Spotr Local Fitness Finder

A FULL-STACK WEBSITE BY TEAM SINISTER TRANQUILITY

- Nick Gonzalez
- Max Goduco
- Kate Remick
- Matt Elmore

presented 27 May 2022 for Skill Distillery class SD33

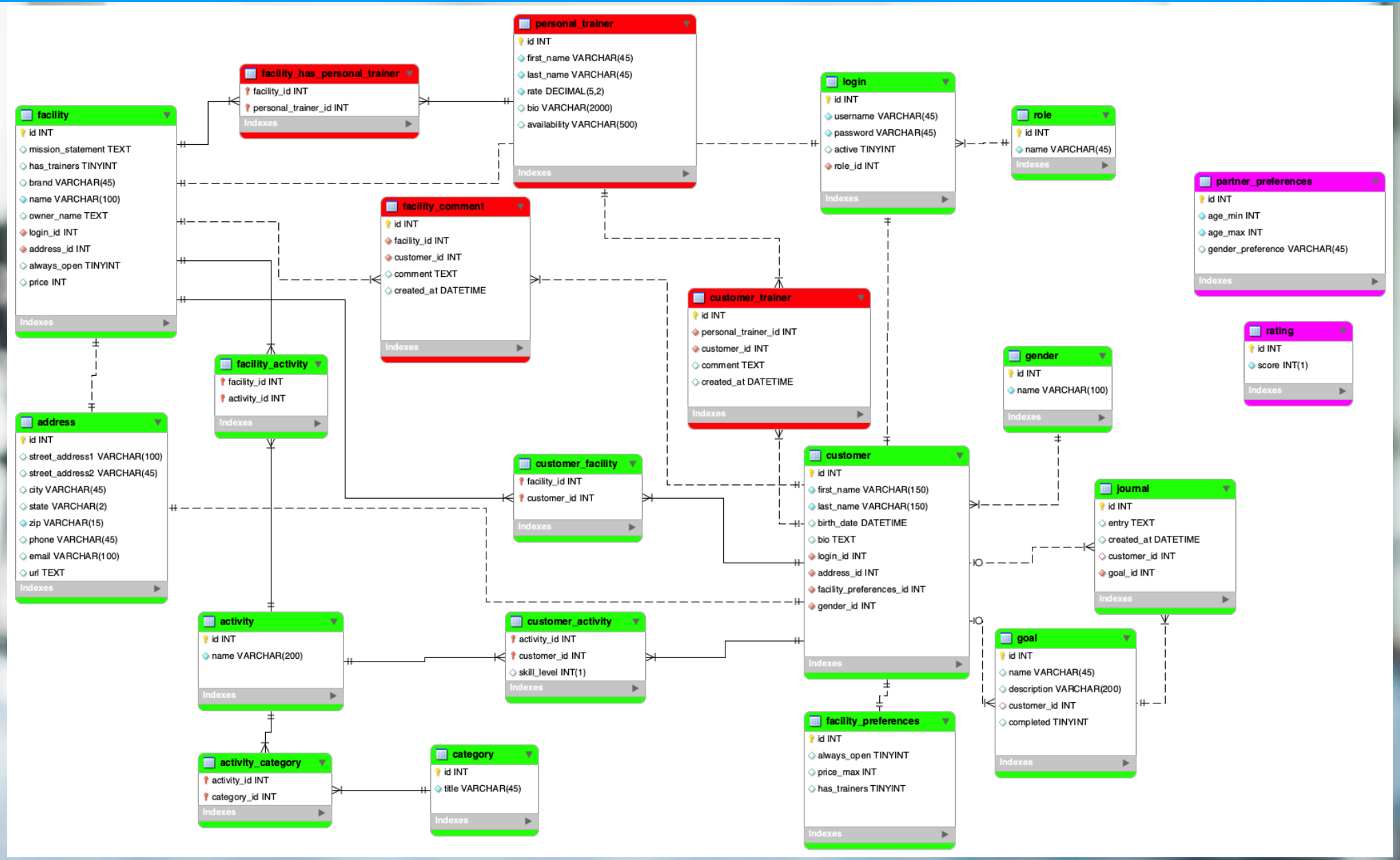
The Team

Outline

- Site Demonstration
- Database Overview
- DAOs – Database Access Objects
- Controllers
- Front-End
- Lessons Learned
- Trello
- Technologies Used
- Conclusion & Questions

Demonstration

Database Diagram



Controllers – JSP ,Controller, & Command Objects

```
@RequestMapping(path = "createCustomer.do", method = RequestMethod.POST)
public ModelAndView createCustomer(Customer customer, String dob, Address address, Gender gender,
        FacilityPreferences prefs, HttpSession session,@RequestParam("skillLevel") Integer[] skillLevels ,
        @RequestParam("activities") String... activityId) {
    ModelAndView mav = new ModelAndView();
    DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");
    customer.setBirthDate(LocalDate.parse(dob, formatter));
    System.out.println(customer.getBirthDate());
    customer.setAddress(address);
    customer.setGender(gender);
    customer.setFacilityPreferences(prefs);
    Login login = (Login) session.getAttribute("loggedInUser"); // HELPED FROM JEREMY
    customer = customerDao.createCustomer(login, customer);
    Integer[] usableSkillLevels = new Integer[activityId.length];
    int skillCount = 0;
    for (Integer skill : skillLevels) {
        System.out.println(skill);
        if( skill != null) {
            usableSkillLevels[skillCount] = skill;
            skillCount ++;
        }
    }
    List<CustomerActivity> customerActivities = new ArrayList<CustomerActivity>();
    if (activityId != null && activityId.length > 0) {
        for (int i = 0; i < activityId.length; i++) {
            CustomerActivity ca = new CustomerActivity();
            ca.setActivity(customerDao.findActivityById(Integer.parseInt(activityId[i])));
            ca.setSkillLevel(usableSkillLevels[i]);

            ca.setCustomer(customer);
            customerActivities.add(ca);
        }
    }
    customer = customerDao.editActivities(customer.getId(), customerActivities);
    customer = customerDao.findCustomerById(customer.getId());
    session.setAttribute("customer", customer);
    mav.setViewName("redirect:viewCustomer.do");
    return mav;
}
```

DAOs – Separate Customer & Customer Activity

```
@Override
public Customer createCustomer(Login user, Customer customer) {
    System.out.println("**********creating customer");
    user = em.find(Login.class, user.getId());
    customer.setLogin(user);
    em.persist(customer);
    em.flush();
    return customer;
}
```

DAOs – Persisting Customer & Other OBJs to DB

```
@Override
public Customer editActivities(int customerId, List<CustomerActivity> activities) {
    Customer editCustomer = em.find(Customer.class, customerId);
    String sql = "DELETE FROM CustomerActivity ca WHERE ca.customer.id = :id";
    em.createQuery(sql).setParameter("id", editCustomer.getId()).executeUpdate();
    em.flush();
    em.clear();
    editCustomer = em.find(Customer.class, customerId);
    List<CustomerActivity> newActivities = new ArrayList<>();
    for (CustomerActivity ca : activities) {
        ca.setActivity(em.find(Activity.class, ca.getActivity().getId()));
        ca.setCustomer(editCustomer);
        em.persist(ca);
        newActivities.add(ca);
    }
    System.out.println("*************** Setting cas  for customer");
    editCustomer.setCustomerActivities(newActivities);
    em.find(Customer.class, editCustomer.getId());
    return editCustomer;
}
```

Controller – Edit & Update the Customer Object

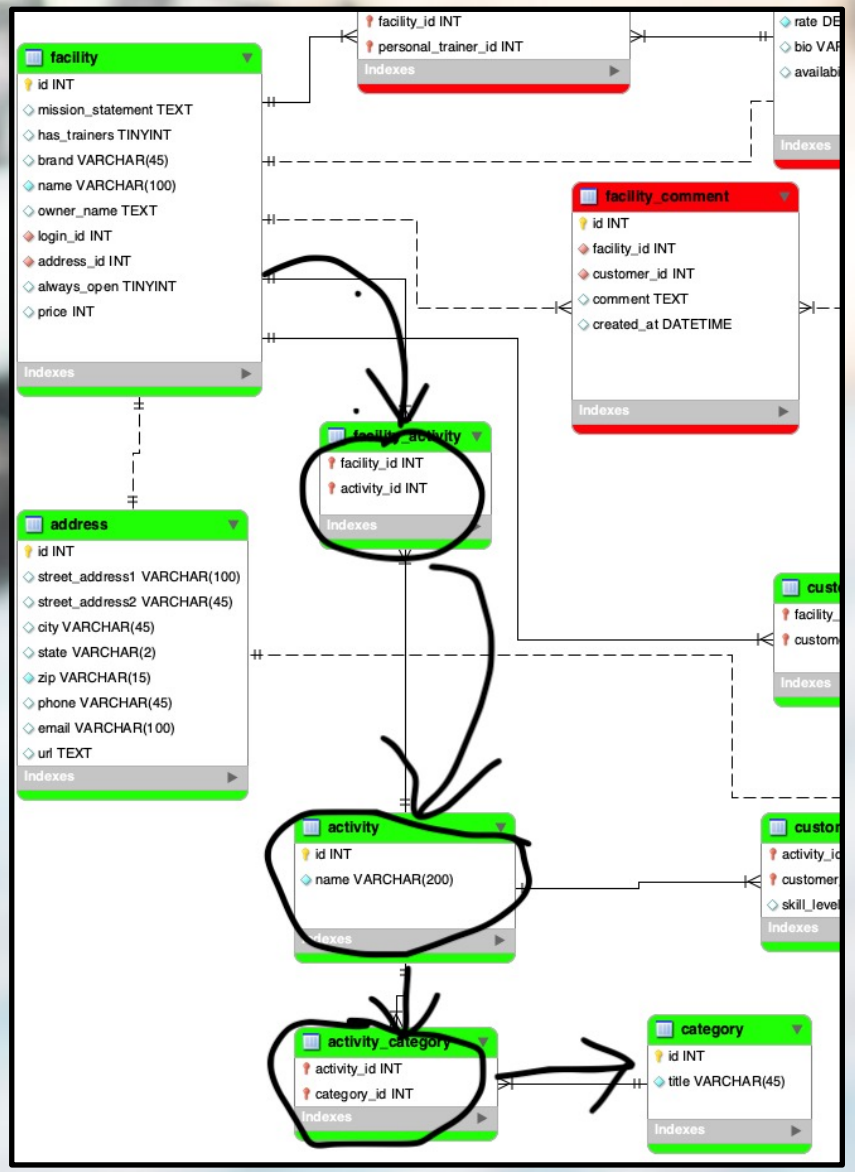
```java
@RequestMapping(path = "editCustomerInfo.do", method = RequestMethod.POST)
public ModelAndView editCustomer(Customer customer, String dob, Address address, Gender gender,
        FacilityPreferences prefs, HttpSession session, @RequestParam("skillLevels") Integer[] skillLevels, @RequestParam("activities") String... activities) {
    ModelAndView mav = new ModelAndView();
    DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");
    customer.setBirthDate(LocalDate.parse(dob, formatter));
    customer.setAddress(address);
    customer.setGender(gender);
    Integer[] usableSkillLevels = new Integer[activities.length];
    int skillCount = 0;
    for (Integer skill : skillLevels) {
        System.out.println(skill);
        if( skill != null) {
            usableSkillLevels[skillCount] = skill;
            skillCount ++;
        }
    }
    List<CustomerActivity> newActivities = new ArrayList<>();
    if (activities != null && activities.length > 0) {
        for (int i = 0; i < activities.length; i++) {
            CustomerActivity ca = new CustomerActivity();
            ca.setActivity(customerDao.findActivityById(Integer.parseInt(activities[i])));
            ca.setSkillLevel(usableSkillLevels[i]);
            ca.setCustomer((Customer)session.getAttribute("customer"));
            newActivities.add(ca);
        }
    }
    Customer editedCustomer = customerDao.editCustomerInfo(customer,((Customer) session.getAttribute("customer")).getId());
    editedCustomer = customerDao.editActivities(editedCustomer.getId(), newActivities);
    editedCustomer = customerDao.editFacilityPreferences(editedCustomer.getId(), prefs);
    editedCustomer = customerDao.findCustomerById(editedCustomer.getId());

    session.setAttribute("customer", editedCustomer);
    mav.setViewName("redirect:editedCustomerInfo.do");
    return mav;
}
```

SQL vs. JPQL

```
SELECT facility.* FROM facility
JOIN facility_activity ON facility.id =
facility_activity.facility_id
JOIN activity ON activity.id =
facility_activity.activity_id
JOIN activity_category ON activity.id =
activity_category.activity_id
JOIN category on
activity_category.category_id =
category.id WHERE category.id = 1;
```

Turns into:

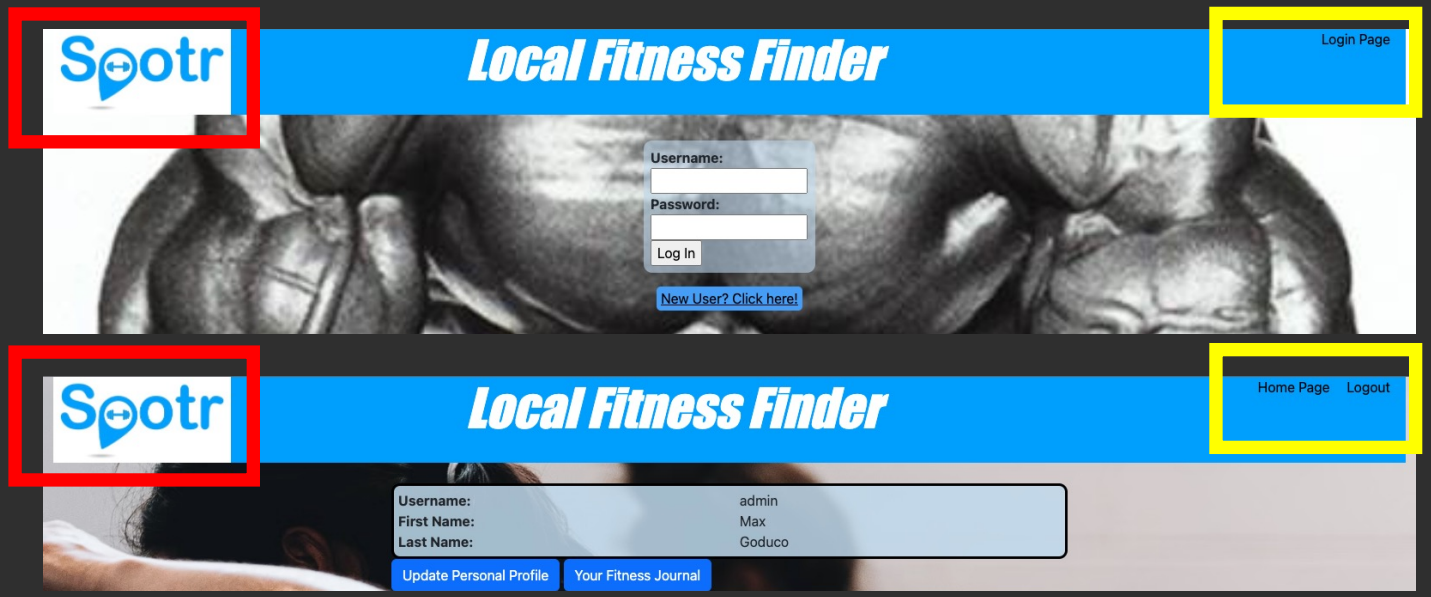
```
SELECT f FROM Facility f
JOIN f.activities fa
JOIN fa.categories fac
WHERE fac.id = :categoryId
```



Frontend – Banner Include with JSP

```
<!-- ADD BOOTSTRAP INTO THE PROJECT -->
<jsp:include page="bootstrapHead.jsp"></jsp:include>
<!-- ADD CSS FILES INTO THE PROJECT -->
<link rel="stylesheet" href="../css/banner.css">

</head>

<body>
```

```
  <div class="banner-wrapper">
    <section class="header">
      <div class="logo">
        <c:if test="${! empty loggedInUser}">
          <c:if test="${! empty customer }">
            <a href="viewCustomer.do"><img class="logo"
              src="../images/spotr_logo_sm.jpg" class="logo" id="logo" /></a>
          </c:if>
          <c:if test="${! empty facility }">
            <a href="viewFacility.do"><img class="logo"
              src="../images/spotr_logo_sm.jpg" class="logo" id="logo" /></a>
          </c:if>
        </c:if>
        <c:if test="${empty loggedInUser }">
            <a href="home.do"><img class="logo"
              src="../images/spotr_logo_sm.jpg" class="logo" id="logo" /></a>
        </c:if>
```




```
      <div class="tagline">
        <h1 class="tagline">Local Fitness Finder</h1>
        <c:if test="${! empty loggedInUser}">
          <c:if test="${! empty customer }">
            <span class="buttons"><a href="viewCustomer.do" class="banner">Home Page</a><a href="logout.do" class="banner">Logout</a></span>
          </c:if>
          <c:if test="${! empty facility }">
            <span class="buttons"><a href="viewFacility.do" class="banner">Home Page</a><a href="logout.do" class="banner">Logout</a></span>
          </c:if>
        </c:if>
        <c:if test="${empty loggedInUser }">
            <span class="buttons"><a href="home.do" class="banner">Login
            Page</a></span>
        </c:if>
      </div>
    </section>
  </div>
```

```
<main class="container-fluid">

  <div class="banner-insert">
    <jsp:include page="banner.jsp"></jsp:include>
  </div>
```

```
  <!-- ADD BOOTSTRAP INTO THE PROJECT -->
  <jsp:include page="bootstrapFoot.jsp"></jsp:include>
</body>
</html>
```

Frontend – Bootstrap, JSTL, CSS, Flex

```
<div id="facilities" class="facilities">
    <div class="field-body">
      <table class="form-alignment">
        <th>Our Facility Provides:</th>
        <thead>
        </thead>
        <tbody>
          <c:forEach var="activity" items="${cFacility.activities}">
            <tr>
              <td>${activity.name}</td>
            </tr>
          </c:forEach>
        </tbody>
      </table>
      <br> <br>
    </div>
    <div class="row">
      <div class="col-sm-4"></div>
      <div class="col-sm-4">
        <br>

        <c:if test="${!customer.facilities.contains(cFacility) }">
          <a href="addFacilities.do?facilityId=${cFacility.id }"
            class=" col-sm-6 offset-3"><button class="btn btn-primary">Add
              Facility</button></a>
        </c:if>
        <c:if test="${customer.facilities.contains(cFacility) }">
          <a href="removeFacilities.do?facilityId=${cFacility.id }"
            class=" col-sm-6 offset-3"><button class="btn btn-primary">Remove
              Facility</button></a>
        </c:if>
      </div>
      <br>
    </div>
    <div class="col-sm-4"></div>
    <br>
  </div>
```

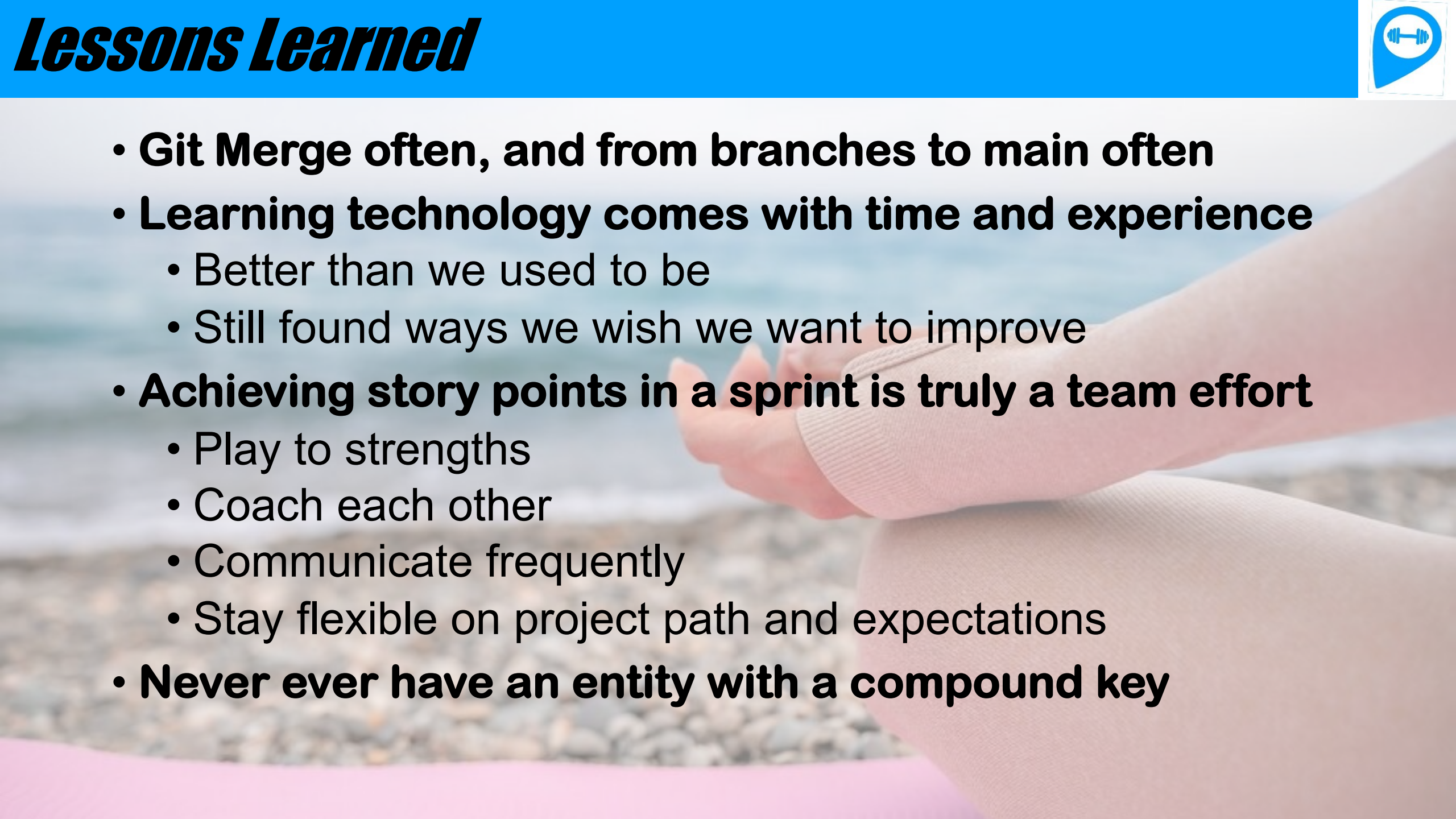
CSS id & class LABELS FOR SPECIFIC FORMATTING

JSTL FOR CONDITIONAL FORMATTING AND DATA

FlexBox TO SIZE ELEMENTS WITH STANDARDS ACROSS PAGES

Bootstrap TO QUICKLY CREATE BEAUTIFUL DESIGN

Lessons Learned



- **Git Merge often, and from branches to main often**
- **Learning technology comes with time and experience**
  - Better than we used to be
  - Still found ways we wish we want to improve
- **Achieving story points in a sprint is truly a team effort**
  - Play to strengths
  - Coach each other
  - Communicate frequently
  - Stay flexible on project path and expectations
- **Never ever have an entity with a compound key**

Trello – Sprint Management

Technologies Used

- **Frontend**
  - HTML/CSS
  - Bootstrap
  - Java Server Pages (JSP)
  - Java Standard Tag Library (JSTL)
- **Backend**
  - Java
  - Spring MVC (Model-View-Control)
  - MySQL Workbench
  - MySQL
  - Gradle
  - Unix Terminal and shell commands
  - Java Persistence API (JPA)
  - Apache Tomcat
- **Collaboration**
  - Git and Github
  - Zoom
  - Slack
  - Trello
- **Research**
  - Google
  - StackOverflow
  - Mozilla Developer Network
  - SD SQUAD

Conclusion & Questions