Innovation Drive and Technology Expo

Design Document

Chris Ciolek

Nicholas Spencer

Maxx Achtman

Apolonio Cazares

[**Introduction**](#_y97mk3yipjzk) **9**

[Goals and Objectives](#_sanicynhkon2) 9

[Statement of Scope](#_u6d94tsivka2) 9

[Functional Requirements](#_qftmc61dnx8f) 9

[Non-Functional Requirements](#_dcwfq2ewx6bg) 9

[Hardware Requirements](#_nlbsszu6ahxm) 10

[Software Requirements](#_zyer5f5w2l) 10

[Inverse Requirements](#_xbxx9yt09dc) 10

[Deliverables:](#_hc61j4wrzehq) 10

[Inputs](#_uo4q8gjgoh9s) 10

[Processing](#_fou09ez8y8qk) 10

[Output](#_pupz23ctd7t5) 10

[Software Context](#_82pz9u73hcrz) 10

[Major Constraints](#_1khsly7jnwt3) 11

[**Data Design**](#_t7a3yk20rjhj) **11**

[Internal Software Data Structure](#_97zc0bkl94xh) 11

[Global Data Structure](#_97zc0bkl94xh) 11

[Temporary Data Structure](#_97zc0bkl94xh) 11

[Database Description](#_97zc0bkl94xh) 11

[**Architectural and Component-Level Design**](#_sf0xhw1f87pl) **12**

[Program Structure](#_ddnkjo18askb) 12

[Architecture Diagram](#_vs51zoxy2nlv) 13

[Alternatives](#_vs51zoxy2nlv) 13

[Description for UI](#_fmd58af8isz0) 14

[Processing Narrative for UI](#_2zzqjnajhzi) 14

[UI Description](#_2zzqjnajhzi) 14

[UI Open Webpage](#_2zzqjnajhzi) 14

[Input](#_cstwsl77wbgb) 14

[Output](#_q8av9xcr6b2) 14

[Algorithmic Model](#_cstwsl77wbgb) 14

[Restrictions / Limitations](#_cstwsl77wbgb) 14

[Local Data Structures](#_4q02i05ent6a) 14

[Performance Issues](#_mak0i6xmz3e2) 14

[Design Constraints](#_ikc205vk3zsy) 15

[Size of the screen for viewing the website would result in dynamically changing the size of the site](#_saabpo4v8vm0) 15

[UI Add User Registration](#_dk4j3x6s6q8v) 15

[Input](#_pp1k0qmydvq0) 15

[Output](#_dbsvp43h5itl) 15

[Algorithmic Model](#_cfvbtwfd3gye) 15

[Restrictions / Limitations](#_g0waua1bn081) 15

[Local Data Structures](#_ezbc7e458hb8) 15

[Performance Issues](#_foe066de3b0k) 15

[UI Add Technology Registration](#_exakk6pjb73r) 15

[Input](#_m780g6hijpc7) 15

[Output](#_uu7flq66qqq7) 15

[Algorithmic Model](#_82idbxsrsxg7) 15

[Restrictions / Limitations](#_8opq2demosh8) 16

[Local Data Structures](#_g9m9b8dh3bfv) 16

[Performance Issues](#_camjpda1g745) 16

[UI Edit User Registration](#_f0jwxcidlc6m) 16

[Input](#_hhj4ilbcuf68) 16

[Output](#_a5n1q3cucqij) 16

[Algorithmic Model](#_2ho65f6ri67n) 16

[Restrictions / Limitations](#_8xlfv5pzl3ez) 16

[Local Data Structures](#_8vk5ftmeezpq) 16

[Performance Issues](#_96bk1l51u6lj) 16

[UI Edit Technology Registration](#_xx2h0pyh9yg2) 16

[Input](#_la21co7cum9l) 16

[Output](#_jiaylh9jx005) 16

[Algorithmic Model](#_44xmmb73hnx5) 17

[Restrictions / Limitations](#_lz8p5vy4egfx) 17

[Local Data Structures](#_v02os1w0depb) 17

[Performance Issues](#_3bd9o9aokg16) 17

[UI Export Data to Excel](#_oukrcprrnwn3) 17

[Input](#_d67wp7ok0aj9) 17

[Output](#_bjas7fvosyyf) 17

[Algorithmic Model](#_ki2nkpvrgdbw) 17

[Restrictions / Limitations](#_dp0i7k561f0p) 17

[Local Data Structures](#_zd6gposw5uyv) 17

[Performance Issues](#_7nntoz49yswp) 17

[UI Change View of Webpage](#_22avhmy745be) 18

[Input](#_910yxsl4scoe) 18

[Output](#_xaw1pthq5v9i) 18

[Algorithmic Model](#_w82dvq235xzf) 18

[Restrictions / Limitations](#_564a4uwtn9y1) 18

[Local Data Structures](#_1dm01ddwr8lm) 18

[Performance Issues](#_ub59glhiq9z5) 18

[UI Digital Waiver](#_kj9ar8gruhv8) 18

[Input](#_bvu4daxypvwa) 18

[Output](#_raj67fps5cga) 18

[Algorithmic Model](#_wr1l0pjzo794) 18

[Restrictions / Limitations](#_3m00cz492jgv) 18

[Local Data Structures](#_1wanyl2f7zsu) 18

[Performance Issues](#_gv2nn3jkkprl) 19

[Description for Logic](#_5jyp8co09gzd) 19

[Processing Narrative for Logic](#_rp2ljufmzr58) 19

[Logic Description](#_1jcuqhpedic) 19

[Logic for changing page](#_6hdpb8v3xhpy) 19

[Input](#_cstwsl77wbgb) 19

[Output](#_355jpd6o2lku) 19

[Algorithmic Model](#_cstwsl77wbgb) 19

[Restrictions / Limitations](#_cstwsl77wbgb) 19

[Local Data Structures](#_cstwsl77wbgb) 19

[Performance Issues](#_4hfh3cw9doll) 19

[Design Constraints](#_8dsurijmla2l) 20

[Logic for Registering User](#_imh5xjckb9bq) 20

[Input](#_vemx5rgj1g72) 20

[Output](#_iiq5fivljtft) 20

[Algorithmic Model](#_4a9ffcqr4d2e) 20

[Restriction / Limitations](#_xpemshgvqdn8) 20

[Local Data Structures](#_u87begtl7web) 20

[Performance Issues](#_aq6zvz9vna92) 20

[Design Constraints](#_o0dkql8j062c) 20

[Logic for Registering Technology](#_v42zxvc8omkr) 20

[Input](#_d9ptdz14rhfy) 20

[Output](#_uasc2qn18gm) 20

[Algorithmic Model](#_p1f3lscuohzv) 20

[Restriction / Limitations](#_ddjstg5j7pl1) 21

[Local Data Structures](#_4xwkzczchk2o) 21

[Performance Issues](#_rv7thz50r8le) 21

[Design Constraints](#_f9ftzuoavt2l) 21

[Logic for Changing User information](#_dl2rxp74e0g) 21

[Input](#_4k62iq6zjeyb) 21

[Output](#_4sglrcnmz4vk) 21

[Algorithmic Model](#_tsk18ly2l2ed) 21

[Restriction / Limitations](#_4iaqharp0zg) 21

[Local Data Structures](#_h1vhrxx33v8t) 21

[Performance Issues](#_4fw4t8ew7i6h) 21

[Design Constraints](#_ve9f7ug48k4n) 22

[Logic for Changing Technology information](#_nxivylghs2ma) 22

[Input](#_7puucgt635pq) 22

[Output](#_i856ocyvv4ev) 22

[Algorithmic Model](#_4mmlov7vdk0q) 22

[Restriction / Limitations](#_m6pdlwxiz65t) 22

[Local Data Structures](#_8adbqe0ap4m) 22

[Performance Issues](#_dvk45qummzed) 22

[Design Constraints](#_n9xhedt4yqb) 22

[Logic for Exporting data to Excel](#_uynnr9i6sxs7) 22

[Input](#_dznyvnaqwonq) 22

[Output](#_8rklhbp58acv) 22

[Algorithmic Model](#_yenpcjo9a71o) 22

[Restriction / Limitations](#_z41ax1la2ftg) 23

[Local Data Structures](#_41o8uaj5mv9) 23

[Performance Issues](#_e2coaksuiomr) 23

[Design Constraints](#_3kweed4zjevk) 23

[Description for Data](#_vs3n5cod2tjo) 23

[Processing narrative for Data](#_yo4j4cc0ripx) 23

[Data Interface Description](#_3gnetsywe9yb) 23

[Adding User to the database](#_e8vbdrxs95fe) 23

[Input](#_3ezcjhqwzlay) 23

[Output](#_kwt2l3s9dycf) 23

[Algorithmic Model](#_iuzfw6ea1ofn) 24

[Restrictions / Limitations](#_cbyzrasuj7gw) 24

[Performance Issues](#_mesp0vw21eqe) 24

[Design Constraints](#_l4eqafz3cvxi) 24

[Adding Technology to the database](#_dhgnqut1iwzz) 24

[Input](#_f4865aaxdzyk) 24

[Output](#_g27z862txcsp) 24

[Algorithmic Model](#_3r8ekycfgww9) 24

[Restrictions / Limitations](#_1bz9f4xr270d) 24

[Performance Issues](#_9g6faej3pmsd) 24

[Design Constraints](#_5prt4urva2ep) 24

[Updating User to the database](#_bn0qsyfk8bsf) 24

[Input](#_sc3bdl4k0wx4) 24

[Output](#_ehvsjdvuu5pc) 24

[Algorithmic Model](#_gyx3c0laynmy) 25

[Restrictions / Limitations](#_h4tr9zkuvfrr) 25

[Performance Issues](#_7o3y16dojc36) 25

[Design Constraints](#_sh9n2f9xbs5e) 25

[Updating Technology to the database](#_atz9xjgq8z6m) 25

[Input](#_9pw7kzqkue5l) 25

[Output](#_k3dsrfg6646t) 25

[Algorithmic Model](#_cil9w5ha2pyj) 25

[Restrictions / Limitations](#_4q7lu1qrtdua) 25

[Performance Issues](#_mztykxpz4vr9) 25

[Design Constraints](#_6s062seovhe) 25

[Software Interface Description](#_2gyu8mr14xh6) 26

[External Machine Interfaces](#_sfx2qyhp9rku) 26

[External System Interfaces](#_sfx2qyhp9rku) 26

[Human Interface](#_sfx2qyhp9rku) 26

[**User Interface Design**](#_7osf7vj7huls) **26**

[Description of the User Interface](#_s3chhy4b1t0u) 26

[Screen Images](#_pwfjacz2bbyw) 26

[Objects and Actions](#_pwfjacz2bbyw) 30

[Interface Design Rules](#_lcxxa2ladupj) 31

[Components Available](#_lcxxa2ladupj) 31

[UIDS Description](#_lcxxa2ladupj) 31

[**Restrictions, Limitations, and Constraints**](#_rtr2hv7tnvf3) **31**

[**Testing Issues**](#_8hl31kevqaky) **32**

[Classes of Tests](#_b3tyjihk6t54) 32

[Expected Software Response](#_16yehishfceh) 33

[Performance Bounds](#_epb8ylgaks77) 34

[Identification of Critical Components](#_nb7h77dop3k1) 34

[**Project Estimates**](#_qldnhqkxcoty) **34**

[Historical Data Used for estimates](#_clqzit3e4fns) 34

[Estimation Techniques Applied and Results](#_mpotpcpfo4r7) 34

[PMP Completion Estimation](#_k6fppojryhio) 34

[Overall Project Estimation](#_83uhux8n0qzl) 34

[Line-of Code Estimate](#_nlksyloo04ih) 35

[Function Estimate](#_9j5jpjgpvyy0) 35

[Total overall project time estimate in hours of effort](#_ke9gt27ixw8k) 35

[Estimation techniques applied and results](#_uit6bz9lgn67) 35

[Estimation Technique 1 - Lines of Code](#_9nqlyk4gt2tk) 35

[Estimation for Technique 1 - Lines of Code](#_639mv1eo2e9w) 35

[Estimation Technique 2 - Function Points](#_54pcr4o635t8) 35

[Estimate of Technique 2 - Function Points](#_kus95cjesa3e) 36

[Estimation technique 3 - COCOMO](#_e61vaoceiy5v) 36

[Estimate for Technique 3 - Final](#_epniw34lxobv) 37

[Project Resources](#_mpotpcpfo4r7) 37

[Budget Estimation](#_mpotpcpfo4r7) 38

[**Risk Management**](#_1537dixfz38l) **38**

[Project Risks](#_dy4i9k4929t) 38

[Risk Table](#_5p4kdmfnf1h4) 38

[Overview of Risk Mitigation, Monitoring, Management](#_v32vtolge7r3) 40

[**Appendices**](#_576a496smqtu) **40**

[Requirements Traceability Matrix](#_ymvzq74rqs1) 40

[Packaging and Installation Issues](#_61gj1t8dxa9z) 41

[Design Metrics to be Used](#_61gj1t8dxa9z) 41

[UML Models](#_61gj1t8dxa9z) 41

[Use Case Model:](#_891uuyic0soh) 41

[Database Diagram:](#_rgrf9nnel5ef) 41

[Control Flow Diagrams:](#_83fr0av9r2jr) 43

[State Diagrams:](#_h7bhdemyskp8) 46

[Control Flow Diagrams:](#_x3a18zri3t15) 49

[Functional Decomposition Diagram:](#_mdydz421xx99) 51

[Task Network Diagram:](#_t3j8ah59juvt) 51

[Use Case Sequence Diagrams:](#_azp24gil2z1i) 52

[Register a Technology](#_8tg6zq7jlq9c) 52

[Pull List of Technologies](#_hhsppwmsaf3w) 53

[Register as Evaluator](#_jbwgzflcvwzh) 54

[Register as Supplier or Ford](#_fy9of0wwf29r) 54

[Modify Registration](#_t3qhf642k8ek) 55

[Sign Waiver](#_y5dy1428i9eg) 56

[Confirm Registration](#_8egasqg5pgo) 56

[Print Attendee Badge](#_pojqwftwkl7u) 57

[Send Email](#_1j7xrk4x23ra) 57

[Open or Close Event Registration](#_15p0nln099mc) 58

[Add or Remove Event Dates](#_6bfuzsi3ftbm) 59

[Check in Attendee](#_siaq1ffnfjsg) 60

[Add or Remove Technology Category](#_gm1ev9pwssrk) 60

[Add or Remove Technology](#_ujpoo4hooyp7) 61

[Generate Badge Sheet](#_okmkp0nyo4zj) 61

[Generate QR Code](#_qgu1w2nb6egi) 62

# 

# 

# 

# 

# 

# 

# Introduction

## Goals and Objectives

The goal for this software is to allow users to select dates and register to be a participant in Ford's IDTE event, and receive confirmation emails regarding the event. Additionally, each user will have a generated QR code on their ticket that will let them check in when they arrive at the IDTE.

## Statement of Scope

### Functional Requirements

* + - 1. User can register as a supplier, presenter, or evaluator
      2. A waiver needs to be signed before a user can sign up to attend the event
      3. Attendees for the event can modify their registration at any point before an event when asking an admin
      4. ~~Administration confirms a registration from a requested attendee~~
      5. Administration can view how many attendees there would be on a given day for the event
      6. A superadmin would be created that can have access to administrative accounts
      7. Administration can change dates of events
      8. Administration can open or close the registration dates for events
      9. Email that is sent to the attendee contains a QR code with their information provided during registration
      10. Badges would be printed on the day of the event by administration
      11. Administration can view technologies that are submitted by users
      12. Administration can export databases from Attendee and Technology Databases into excel documents
      13. Scan QR Code Page can handle both camera images and imported images
      14. Attendee Checkin can search for specific attendee and populate the attendee ID

### Non-Functional Requirements

* + - 1. Application can run on mobile
      2. Application can run on any web browser
      3. Database is secure and cannot be accessed outside of select individuals
      4. ~~User information for attendees can only be viewed by their own person~~
      5. Information stored in database is only for IDTE
      6. Administration would be the only ones to view any attendees
      7. Administration would be the only ones to view any technologies

### Hardware Requirements

* + - 1. Windows OS
      2. Mac OS
      3. Web camera
      4. Mobile Phone

### Software Requirements

* + - 1. Node.Js
      2. HTML
      3. CSS
      4. JavaScript
      5. Java
      6. Sqlite

### Inverse Requirements

N/A

### Deliverables:

#### Inputs

* + - * 1. User credentials
        2. First Name
        3. Last Name
        4. Email Address
        5. Association with event
        6. Dates to attend event

#### Processing

* + - * 1. Data from inputs to database
        2. Generate QR code

#### Output

* + - * 1. Email sent to user with QR code

## Software Context

This web based application would allow users to register for the IDTE event hosted by Ford. A user would have to first access the website and sign a waiver. Once that is completed, the user would then register for the event by providing the necessary information as requested for the database system. Upon completion, the user would be sent an email with a generated QR code that would be presented upon the day of the event.

## Major Constraints

Lack of connectivity to internal Ford Servers for proper testing. Can only run live testing locally until access is granted. Without it, we would be sending the customer a non-fully tested product.

# Data Design

## Internal Software Data Structure

The data structure would be talked through the UI, to a validation system, and transfer it to the database. When an admin wishes to view the current listing of attendees or technologies that are submitted, a link would tell the database to display its contents in a UI fashion table.

## Global Data Structure

The ability to pass back and forth information in the database to the user interface would be denoted here. This is used for storing needed information into the database and make it as easy to allow the database to display the data in a user friendly manner on the site.

## Temporary Data Structure

N/A

## Database Description

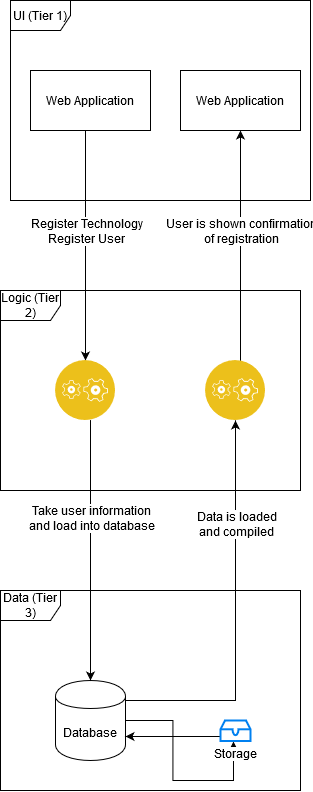
There are two databases that are created within our website. The first one would house the technologies that a user can submit. The fields would be the concept title, description, technology category, concept type, shipping city, shipping country, source, ford contact and presenter, director, and supplier company. The other database is for attendees. This would contain an attendee’s email, first and last name, phone number, city, country, technology to be associated with, and dates of attendance.

# Architectural and Component-Level Design

## Program Structure

Our program is structured on a 3 layer system. This focuses on the UI, the logic that is run behind the scenes, and having a database to store all information with our website. We chose the n-tier structure since our website is built on the three fundamental features, Frontend UI, logic, and backend server database.

### Architecture Diagram



### Alternatives

N/A

## Description for UI

### Processing Narrative for UI

The user would be able to view the web application. From there, a user can choose to do features such as register a technology or registration for IDTE. They could also view information of the event, where it is located, and who to contact. If the user who is on the site is an admin, they would have special rights to their own panel.

### UI Description

User would enter the URL to access our site. This output would be the view of the webpage.

### UI Open Webpage

#### Input

Enter URL

#### Output

Homepage of the website

#### Algorithmic Model

Input the URL, talk with the logic to load the homepage

#### Restrictions / Limitations

N/A

#### Local Data Structures

N/A

#### Performance Issues

N/A

#### Design Constraints

#### Size of the screen for viewing the website would result in dynamically changing the size of the site

### UI Add User Registration

#### Input

Into text fields: Email, Name, Phone Number, Technology Number, Ford Contact

#### Output

User would click the submit button and receive a confirmation email

#### Algorithmic Model

Text boxes are displayed to the user, and the user would enter a string value into each field. Once all fields are filled out, click the submit button

#### Restrictions / Limitations

N/A

#### Local Data Structures

N/A

#### Performance Issues

N/A

### UI Add Technology Registration

#### Input

Into text fields: Technology name, category, Name, email, phone number, Ford contact

#### Output

User would click the submit button and receive a confirmation email

#### Algorithmic Model

Text boxes are displayed to the user, and the user would enter a string value into each field. Once all fields are filled out, click the submit button

#### Restrictions / Limitations

N/A

#### Local Data Structures

N/A

#### Performance Issues

N/A

### UI Edit User Registration

#### Input

User clicks the link in the email to change registration information

#### Output

User would click the submit button and receive a confirmation email

#### Algorithmic Model

User would click their email link to change the registration information. User would change the information where needed. Click the submit button

#### Restrictions / Limitations

N/A

#### Local Data Structures

N/A

#### Performance Issues

N/A

### UI Edit Technology Registration

#### Input

User clicks the link in the email to change registration information

#### Output

User would click the submit button and receive a confirmation email

#### Algorithmic Model

Users would click their email link to change the registration information. Users would change the information where needed. Click the submit button

#### Restrictions / Limitations

N/A

#### Local Data Structures

N/A

#### Performance Issues

N/A

### UI Export Data to Excel

#### Input

Admin would click a button on the admin page view to export the databases to an excel document.

#### Output

Excel document would be created and can be viewable to the user

#### Algorithmic Model

User is an admin. Select the admin page. Select Export to Excel button

#### Restrictions / Limitations

User would have to be an admin

#### Local Data Structures

N/A

#### Performance Issues

N/A

### UI Change View of Webpage

#### Input

User selects an option to change the view of the website to a new page

#### Output

User is shown the new page on the website

#### Algorithmic Model

Select button on navigation view. User would be shown the selected page

#### Restrictions / Limitations

N/A

#### Local Data Structures

N/A

#### Performance Issues

N/A

### UI Digital Waiver

#### Input

User would select Register Technology or Register User

#### Output

User is shown a digital waiver that needs to be signed

#### Algorithmic Model

Select Register Technology or Register User. Enter name in the text box available. Select submit button

#### Restrictions / Limitations

N/A

#### Local Data Structures

N/A

#### Performance Issues

N/A

## Description for Logic

### Processing Narrative for Logic

The logic layer would run the commands that are selected by the user. These options could be viewing a certain page on the webpage, viewing the values in the database, exporting an excel document of the databases, etc. It all depends on user restrictions that are defined by an admin.

### Logic Description

Commands would be run depending on the selections chosen by the user. When a button is selected, the user would be taken to that page view or the frontend would talk to the backend.

### Logic for changing page

#### Input

Click on a button on the UI to load a new section of the web app

#### Output

Logic behind the scenes is run and would talk to the database or load a new page. Depending on the user input

#### Algorithmic Model

Click on button, wait for talk to the server, load the changes

#### Restrictions / Limitations

Need to be connected to website

#### Local Data Structures

N/A

#### Performance Issues

N/A

#### Design Constraints

Size of the screen can affect the button a user can click. If it is small enough, then they may load a different page then what they would want to.

### Logic for Registering User

#### Input

User has data filled out and clicked on the submit button

#### Output

Data is added to the database

#### Algorithmic Model

The logic would check if all necessary fields are filled out. If all fields are filled out, add it to the database. If not, notify user with an error and allow them to fill out missing fields

#### Restriction / Limitations

N/A

#### Local Data Structures

N/A

#### Performance Issues

N/A

#### Design Constraints

N/A

### Logic for Registering Technology

#### Input

User has data filled out and clicked on the submit button

#### Output

Data is added to the database

#### Algorithmic Model

The logic would check if all necessary fields are filled out. If all fields are filled out, add it to the database. If not, notify user with an error and allow them to fill out missing fields

#### Restriction / Limitations

N/A

#### Local Data Structures

N/A

#### Performance Issues

N/A

#### Design Constraints

N/A

### Logic for Changing User information

#### Input

User would have their registration loaded and changed

#### Output

Data is updated to the database

#### Algorithmic Model

The logic would check if any changes are made. If there are, then it will update the values in the database. If no changes are made, then nothing in the database would change.

#### Restriction / Limitations

N/A

#### Local Data Structures

N/A

#### Performance Issues

N/A

#### Design Constraints

N/A

### Logic for Changing Technology information

#### Input

User would have their technology information loaded and changed

#### Output

Data is updated to the database

#### Algorithmic Model

The logic would check if any changes are made. If there are, then it will update the values in the database. If no changes are made, then nothing in the database would change.

#### Restriction / Limitations

N/A

#### Local Data Structures

N/A

#### Performance Issues

N/A

#### Design Constraints

N/A

### Logic for Exporting data to Excel

#### Input

An admin would make the selection to export the database to an excel document

#### Output

The logic would generate an excel file from the data pulled in the database

#### Algorithmic Model

The logic would look at the database. Then organize the data according to columns. Place each value into the correct cells. Export to an excel file.

#### Restriction / Limitations

Need to be an admin to run this feature

#### Local Data Structures

N/A

#### Performance Issues

N/A

#### Design Constraints

N/A

## Description for Data

### Processing narrative for Data

The Data layer would house the data that is populated within the website. Data included would be Users attending the event, technologies that would be on display, and what days to expect the technologies.

### Data Interface Description

The database is used to house all information for the website. Any inputs loaded into the database would be controlled through the UI layer of the site. From there, logic would be conducted to know what to add into the database or what to change.

### Adding User to the database

#### Input

User has clicked the submit button and logic was ran

#### Output

Data is loaded into the database into the correct locations

#### Algorithmic Model

Database would look at the reference in the table. Place the values according to the pointers in the backend

#### Restrictions / Limitations

N/A

#### Performance Issues

N/A

#### Design Constraints

N/A

### Adding Technology to the database

#### Input

User has clicked the submit button and logic was ran

#### Output

Data is loaded into the database into the correct locations

#### Algorithmic Model

Database would look at the reference in the table. Place the values according to the pointers in the backend

#### Restrictions / Limitations

N/A

#### Performance Issues

N/A

#### Design Constraints

N/A

### Updating User to the database

#### Input

User has clicked the submit button and logic was ran

#### Output

Data is loaded into the database into the correct locations

#### Algorithmic Model

Database would look at the reference in the table. Place the values according to the pointers in the backend

#### Restrictions / Limitations

N/A

#### Performance Issues

N/A

#### Design Constraints

N/A

### Updating Technology to the database

#### Input

User has clicked the submit button and logic was ran

#### Output

Data is loaded into the database into the correct locations

#### Algorithmic Model

Database would look at the reference in the table. Place the values according to the pointers in the backend

#### Restrictions / Limitations

N/A

#### Performance Issues

N/A

#### Design Constraints

N/A

## Software Interface Description

### External Machine Interfaces

N/A

### External System Interfaces

We are talking to an internal Ford server. This will ensure security to the users and technologies that are submitted for IDTE.

### Human Interface

The user would be able to view a web page that is accessible through a web browser.

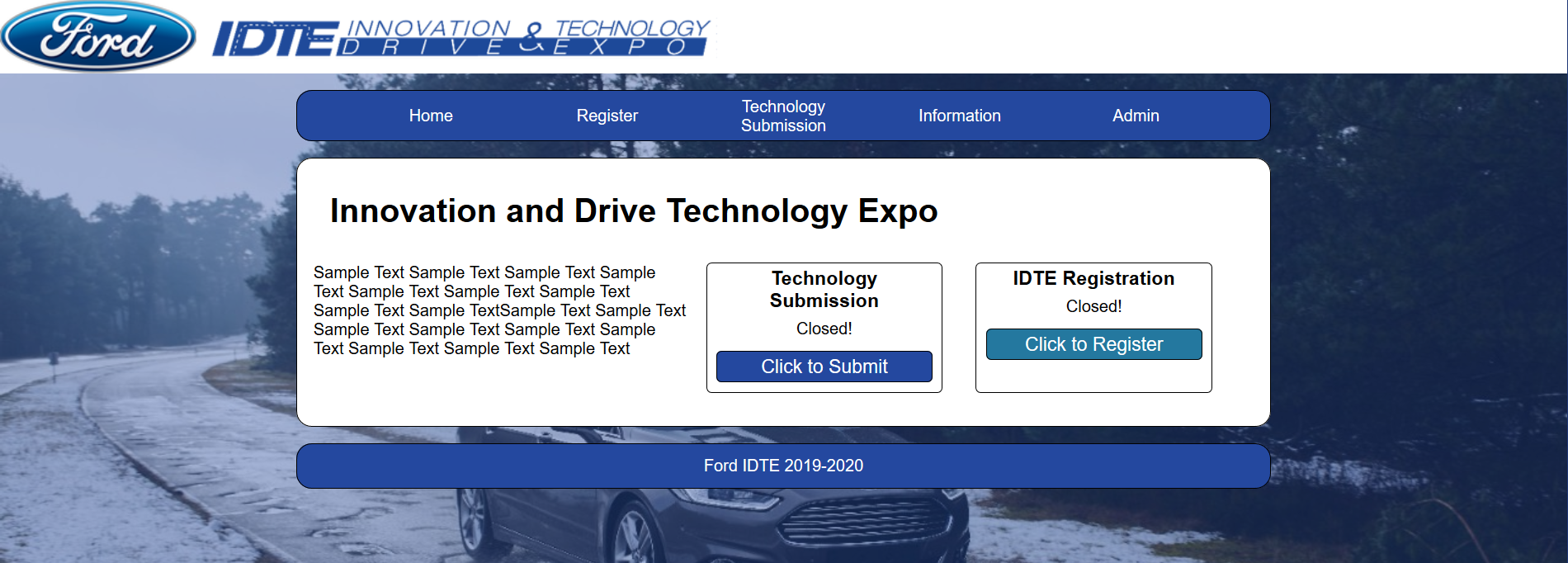
# User Interface Design

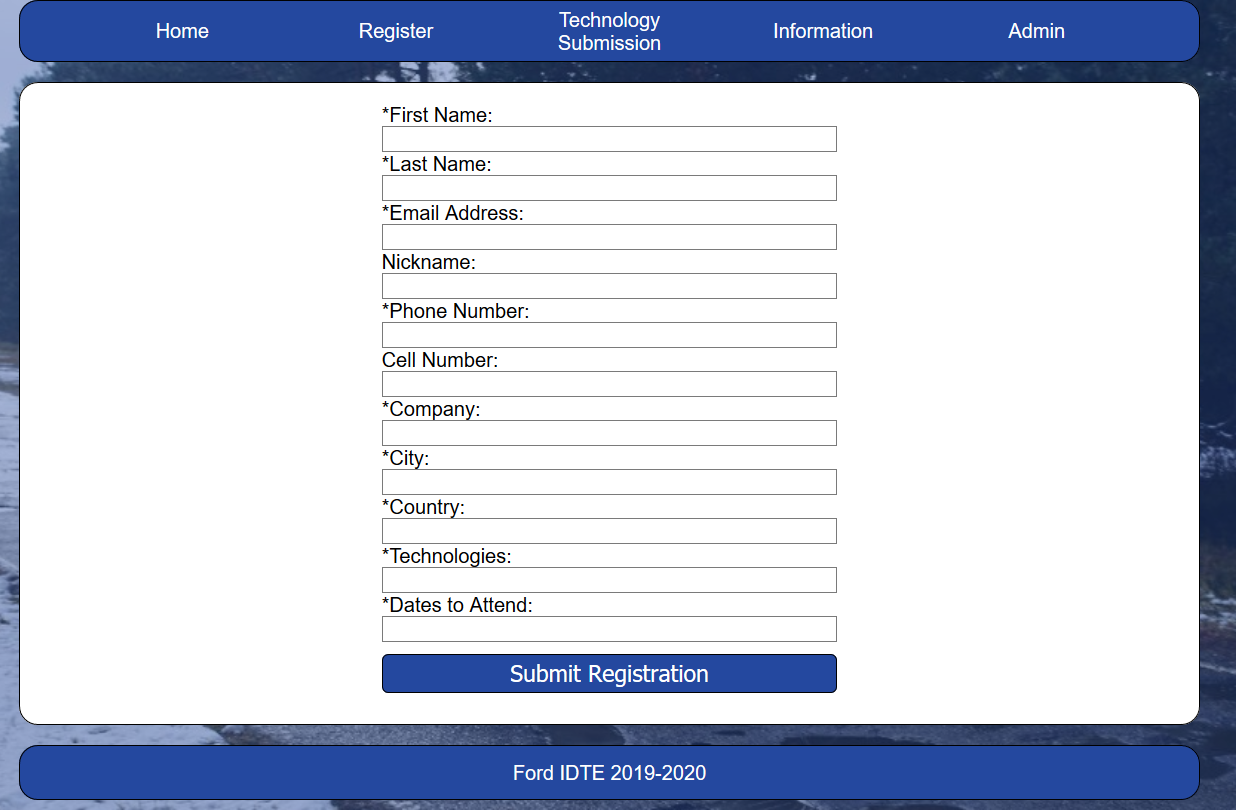
## Description of the User Interface

The user interface is a web page browser. There is a navigation bar for the user to have access to. There are options to submit a technology or register as a user. For registering as a user or registering technology, a user must sign a digital waiver. There is an admin only page view that would be detailed more in screen shots. A drop down is included for information regarding the event.

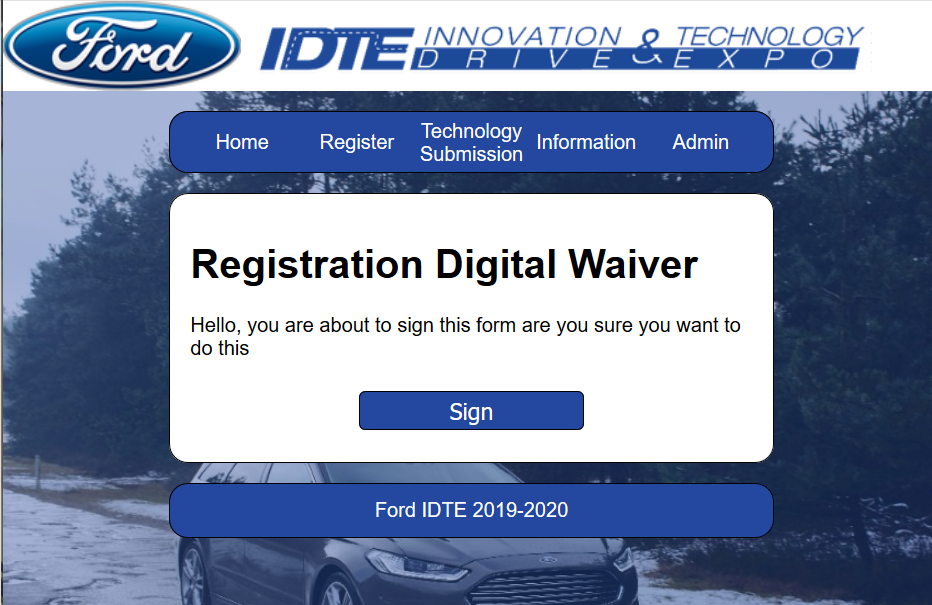
### Screen Images

Homepage:

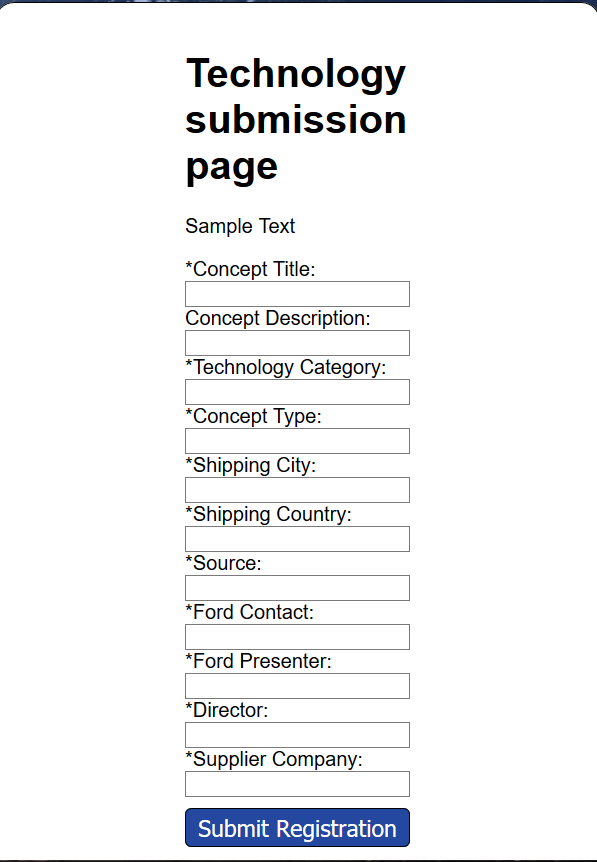
Register User:



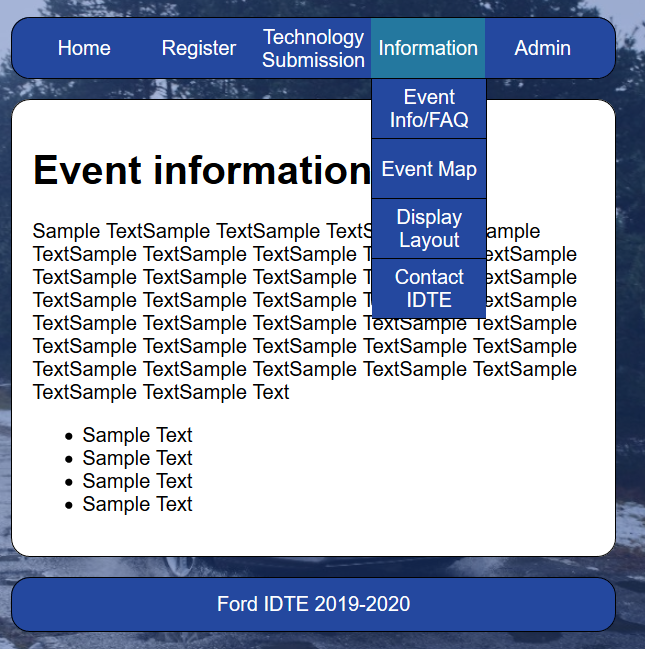
Digital Waiver:



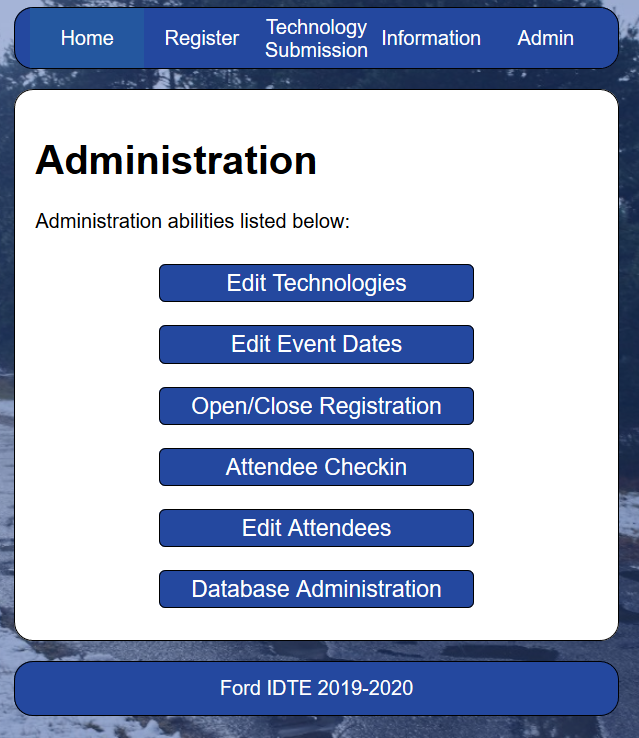
Technology submission page:



Drop down for Information:



Admin page:



### Objects and Actions

* Clicking the sign button on digital waiver takes user to the technology or user page
* Selecting the home button takes the user to the homepage
* Selecting the register button would allow the user to choose to register as a supplier or evaluator
* Selecting technology submission option would allow the user to register a technology
* Hovering over the information tab would take the user to view options Event Info, Event Map, Display Layout, Contact IDTE
* Selecting one of the above options takes the user to the respective view on the webpage
* Selecting the admin page would take user to the admin page view
* Selecting Edit technologies would allow an admin to make changes to a technology
* Selecting Edi users would allow an admin to make changes to a user
* Admins can open or close registrations for technology and users
* Selecting the attendee checkin button would show current users that are listed to attend
* Selecting Edit Event Dates would change the dates of IDTE, this can ensure the site can be used for continuous years
* Database Administration button would allow an admin to add a new attendee (user)

## Interface Design Rules

* Keep it simple
* Users are in control
* Make user interfaces consistent

## Components Available

* Buttons
* Navigator
* Drop downs
* Text fields
* Labels

## UIDS Description

* Node.Js
* HTML
* CSS
* JavaScript
* Java
* MySQL

# Restrictions, Limitations, and Constraints

* 1. A proposed earlier date for the project may prove difficult to meet each respective requirement for the software. Additionally, the expedited schedule may sacrifice some overall software stability for an earlier delivery.
  2. Due to the single threaded nature of Node, CPU heavy processes may slow down the software as a whole while new functions are queued. This also means that if a part of the software causes Node to crash, all requests on the thread will crash as well.
  3. Each team member will be using his own personal computer when working on the project, and as such each machine may perform differently when trying to use the software.

# Testing Issues

## Classes of Tests

* + 1. Logging in
       1. User goes to log in page
       2. User enters username and password
       3. User clicks “Submit”
    2. Registering for an Event
       1. Once logged in on the home screen, the user selects their registration type.
       2. User signs waiver and agrees to the terms and conditions and clicks “Next”
       3. User enters personal information and clicks “Next”
       4. User selects the dates they will be registering for
       5. For each respective date, the user selects a technology number.
       6. User clicks “Next”
       7. A summary of the entered information is displayed and verified by the user for it’s accuracy; once the user is satisfied with the entered information, they click “Submit”.
    3. Changing Registration Info
       1. Once logged in on the home screen, user selects “Modify Registration”.
       2. User searches for their name and clicks “Submit”.
       3. User updates any information as needed, and clicks “Submit”
    4. Deleting Attendee
       1. Once logged in on the home screen, user selects “Modify Registration”.
       2. User searches for their name and clicks “Submit”.
       3. User clicks “Delete Attendee”
       4. User confirms deleting attendee
    5. Attendee Check-in
       1. Once logged in on the home screen, user selects “Admin”.
       2. User selects “Attendee Check-In”
       3. User searches for attendee information and selects “Submit”, or scans the QR code on the attendees ticket.
    6. Adding Technologies
       1. Once logged in on the home screen, user selects “Admin”.
       2. User selects “Technologies”.
       3. User enters technology number and technology name.
       4. User selects “Submit”.
    7. Adding Categories
       1. Once logged in on the home screen, user selects “Admin”.
       2. User selects “Categories”.
       3. User enters category and selects “Submit”.
    8. Setting Registration Dates
       1. Once logged in on the home screen, user selects “Admin”.
       2. User selects “Set Event Dates”
       3. User selects months and days for the dates.
       4. User selects “Submit”

## Expected Software Response

* + 1. Logging in
       1. Login will be granted upon verifying the entered credentials were correct. If the credentials were invalid, the user will not be logged in.
    2. Registering for an event
       1. User’s information will be recorded and the individual will be registered for the entered dates.
    3. Changing Registration info.
       1. The desired information will be updated according to the users needs.
    4. Deleting Attendee
       1. The user will no longer be registered for the entered dates, and registration information for the individual will be removed from the database.
    5. Attendee Check-in
       1. Uer will be checked into the event if they had registered prior to attempting to check-in.
    6. Adding Technologies
       1. A new technology will be added and available for selection by users when registering.
    7. Adding Categories
       1. A new category will be added and available for selection by users when registering.
    8. Setting Registration Dates
       1. Dates for the event will be set and able to be selected by attendees when registering.

## Performance Bounds

* + 1. 6.1.3 and 6.1.4 will also be used to test the software’s response for when an attendee’s information cannot be found within the database.

## Identification of Critical Components

* User Interface
* Valid Data
* Invalid Data
* Data Export

# Project Estimates

## Historical Data Used for estimates

* School Projects
* Work Projects
* Vacation Time
* Weather Conditions
* School Scheduling
* Work Scheduling
* Meetings with client
* SLOC/FP Data from qsm.com
* FP calculator from University of Southern California

## Estimation Techniques Applied and Results

### PMP Completion Estimation

* 100 Hours

### Overall Project Estimation

#### Line-of Code Estimate

* 6527 SLOC average

#### Function Estimate

* 184 Function Points

#### Total overall project time estimate in hours of effort

* Effort: 24.524 person-months
* ~3923 hours
* Per person:
* 6.131 person months, 980.75 hours

## Estimation techniques applied and results

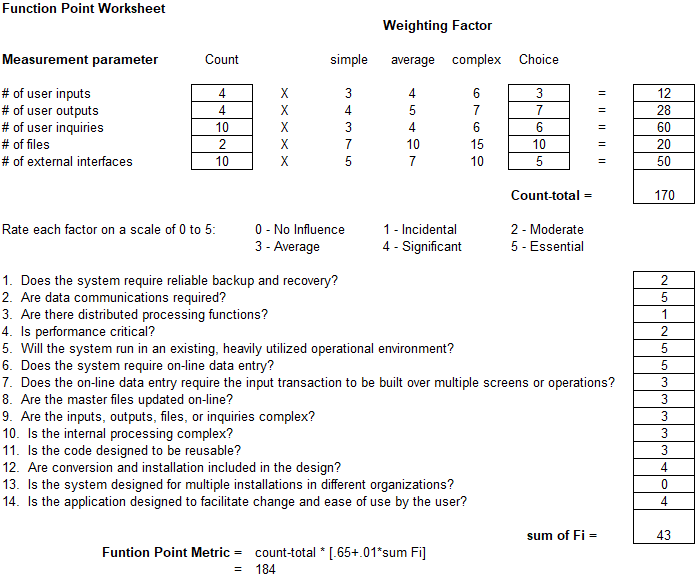
### Estimation Technique 1 - Lines of Code

* Several components to track, including:
* Front end React code, ~100 LOC/page, ~50 LOC/React Component. Estimated ~25 webpages, ~10 React components.  
    
  Front end estimation = 100\*25 + 50\*10 = 3000 LOC
* Server end Nodejs Code, ~30 LOC/CRUD operation, ~150 LOC/database object, ~200 LOC session management, ~200 LOC validation, ~12 CRUD ops, ~3 database objects  
    
  Server side estimation = 30 \* 12 + 150 \* 3 + 200 + 200 = 1210 LOC
* Configuration files for node, babel, and webpack, ~100 LOC NPM config, ~150 LOC webpack config, ~20 LOC babel config  
    
  Configuration estimation = 100 + 150 + 20 = 270 LOC
* Database SQL scripts, ~70 LOC user table, ~40 LOC technology table  
    
  Database estimation = 70 + 40 = 110 LOC

### Estimation for Technique 1 - Lines of Code

* Total estimation is as follows:  
    
  Front end + server end + configuration + database scripts  
  3000 LOC + 1210 LOC + 270 LOC + 110 LOC  
    
  Final Estimation = 4590 LOC

### Estimation Technique 2 - Function Points



* Total amount of function points calculated using calculator created by University of Southern California

### Estimate of Technique 2 - Function Points

* Total number of SLOC using function points and JavaScript as a reference according to QSM resources, is as follows:  
    
  184 \* 46 = 8464

### Estimation technique 3 - COCOMO

* To recap, a component evaluation of SLOC resulted in an estimated 4590 SLOC
* A function point evaluation for SLOC resulted in an estimated 8464 SLOC
* Don’t reconcile well, average is 6527 SLOC
* COCOMO equation is as follows:
* E is the resulting effort required in person-months, a is 3.0 for semi detached, and b is 1.12. KLOC is the line of code total in thousands.
* COCOMO result for LOC evaluation -> E = 16.536
* Result for FP evaluation -> E = 32.809
* Average evaluation -> E = 24.524
* We will continue with the average evaluation

### Estimate for Technique 3 - Final

* Average COCOMO evaluation for effort required was 24.524 person-months
* With a 4 man team, that is 6.131 Months Required for scheduling
* Wage per person at $75/hour over 6.131 months results in $73,572 per person
* 4 people on team, total cost from wages is $294,288

## Project Resources

* + 1. People
       1. Contacts at Ford to help with verification and deployment to Ford servers
    2. Hardware
       1. (Potentially) SuSE server at Ford for development deployment and final production deployment
    3. Software
       1. VSCode for text editing
       2. Nodejs for server side scripting
       3. Reactjs/Angularjs for client side rendering and routing
       4. NPM for package management and development scripts
       5. Babel7 for ES6 JavaScript compilation
       6. Webpack for creating deployment ready assets
       7. Windows/MAC operating systems for developing on
       8. SuSE operating system for deployment
       9. WebSphere WAS for deployment
    4. Tools
       1. Personal computers and laptops for development

## Budget Estimation

* + 1. As pulled from 2.3.6, it will cost the project $294,288 for completion of project in allotted time
       1. **Note:** we are excluding production costs such as databases and servers

# Risk Management

## Project Risks

* Poor Estimation on functionalities added during project creation
* Poor Estimation on Main Functionality
* Project Specifications changes
* Underestimation of Program Complexity
* Absence of Team Leader
* Absence of Other Team Members
* Inclement Weather
* Appearance of new technologies needed for functionalities
* Absence of Client

## Risk Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Risk | Probability | Impact | Mitigation Plan | Contingency Plan |
| Poor Estimation on Functionalities added during project creation | Medium | Low | Get regular updates and design main functionality to not rely on any additional functionalities | Drop support and expectations for respectable additional functionalities |
| Poor Estimation on Main Functionality | Medium | High | Get regular updates from team members to gauge progress, re-evaluate estimations regularly | Drop additional functionalities and focus on main |
| Project Specification changes | High | Medium | Focus on designing the project instead of programming it, design components modularly so if something changes the whole project doesn’t have to be redone | Change components that are affected by specification change |
| Underestimation of program complexity | Medium | Medium | Keep program functionality simple, have algorithms planned before creation of code | Reduce or remove functionality causing problems |
| Absence of Team Leader | Low | Medium | Finish Planning as early as possible, stay in contact | Assign new team leader and work off of what we have |
| Absence of Other Team Members | Low | Medium | Communicate regularly | Pick up the slack from other team members if items start falling behind |
| Inclement weather | Medium | Low | Adhere to schedule | Attempt digital meeting |
| Appearance of new technologies needed for functionalities | Medium | Medium | Attempt to learn early | Redesign plan to omit new technologies |
| Absence of Client | Low | Medium | Communicate regularly with client and raise questions immediately | Continue to focus on tasks that we know of for the project and wait until more news is brought up form the client |

## Overview of Risk Mitigation, Monitoring, Management

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Impact | | | | |
| Probability |  | Low | Medium | High | Critical |
| Low |  |  |  |  |
| Medium |  |  |  |  |
| High |  |  |  |  |
| Legend | | | | | |
| Monitor | | Execute Mitigation Plan | | Execute Contingency Plan | |

* Re-evaluate risks once every one-two weeks

# Appendices

## Requirements Traceability Matrix

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| U/R | R1 | R2 | R3 | R4 | R5 | R6 | R7 | R8 | R9 | R10 |
| US1 | X | X |  |  |  |  |  |  |  |  |
| US2 | X | X |  |  |  |  |  |  |  |  |
| US3 |  |  | X |  |  |  |  |  |  |  |
| US4 |  | X |  |  |  |  |  |  |  |  |
| US5 | X | X |  |  |  |  |  |  |  |  |
| US6 | X | X |  | X |  |  |  |  |  |  |
| US7 |  |  |  | X | X |  | X |  |  |  |
| US8 |  |  |  |  |  |  |  |  |  | X |
| US9 |  |  |  |  |  |  |  |  | X |  |
| US10 |  |  |  |  |  |  |  | X |  |  |
| US11 |  |  |  |  |  |  | X |  |  |  |
| US12 |  |  |  |  |  |  |  |  | X | X |
| US13 |  |  |  |  |  |  | X | X |  |  |
| US14 |  |  |  |  |  |  | X | X |  |  |
| US15 |  |  |  |  |  | X |  |  |  |  |
| US16 |  |  |  |  |  | X |  |  |  |  |
| US17 |  |  |  |  |  |  |  |  | X |  |
| US18 |  |  |  |  |  |  |  |  | X |  |

## Packaging and Installation Issues

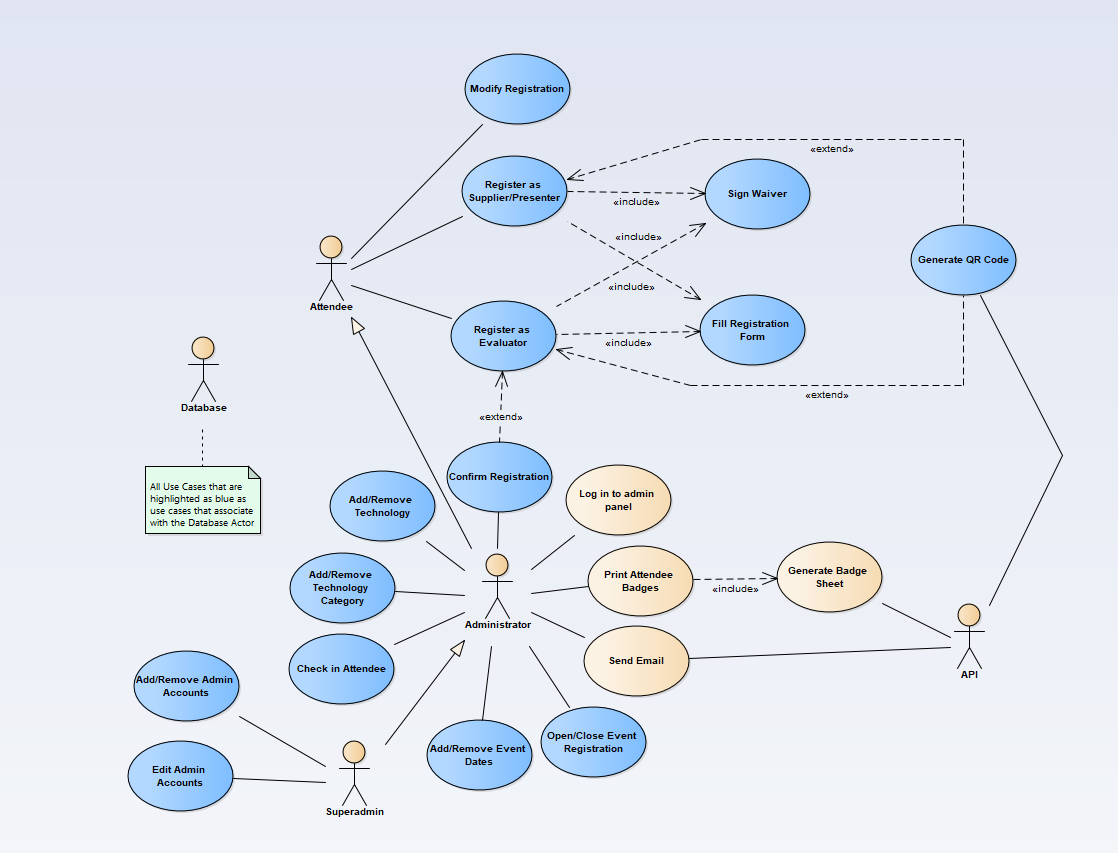
N/A

## Design Metrics to be Used

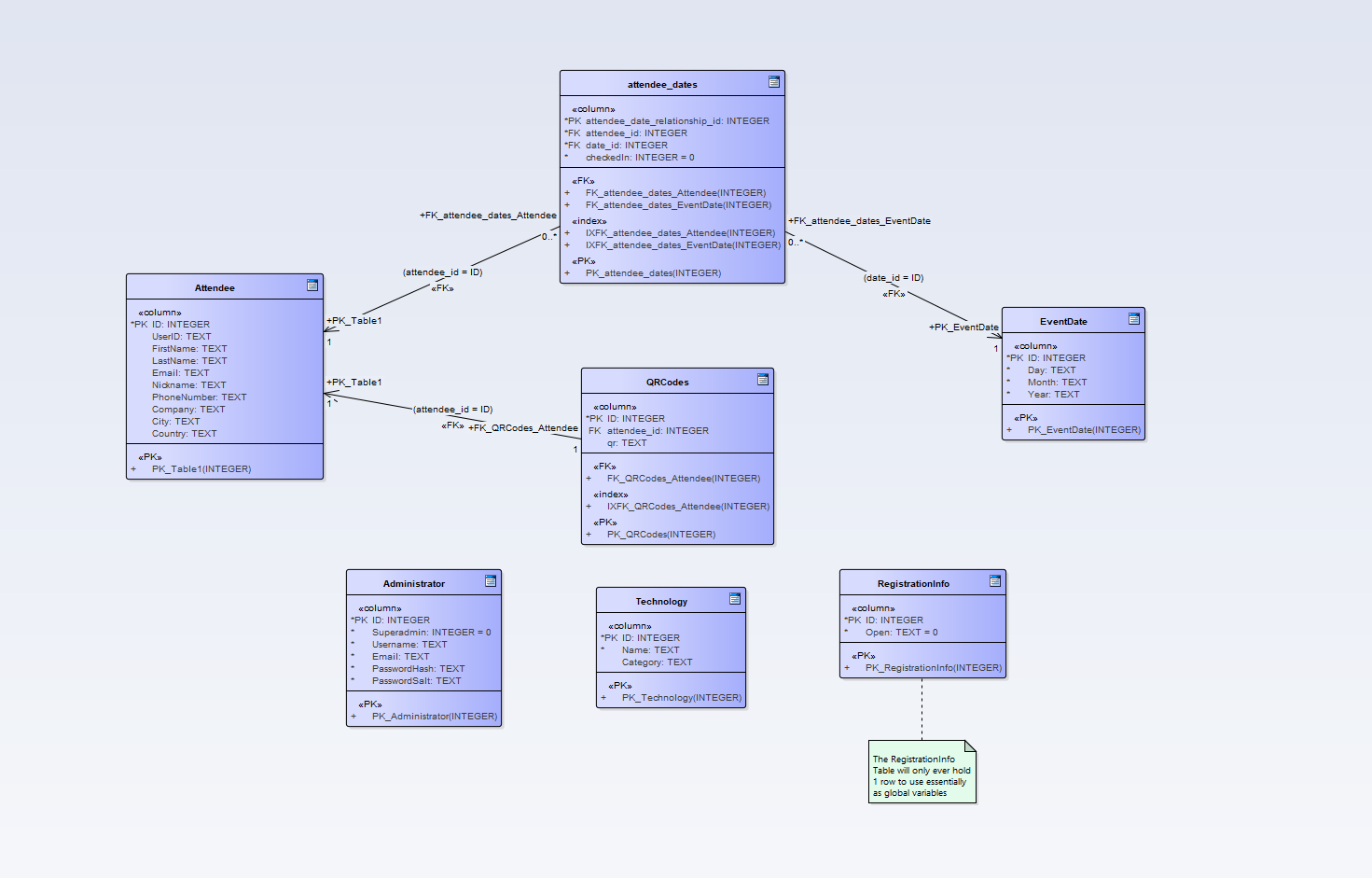
Track efficiency through Kanban board in order to track the work of the team. Have a set goal of points to be completed within two week sprints to ensure a successful project upon completion.

## UML Models

## Use Case Model:

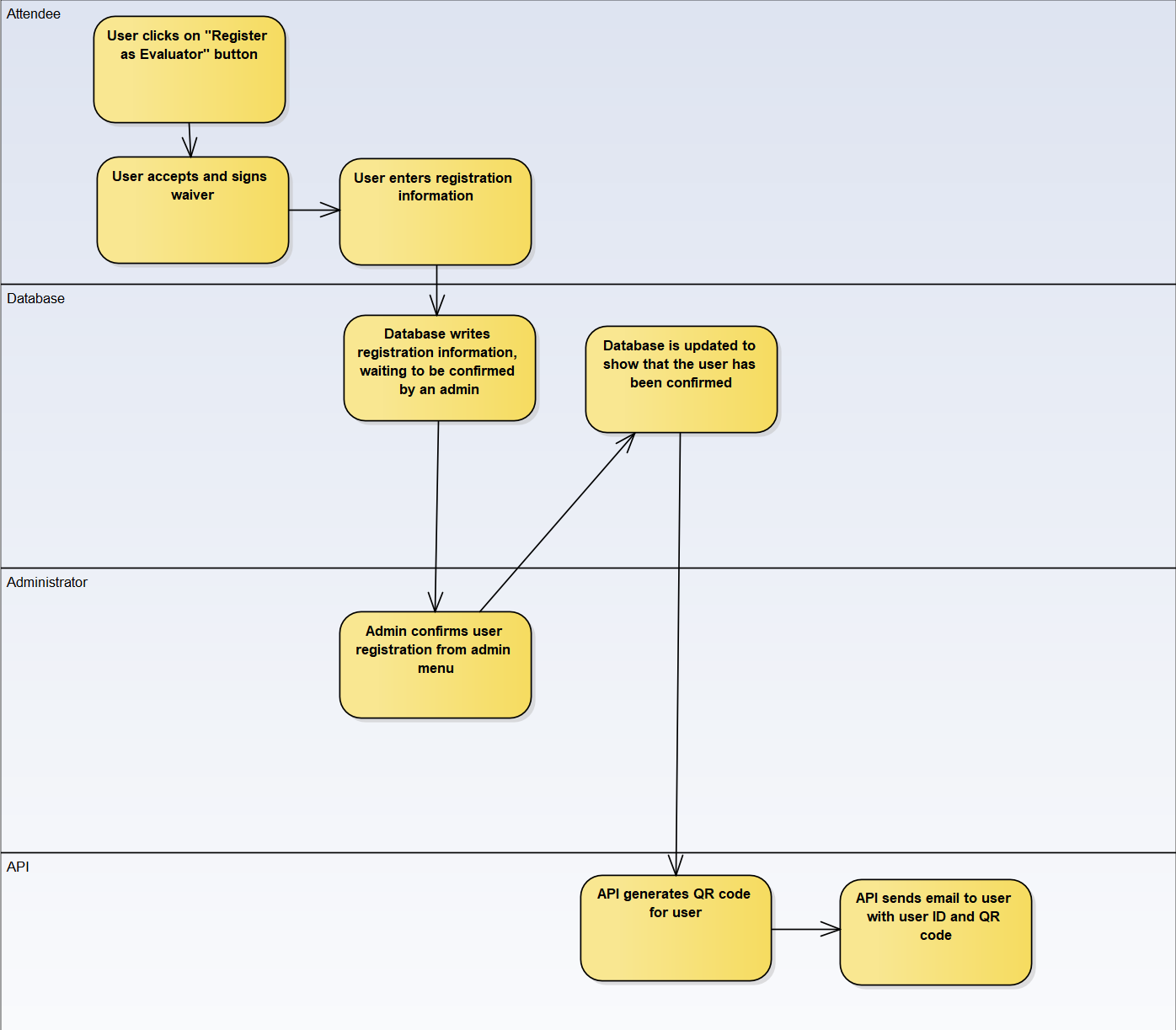


## Database Diagram:

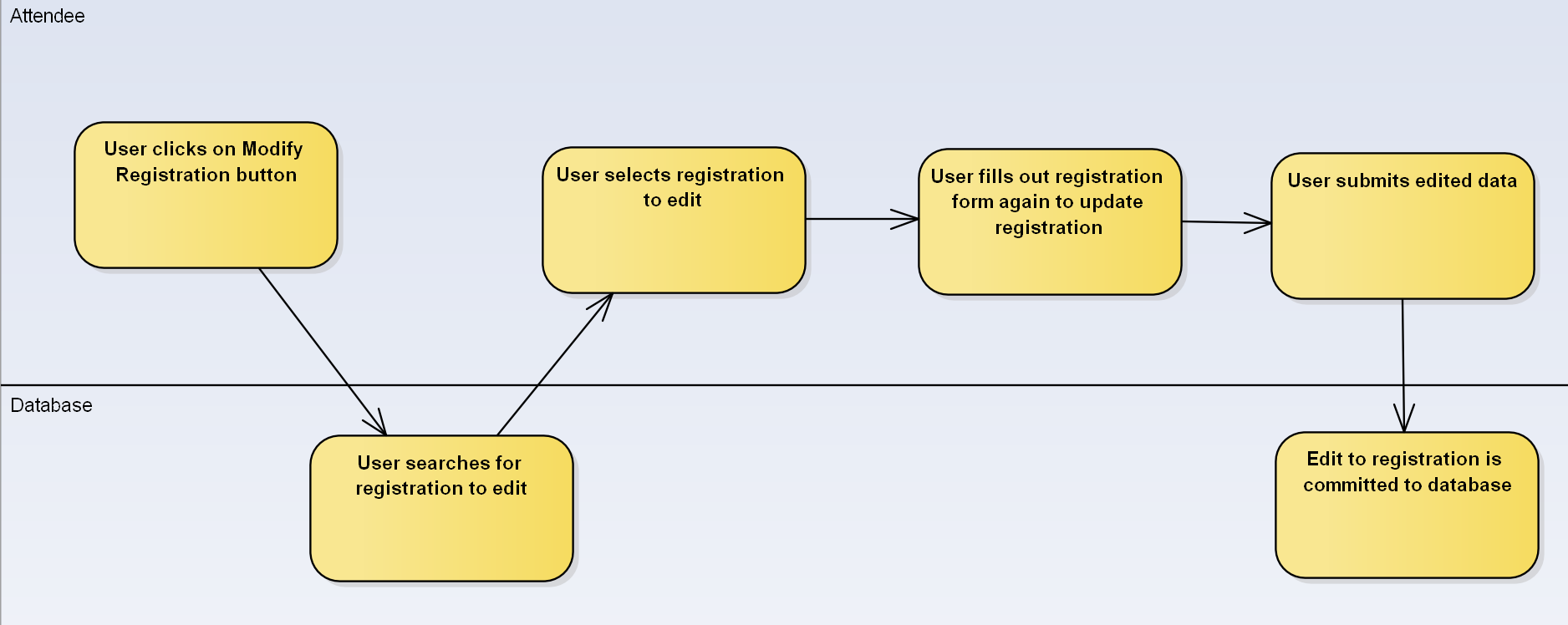


## Control Flow Diagrams:

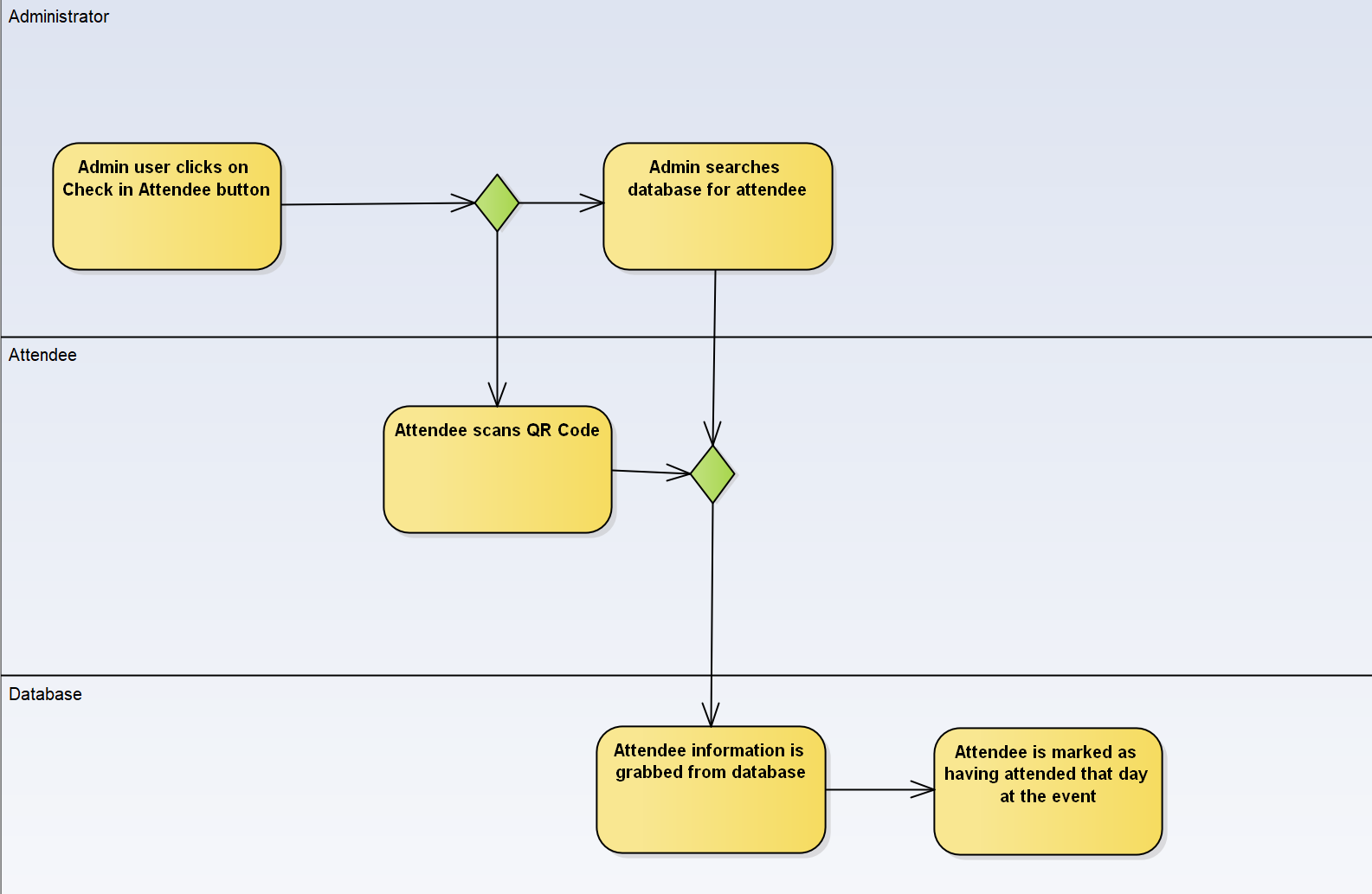
* Attendee Registration:



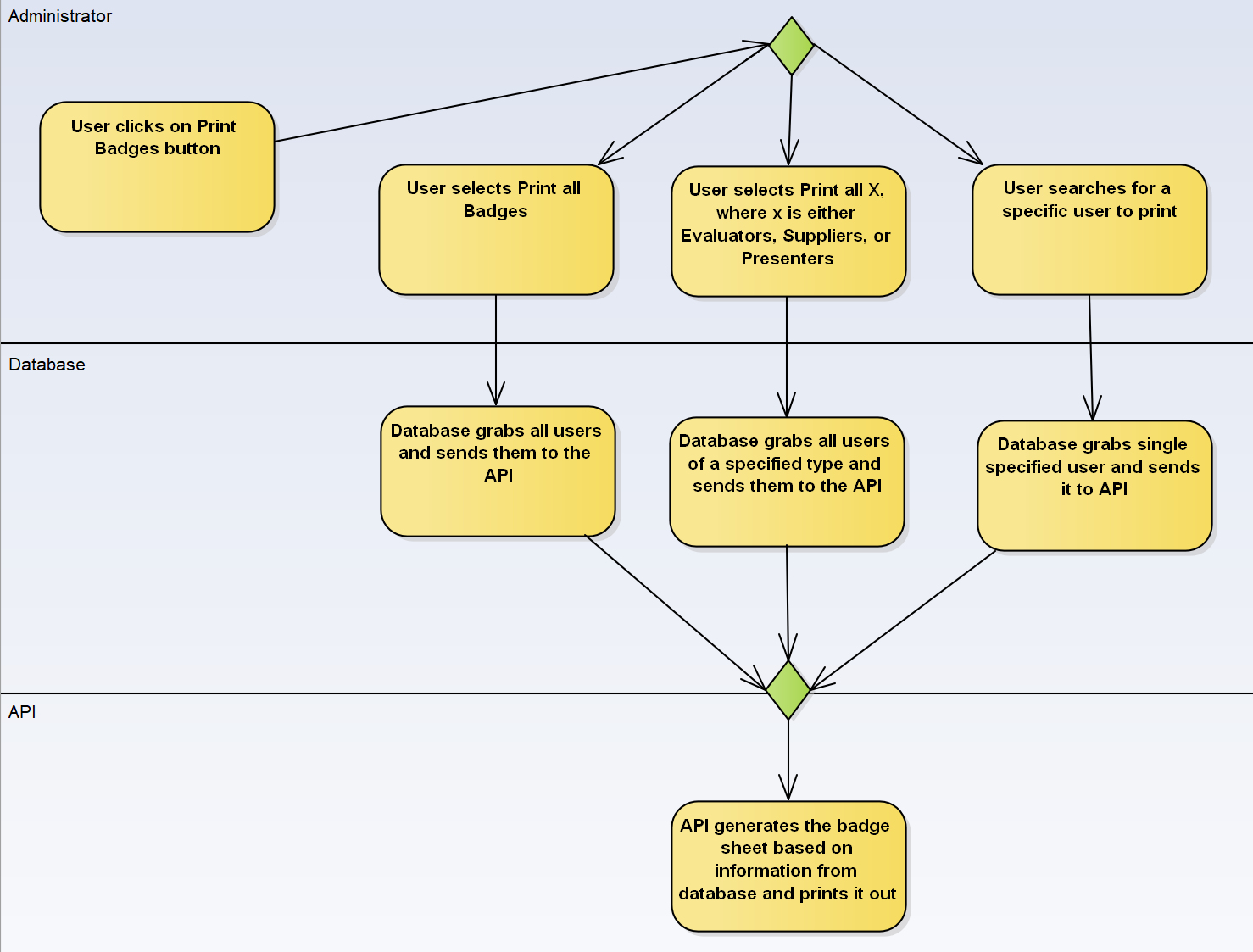
* Modify Registration:



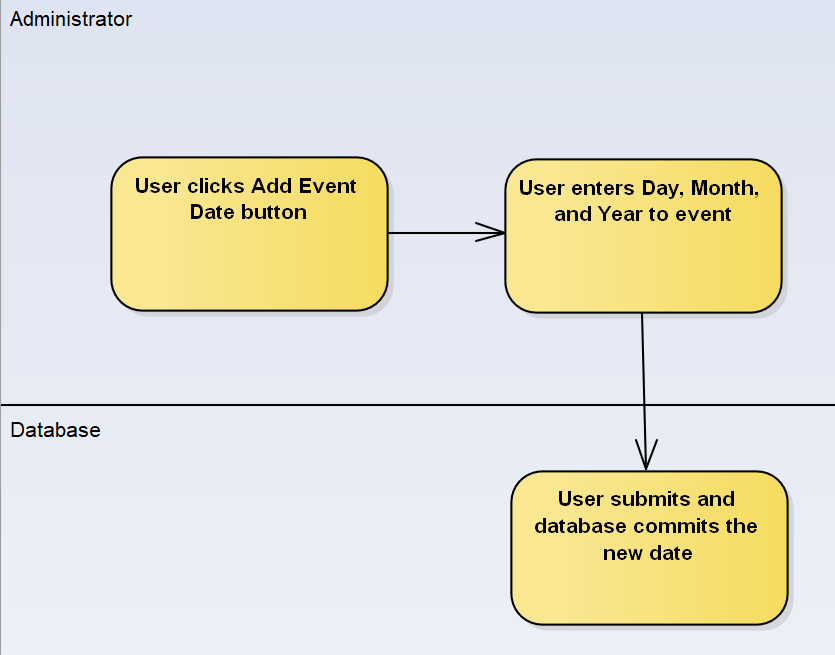
* Attendee Checkin:



* Print Attendee Badges:

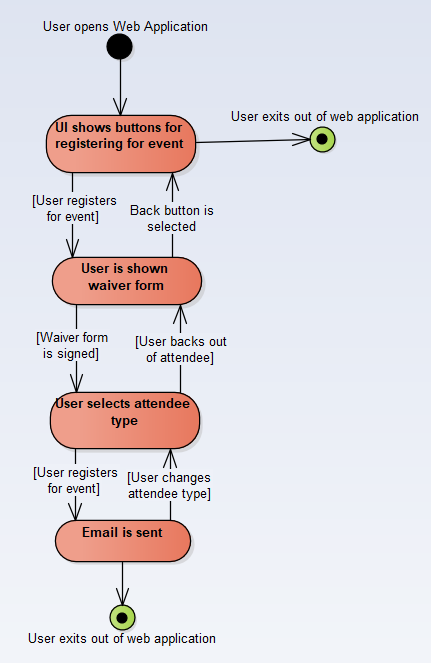


* Create Date for Registration or Technology:

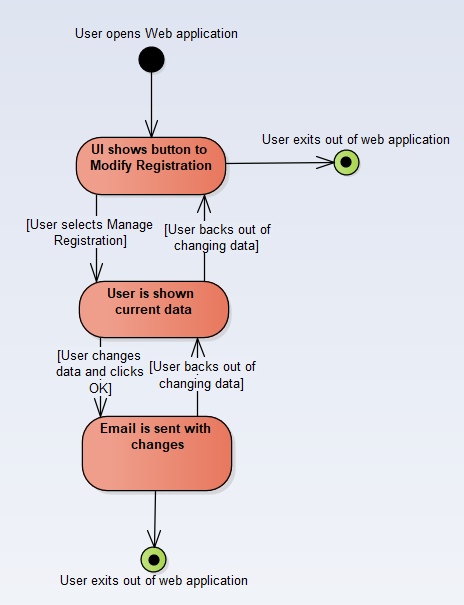


## State Diagrams:

* User Registers for event:

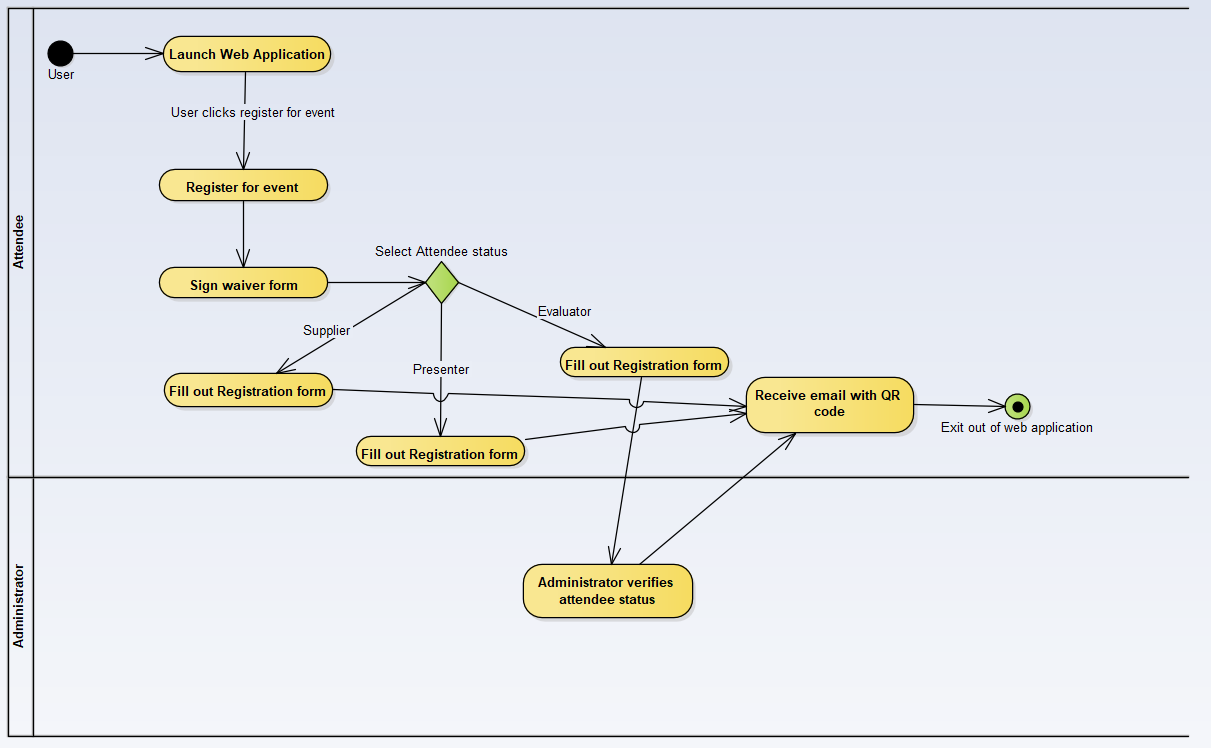
****

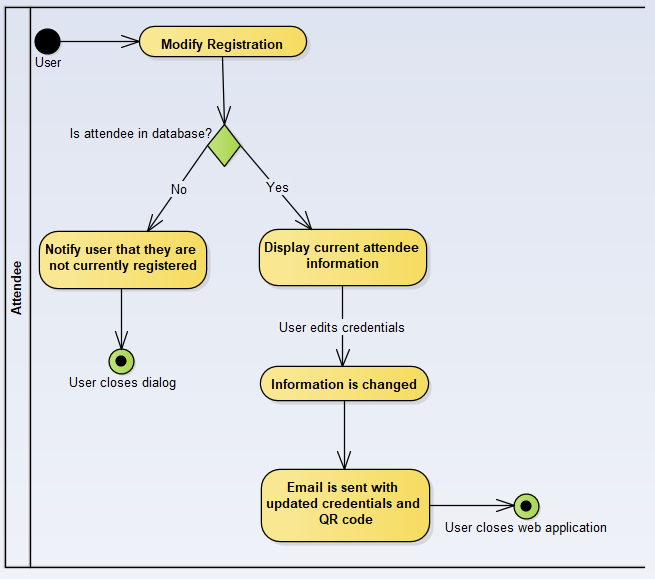
* User Modifies Registration



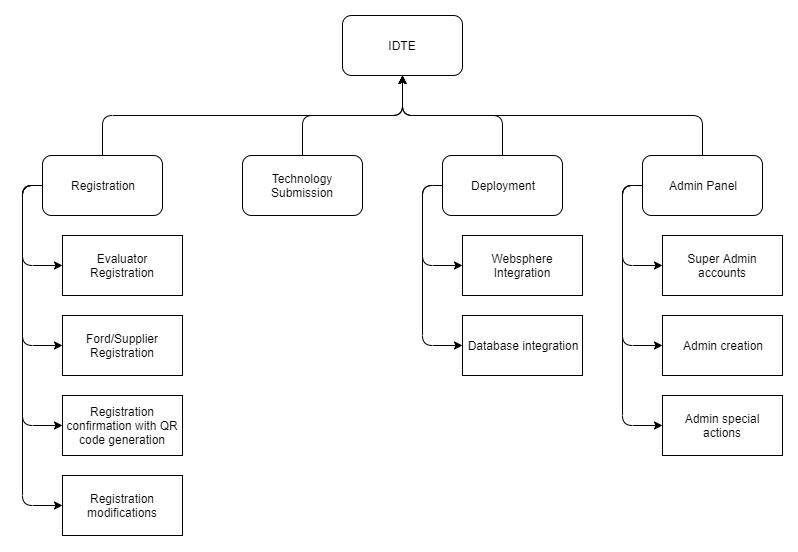
## Control Flow Diagrams:

* User registers for Event:

**-** User Modifies Registration:



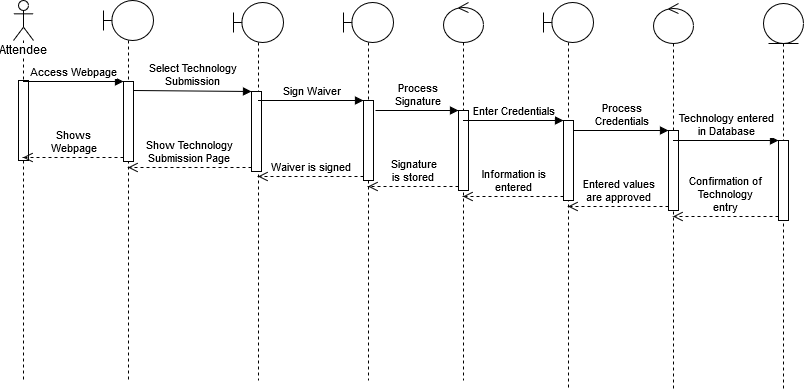
## Functional Decomposition Diagram:



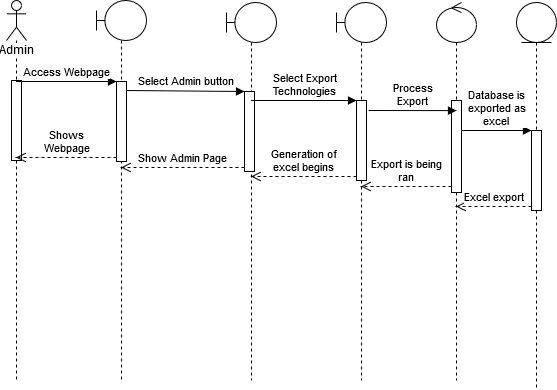
## Task Network Diagram:

## Use Case Sequence Diagrams:

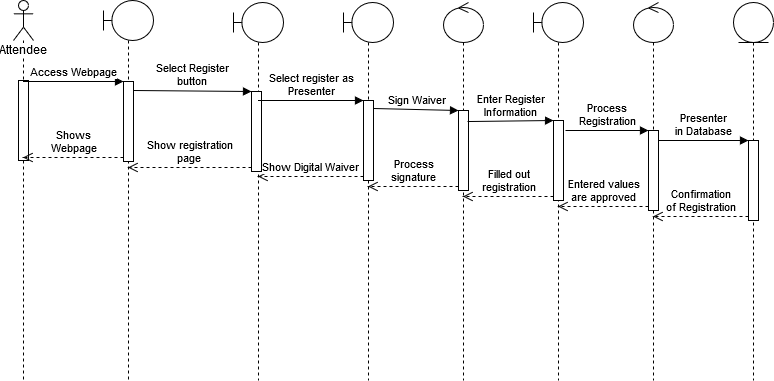
### Register a Technology



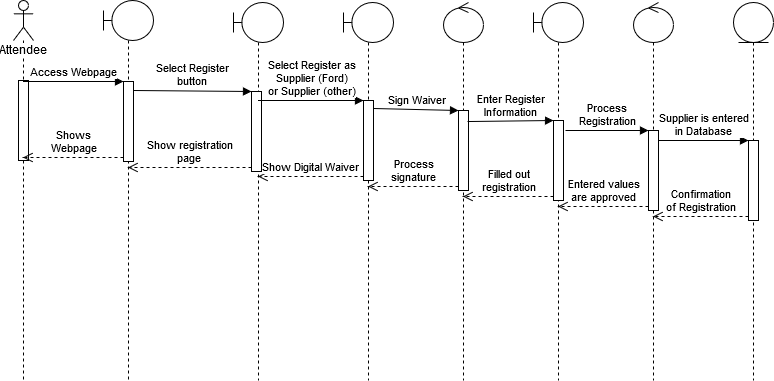
### Pull List of Technologies



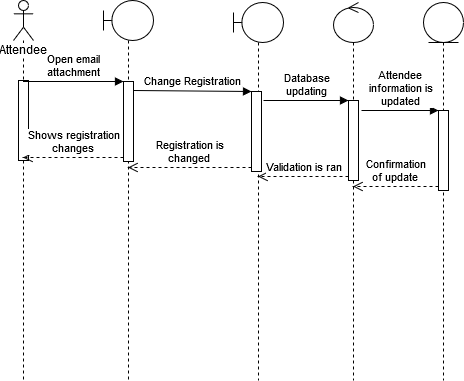
### Register as Evaluator



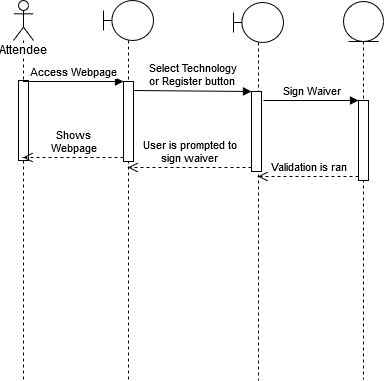
### Register as Supplier or Ford



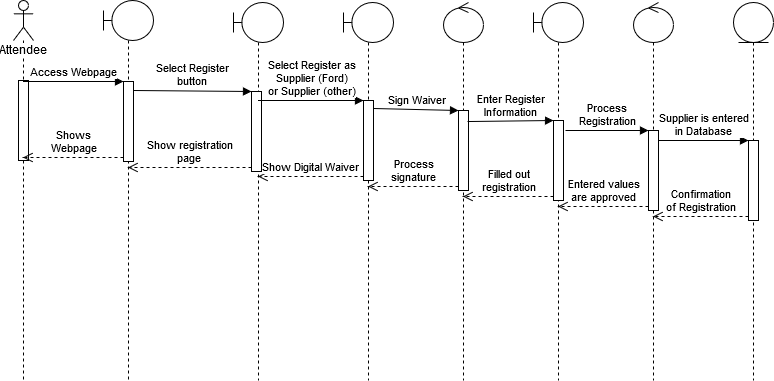
### Modify Registration



### Sign Waiver

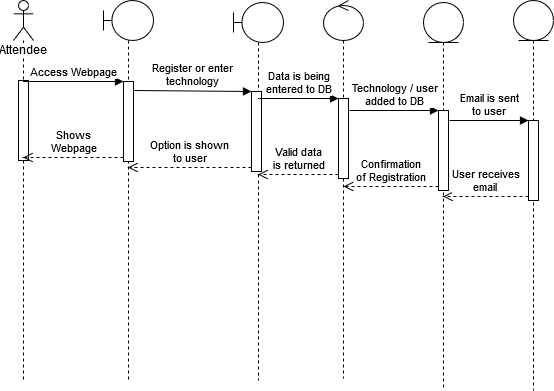


### Confirm Registration

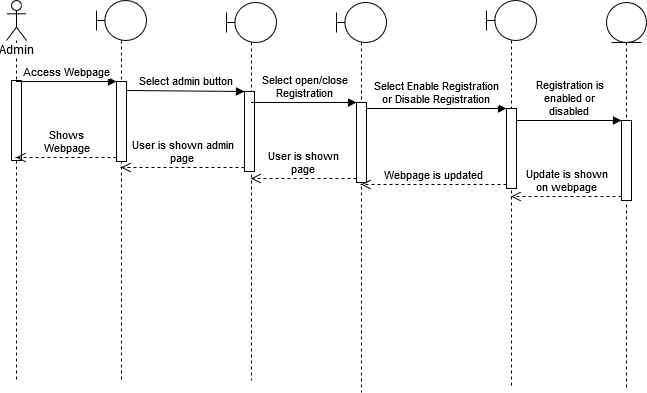


### Print Attendee Badge

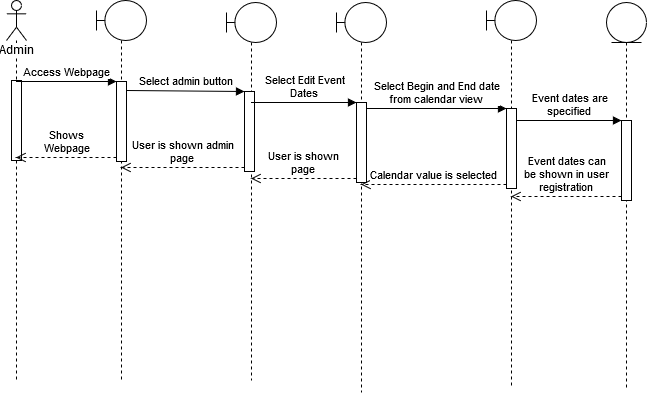
### Send Email



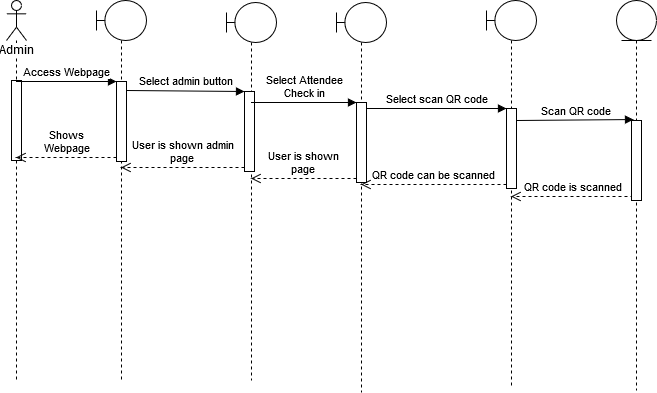
### Open or Close Event Registration



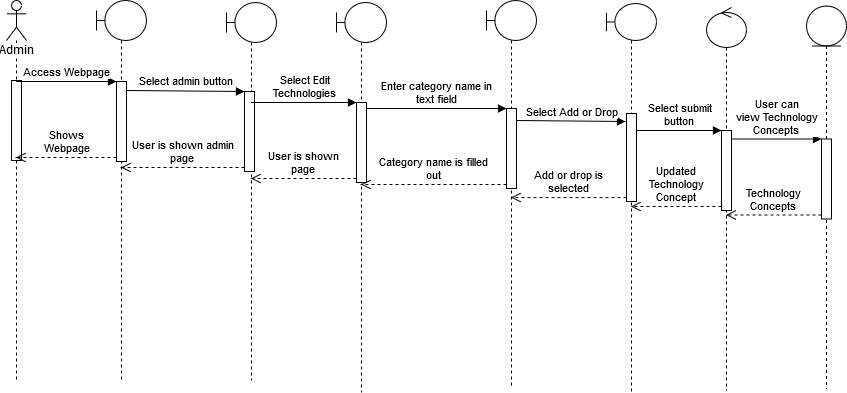
### Add or Remove Event Dates



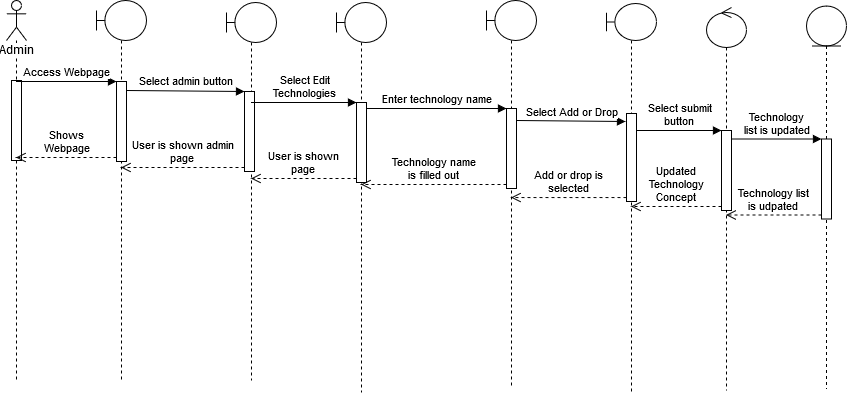
### Check in Attendee



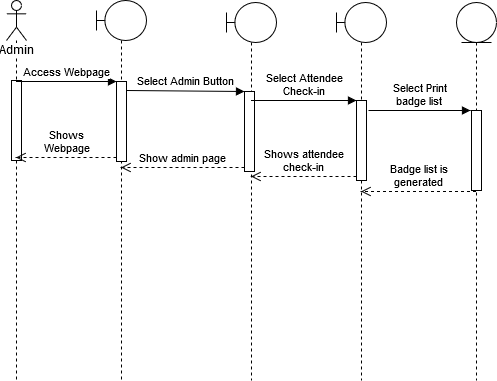
### Add or Remove Technology Category



### Add or Remove Technology



### Generate Badge Sheet



### Generate QR Code

