");
</script>
```

BMI Tracker

Fill in your assessments week by week and track your BMI. Trainers will



about

```
#{extends 'main.html' /}
#{set title:'About' /}

#{welcomemenu id:"about"/}

<section class="ui center aligned middle aligned two column grid segment">
  <div class="column">
    <header class="ui header"> BMI Tracker </header>
    <p> Fill in your assessments week by week and track your BMI.
      Trainers will comment on your progress and encourage your to keep going!</p>
  </div>
  <div class="column">
    
  </div>
</section>
```


Register

Name

Name

Gender

Gender

Email

homer@simpson.com

Password

•••••

Address

Height

Name

Starting Weight

Starting Weight

Submit



```

#{extends 'main.html' /}
#{set title:'Signup' /}
#{welcomemenu id:"signup"/}

<div class="ui two column grid basic middle aligned segment">
  <div class="column">
    <form class="ui stacked segment form" action="/register" method="POST">
      <h3 class="ui header">Register</h3>
      <div class="two fields">
        <div class="field">
          <label>Name</label>
          <input placeholder="Name" type="text" name="name">
        </div>
        <div class="field">
          <label>Gender</label>
          <input placeholder="Gender" type="text" name="gender">
        </div>
      </div>
      <div class="field">
        <label>Email</label>
        <input placeholder="Email" type="text" name="email">
      </div>
      <div class="field">
        <label>Password</label>
        <input type="password" name="password">
      </div>
      <div class="field">
        <label>Address</label>
        <input type="text" name="address">
      </div>
      <div class="two fields">
        <div class="field">
          <label>Height</label>
          <input placeholder="Name" type="text" name="height">
        </div>
        <div class="field">
          <label>Starting Weight</label>
          <input placeholder="Starting Weight" type="text" name="startingweight">
        </div>
      </div>
      <button class="ui blue submit button">Submit</button>
    </form>
  </div>
  <div class="column">
    
  </div>
</div>

```

Log-in

Email

Password



```

#{extends 'main.html' /}
#{set title:'login' /}
#{welcomemenu id:"login"/}

<div class="ui two column middle aligned grid basic segment">
  <div class="column">
    <form class="ui stacked segment form" action="/authenticate" method="POST">
      <h3 class="ui header">Log-in</h3>
      <div class="field">
        <label>Email</label> <input placeholder="Email" name="email">
      </div>
      <div class="field">
        <label>Password</label> <input type="password" name="password">
      </div>
      <button class="ui blue submit button">Login</button>
    </form>
  </div>
  <div class="column">
    
  </div>
</div>

```

```

<nav class="ui menu">
  <header class="ui header item"> Play Gym </header>
  <div class="right menu">
    <a id="dashboard" class="item" href="/dashboard"> Dashboard </a>
    <a id="about" class="item" href="/about"> About </a>
    <div class="right menu">
      <div class="ui dropdown item">
        <i class="user icon"></i>
        <div class="menu">
          <a class="item" href="/settings">Settings</a>
          <a class="item" href="/logout">Logout</a>
        </div>
      </div>
    </div>
  </div>
</nav>

<script>
  $("##{_id}").addClass("active item");
</script>

```

menu

The dashboard for 'HOMER SIMPSON' displays the following information:

- User Profile:** HOMER SIMPSON
- Health Metrics:** IDEAL WEIGHT INDICATOR, BMI 42.91, VERY SEVERELY OBESE
- Assessment Input Fields:**
 - Weight: 00.00kg
 - Chest: 00.00cm
 - Thigh: 00.00cm
 - Upper Arm: 00.00cm
 - Waist: 00.00cm
 - Hips: 00.00cm
- Assessment History Table:**

Date	Weight	Chest	Thigh	Upper Arm	Waist	Hips	Trend	Comment	
26-Mar-2017 11:34:49	124.0	45.0	12.0	23.0	33.0	38.0	↔ (Red)		🗑️
19-Mar-2017 08:31:00	123.3	45.0	12.0	23.3	31.0	36.0	↔ (Green)		🗑️
10-Mar-2017 08:31:00	123.3	45.0	12.5	23.3	32.0	38.0	↔ (Green)		🗑️

dashboard

```

#{extends 'main.html' /}
#{set title:'Dashboard' /}
#{menu id:"dashboard"/}

<section class="ui segment">
  #{analytics member:member, memberStats:memberStats/}
  #{addassessment/}
  #{assessment-table assessments:assessments, member:member/}
</section>

```

dashboard

```
#{extends 'main.html' /}  
#{set title:'Dashboard' /}  
#{menu id:"dashboard"/}  
  
<section class="ui segment">  
  #{analytics member:member, memberStats:memberStats/  
  #{addassessment/  
  #{assessment-table assessments:assessments, member:member/  
</section>
```

addassessment

```
<form class="ui segment form" action="/dashboard/addassessment" method="POST">  
  <div class="three fields">  
    <div class="field">  
      <label>Weight</label>  
      <input placeholder="00.00kg" type="text" name="weight">  
    </div>  
    <div class="field">  
      <label>Chest</label>  
      <input placeholder="00.00cm" type="text" name="chest">  
    </div>  
    <div class="field">  
      <label>Thigh</label>  
      <input placeholder="00.00cm" type="text" name="thigh">  
    </div>  
  </div>  
  <div class="three fields">  
    <div class="field">  
      <label>Upper Arm</label>  
      <input placeholder="00.00cm" type="text" name="upperarm">  
    </div>  
    <div class="field">  
      <label>Waist</label>  
      <input placeholder="00.00cm" type="text" name="waist">  
    </div>  
    <div class="field">  
      <label>Hips</label>  
      <input placeholder="00.00cm" type="text" name="hips">  
    </div>  
  </div>  
  <button class="ui blue submit button"> Add Assessment </button>  
</form>
```

analytics

```
<section class="ui segment">  
  <div class="ui four small statistics">  
    <div class="ui statistic">  
      <div class="value">  
        <i class="user icon"></i>  
      </div>  
      <div class="label">  
        ${_member.name}  
      </div>  
    </div>  
    <div class="ui statistic">  
      <div class="value">  
        #{if _memberStats.isIdealBodyweight}  
          <i class="green dashboard icon"></i>  
        #{/if}  
        #{if !_memberStats.isIdealBodyweight}  
          <i class="red dashboard icon"></i>  
        #{/if}  
      </div>  
      <div class="label">  
        Ideal Weight Indicator  
      </div>  
    </div>  
    <div class="ui statistic">  
      <div class="value">  
        <i class="heartbeat icon"></i> ${_memberStats.bmi}  
      </div>  
      <div class="label">  
        BMI  
      </div>  
    </div>  
    <div class="ui statistic">  
      <div class="value">  
        <i class="doctor icon"></i>  
      </div>  
      <div class="label">  
        ${_memberStats.bmiCategory}  
      </div>  
    </div>  
  </div>  
</section>
```

assessment-table



```
<table class="ui celled table">
  <thead>
    #{assessment-header /}
  </thead>
  <tbody>
    #{list items:_assessments, as:'assessment'}
      #{assessment-row-with-delete assessment:assessment, member:_member/}
    #{/list}
  </tbody>
</table>
```



assessment-header

```
<tr>
  <th> Date </th>
  <th> Weight </th>
  <th> Chest </th>
  <th> Thigh </th>
  <th> Upper Arm </th>
  <th> Waist </th>
  <th> Hips </th>
  <th> Trend </th>
  <th> Comment </th>
</tr>
```

assessment-row-with-delete

```
<tr>
  <td> ${_assessment.date.toLocaleString()} </td>
  <td> ${_assessment.weight} </td><td> ${_assessment.chest} </td><td> ${_assessment.thigh} </td>
  <td> ${_assessment.upperarm} </td><td> ${_assessment.waist} </td><td> ${_assessment.hips} </td>
  <td>
    #{if _assessment.trend}
      <a class="ui teal tag label"></a>
    #{/if}
    #{if !_assessment.trend}
      <a class="ui red tag label"></a>
    #{/if}
  </td>
  <td> ${_assessment.comment} </td>
  <td>
    <a href="/dashboard/${_member.id}/deleteassessment/${_assessment.id}" class="ui icon button">
      <i class="icon trash"></i>
    </a>
  </td>
</tr>
```

 **Homer Simpson** 
3 assessments

 **Bart Simpson** 
0 assessments

trainerdashboard

trainermenu

```
#{extends 'main.html' /}
#{set title:'Trainer Dashboard' /}
#{trainermenu id:"dashboard"/}

<section class="ui segment">
  <div class="column">
    #{listmembers members:members/}
  </div>
</section>
```

```
<nav class="ui menu">
  <header class="ui header item"> Play Gym Trainer </header>
  <div class="right menu">
    <a id="dashboard" class="item" href="/trainerdashboard"> Dashboard
  </a>
    <a id="logout" class="item" href="/logout"> Logout </a>
  </div>
</nav>

<script>
  $("##{_id}").addClass("active item");
</script>
```

listmembers


```
<div class="ui relaxed divided list">
  #{list items:_members, as:'member'}
  <div class="item">
    <i class="large github middle aligned icon"></i>
    <div class="content">
      <div class="header">
        <a href="/trainerassessment/${member.id}"> ${member.name} </a>
        <a href="/trainerdashboard/deletemeber/${member.id}">
          <i class="ui large red icon trash"></i>
        </a>
      </div>
      <div class="description"> ${member.assessments.size()} assessments </div>
    </div>
  </div>
  #{/list}
</div>
```



HOMER SIMPSON



IDEAL WEIGHT INDICATOR



42.66
BMI



VERY SEVERELY OBESE

Weight	Chest	Thigh	Upper Arm	Waist	Hips	Comment
123.3	45.0	12.5	23.3	32.0	38.0	

trainerassessment

Comments

Update

Weight	Chest	Thigh	Upper Arm
123.3	45.0	12.0	23.3

```

#{extends 'main.html' /}
#{set title:'Dashboard' /}
#{trainermenu id:"dashboard"/}

#{analytics member:member, memberStats:memberStats/}
<section class="ui segment">
  #{assessment-table-comment assessments:assessments, member:member/}
</section>

```

Comments

Update

assessment-table-comment

```

#{list items:_assessments, as:'assessment'}
  <table class="ui celled table">
    <thead>
      #{assessment-header /}
    </thead>
    <tbody>
      #{assessment-row assessment:assessment, member:_member/}
    </tbody>
  </table>
  <h4 class="ui header">
    Comments
  </h4>
  <form class="ui form" action="/editcomment/${assessment.id}" method="POST">
    <div class="ui field">
      <textarea rows="2" name="comment"> ${assessment.comment} </textarea>
    </div>
    <button class="ui blue submit button"> Update </button>
  </form>
#{/list}

```

settings

```

#{extends 'main.html' /}
#{set title:'Settings' /}
#{menu id:"settings"/}

<div class="ui two column grid basic middle aligned" style="border: 1px solid black; padding: 10px;">
  <div class="column" style="width: 50%; padding-right: 10px;">
    
    <div class="column" style="width: 50%; padding-left: 10px;">
      <form class="ui stacked segment form" action="/se" style="border: 1px solid #ccc; padding: 10px;">
        <h3 class="ui header">Register</h3>
        <div class="two fields">
          <div class="field">
            <label>Name</label>
            <input placeholder="Name" type="text" name="member.name" value="" />
          </div>
          <div class="field">
            <label>Gender</label>
            <input placeholder="Gender" type="text" name="member.gender" value="" />
          </div>
        </div>
        <div class="field">
          <label>Email</label>
          <input placeholder="Email" type="text" name="member.email" value="{member.email}" />
        </div>
        <div class="field">
          <label>Password</label>
          <input type="password" name="member.password" value="{member.password}" />
        </div>
        <div class="field">
          <label>Address</label>
          <input type="text" name="member.address" value="{member.address}" />
        </div>
        <div class="two fields">
          <div class="field">
            <label>Height</label>
            <input placeholder="Name" type="text" name="member.height" value="{member.height}" />
          </div>
          <div class="field">
            <label>Starting Weight</label>
            <input placeholder="Starting Weight" type="text" name="member.startingweight" value="{member.startingweight}" />
          </div>
        </div>
        <button class="ui blue submit button">Submit</button>
      </form>
    </div>
  </div>
</div>

```



Register

Name

Gender

Email

Password

Address

Height

Starting Weight