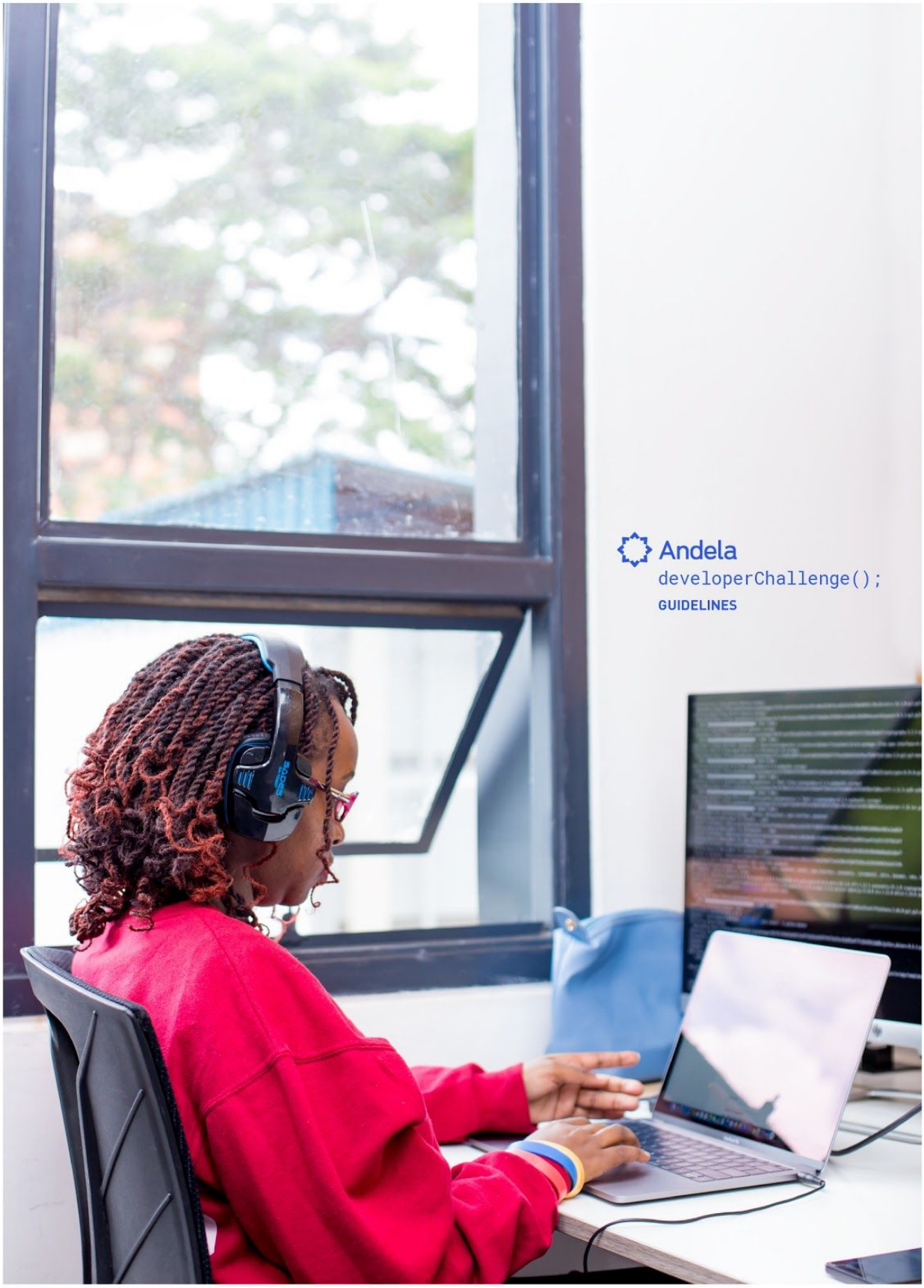
Andela



Andela

developerChallenge();

GUIDELINES

Andela

Andela Developer Challenge

Build A Product: **SendIT**

Andela

BUILD A PRODUCT: SendIT

Project Overview

SendIT is a courier service that helps users deliver parcels to different destinations. SendIT provides courier quotes based on weight categories.

Required Features

1. Users can create an account and log in.
2. Users can create a parcel delivery order.
3. Users can change the destination of a parcel delivery order.
4. Users can cancel a parcel delivery order.
5. Users can see the details of a delivery order.
6. Admin can change the **status** and **present location** of a parcel delivery order.

Optional Features (Extra Credit)

1. The application should display a Google Map with Markers showing the **pickup location** and the **destination**.
2. The application should display computed travel distance and journey duration between the **pickup location** and the **destination.** Leverage Google Maps Distance Matrix Service.
3. The user gets real-time email notification when Admin changes the **status** of their parcel.
4. The user gets real-time email notification when Admin changes the **present location** of their parcel.

NB:

1. The user can only cancel or change the **destination** of a parcel delivery when the parcel's **status** is yet to be marked as **delivered**.
2. Only the user who created the parcel delivery order can cancel the order.

Andela

Preparation Guidelines

These are the steps you ought to take to get ready to start building the project

Steps

1. Create a **Pivotal Tracker Board**

2. Create a **Github Repository, add a README, and clone it to your computer**

   ***Tip**: find how to create a Github Repository here.*

Andela

Challenge 1 - Create UI templates

Timelines

- **Expected Length to Complete: 1 week**
- **Due Date:** Friday, 2nd November 2018

Challenge Summary

You are required to create UI templates with **HTML**, **CSS**, and **Javascript**.

Note:

- *You are not implementing the core functionality yet, you are only building the User Interface elements, pages, and views!*
- *You are to create a pull request to elicit review and feedback for the UI templates when you are done working on them*
- *Do **not** use any CSS frameworks e.g Bootstrap, Materialize, sass/scss.*
- *Do **not** download or use an already built website template.*

Guidelines

1. **On Pivotal Tracker, create user stories to setup the User Interface elements:**
    a. User sign-up and sign-in pages.
    b. A page/pages where a user can do the following:
        i. Create a parcel delivery order.
        ii. Change the destination of a parcel delivery order.
        iii. See the details of a parcel delivery order such as the pickup location, destination, and price.
        iv. Cancel a parcel delivery order.
        v. View all parcel delivery order the individual user has created
    c. A page/pages for a user's profile which, at minimum displays:
        i. The number of parcel delivery order that has been delivered.
        ii. The number of parcel delivery orders that are yet to be delivered (in transit).
        iii. List of all parcel delivery orders.
    d. A page/pages where an Admin can do the following:
        i. Change the **status** of a parcel delivery order.
        ii. Change the **present location** of a parcel delivery order.

2. **On Pivotal Tracker, create stories to capture any other tasks not captured above. A task can be feature, bug or chore for this challenge.**

3. **On a feature branch, create a directory called UI in your local Git repo and build out all the necessary pages specified above and UI elements that will allow the application function into the UI directory**
4. **Host your UI templates on GitHub Pages.**

> ***Tip:*** *It is recommended that you create a* ***gh-pages*** *branch off the branch containing your UI template. When following the GitHub Pages guide, select* ***"Project site" >> "Start from scratch"****. Remember to choose the* ***gh-pages*** *branch as the* ***source*** *when configuring Repository Settings.*

Target skills

After completing this challenge, you should have learned and be able to demonstrate the following skills.

Skill	Description	Helpful Links
Project management	Using a project management tool (Pivotal Tracker) to manage your progress while working on tasks.	• To get started with Pivotal Tracker, use Pivotal Tracker quick start. • Here is a sample template for creating Pivotal Tracker user stories.
Version control with GIT	Using GIT to manage and track changes in your project.	• Use the recommended Git Workflow, Commit Message and Pull Request (PR) standards.
Front-End Development	Using HTML and CSS to create user interfaces.	• See this tutorial • See this tutorial also

Self / Peer Assessment Guidelines

Use this as general guidelines to assess the quality of your work. Peers, mentors, and facilitators should use this to give **feedback** on areas that should be improved on.

Criterion	Does not Meet Expectation	Meets Expectations	Exceed Expectations
Project management	Fails to break down modules into smaller, manageable tasks. Cannot tell the difference between chores, bugs, and features	Breaks down each module into smaller tasks and classifies them. Constantly updates the tool with progress or lack of it	Accurately, assigns points to the tasks. Informs stakeholders of project progress/blockers in a timely manner
Version Control	Does not utilize branching	Utilizes branching,	Adheres

Andela

with Git	but commits to master branch directly instead.	pull-requests, and merges to the develop branch. Use of recommended commit messages.	recommended GIT workflow and uses badges.
Front-End Development	Fails to develop specified HTML/CSS web pages or uses an already built out website template, or output fails to observe valid HTML document structure	Successfully develops HTML/CSS web pages while observing standards such as doctype declaration, proper document structure, no inline CSS in HTML elements, and HTML document has consistent markup	Writes modular CSS that can be reused through markup selectors such as class, id. Understands the concepts and can confidently rearrange divs on request.

Andela

Challenge 2: Create API endpoints

Timelines

- **Expected Length to Complete:** 1 week
- **Due Date:** Friday, 9th November 2018

Challenge Summary

You are expected to create a set of API endpoints already defined below and use data structures to store data in memory. Do NOT use a database.

NB:

- *You are to create a pull request to elicit review and feedback when you are done working on this challenge.*
- *All JavaScript **MUST** be written in **ES6 or higher** and should use **Babel** to transpile down to **ES5**.*
- *Classes/Modules **MUST** respect the **SRP** (Single Responsibility Principle) and **MUST** use the **ES6** methods of module **imports and exports.***

Tools

- Server side Framework: **Node/Express**
- Linting Library: ***ESLint***
- Style Guide: ***Airbnb***
- Testing Framework: ***Mocha*** *or* ***Jasmine***

Guidelines

1. **On Pivotal Tracker, create user stories to set up and test API endpoints that do the following using data structures**
    - Create a parcel delivery order
    - Get all parcel delivery orders
    - Get a specific parcel delivery order
    - Cancel a parcel delivery order
2. **On Pivotal Tracker create stories to capture any other tasks not captured above. The tasks can be feature, bug or chore for this challenge.**
3. **Setup the server side of the application using the specified framework**

4. **Setup linting library and ensure that your work follows the specified style guide requirements**
5. **Setup the test framework**
6. **Version your API using URL versioning starting, with the letter "v". A simple ordinal number would be appropriate and avoid dot notation such as 2.5. An example of this will be: https://somewebapp.com/api/v1/users**
7. **Using separate branches for each feature, create version 1 (v1) of your RESTful API to power front-end pages**
8. **At the minimum, you should have the following API endpoints working:**

EndPoint	Functionality
GET /parcels	Fetch all parcel delivery orders
GET /parcels/<parcelId>	Fetch a specific parcel delivery order
GET /users/<userId>/parcels	Fetch all parcel delivery orders by a specific user
PUT /parcels/<parcelId>/cancel	Cancel the specific parcel delivery order
POST /parcels	Create a parcel delivery order

9. **Write tests for the API endpoints**
10. **Ensure to test all endpoints and see that they work using Postman.**
11. **Integrate TravisCI for Continuous Integration in your repository (with *ReadMe* badge).**
12. **Integrate test coverage reporting (e.g. Coveralls) with a badge in the *ReadMe*.**
13. **Obtain CI badges (e.g. from Code Climate and Coveralls) and add to *ReadMe*.**
14. **Ensure the app gets hosted on Heroku.**

Target skills

After completing this challenge, you should have learned and able to demonstrate the following skills.

Skill	Description	Helpful Links
Project management	Using a project management tool (Pivotal Tracker) to manage your progress while working on tasks.	• To get started with Pivotal Tracker, use Pivotal Tracker quick start. • Here is a sample template for creating Pivotal Tracker user stories.

Andela

Version control with GIT	Using GIT to manage and track changes in your project.	• Use the recommended Git Workflow, Commit Message and Pull Request (PR) standards.
HTTP & Web services	Creating API endpoints that will be consumed using Postman	• Guide to Restful API design • Best Practices for a pragmatic RESTful API
Test-driven development	Writing tests for functions or features.	
Data structures	Implement non-persistent data storage using data structures.	
Continuous Integration	Using tools that automate build and testing when the code is committed to a version control system.	
Holistic Thinking and big-picture thinking	An understanding of the project goals and how it affects end users before starting on the project	

Self / Peer Assessment Guidelines

Use this as general guidelines to assess the quality of your work. Peers, mentors, and facilitators should use this to give **feedback** on areas that should be improved on.

Criterion	Does not Meet Expectation	Meets Expectations	Exceed Expectations
Project management	Fails to break down modules into smaller, manageable tasks. Cannot tell the difference between chores, bugs, and features	Breaks down each module into smaller tasks and classifies them. Constantly updates the tool with progress or lack of it	Accurately, assigns points to the tasks. Informs stakeholders of project progress/blockers in a timely manner
Version Control with Git	Does not utilize branching but commits to master branch directly instead.	Utilizes branching, pull-requests, and merges to the develop branch. Use of	Adheres recommended GIT workflow and uses badges.

		recommended commit messages.	
Programming logic	The code does not work in accordance with the ideas in the problem definition.	The code meets all the requirements listed in the problem definition.	The code handles more cases than specified in the problem definition. The code also incorporates best practices and optimizations.
Test-Driven development	Unable to write tests.	Writes tests that fail.	Writes tests that pass.
HTTP & Web Services	Fails to develop an API that meets the requirements specified	Successfully develops an API that gives access to all the specified endpoints	Handles a wide array of HTTP error codes and the error messages are specific
Continuous Integration	Fails to integrate all required CI tools.	Successfully integrates all tools with relevant badges added to ReadMe.	
Data Structures	Fails to implement CRUD or Implements CRUD with persistence	Implements CRUD without persistence	Uses the most optimal data structure for each operation

Andela

Challenge 3: Create more API endpoints and integrate a database

Timelines

- **Expected Length to Complete:** 1 week
- **Due Date:** Friday, 23rd November 2018

Challenge Summary

You are expected to create all the endpoints required to meet all the requirements listed under the **required features** section and ensure that you persist data with a database**.** You are to write SQL queries that will help you write to and read from your database. The endpoints are to be secured with JWT.

NB:

- *Ensure that Challenge 2 is completed and merged to the **develop** branch before you get started.*
- *You are to create a pull request to elicit review and feedback when you are done working on this challenge.*
- *All Javascript **MUST** be written in **ES6 or higher** and should use **Babel** to transpile down to **ES5***
- *Classes/modules **MUST** respect the **SRP** (Single Responsibility Principle) and **MUST** use the **ES6** methods of module **imports and exports.***
- *Do NOT to use any ORMs.*

Tools

- Database: ***PostgreSQL***

Guidelines

1. **On Pivotal Tracker, create a chore for setting up the database.**
2. **On Pivotal Tracker, create user stories for setting up and testing API endpoints that do the following using database:**
    a. The user can create user accounts and can sign in to the app.
    b. The user can change the destination of a parcel delivery order.
    c. The user can view all parcel delivery orders he/she has created.
    d. Admin can view all parcel delivery orders in the application.
    e. Admin can change the status of a parcel delivery order.
    f. Admin can change the present location of a parcel delivery order

3. **On Pivotal Tracker, create the story(s) for the implementation of token-based authentication using JSON web token (JWT) and the security of all routes using JSON web token.**
4. **On Pivotal Tracker, create stories to capture any other tasks not captured above. The tasks could be feature, bug or chore for this challenge.**
5. **On Pivotal Tracker, create user story(s) to implement one or all out of these optional features:**
    a. The application should display a Google Map with Markers showing the **pickup location** and the **destination**.
    b. The application should display a Google Map with a line connecting both Markers (**pickup location** and the **destination**).
    c. The application should display a Google Map with computed travel distance and journey duration between the **pickup location** and the **destination**.
    d. The user gets real-time email notification when Admin changes the **status** or the **present location** of a parcel.

Note: *Executing the above optional features **after completing the required features** means you have exceeded expectations.*

6. **Setup database.**
7. **Write tests for the endpoints specified below.**
8. **At a minimum, you should have the below-listed API endpoints working**
9. **Test all endpoints with Postman.**
10. **Use API Blueprint, Slate, Apiary or Swagger to document your API. Docs should be accessible via your application's URL.**
11. **Ensure the app gets hosted on Heroku.**

EndPoint	Functionality	Note
POST /auth/signup	Register a user	
POST /auth/login	Login a user	
PUT /parcels/<parcelId>/destination	Change the location of a specific parcel delivery order	Only the user who created the parcel delivery order should be able to change the destination of the parcel.

PUT /parcels/<parcelId>/status	Change the status of a specific parcel delivery order	This endpoint should be accessible by the Admin only.
PUT /parcels/<parcelId>/presentLocation	Change the present location of a specific parcel delivery order	This endpoint should be accessible by the Admin only.

Target skills

After completing this challenge, you should have learned and also be able to demonstrate the following skills.

Skill	Description	Helpful Links
Project management	Using a project management tool (Pivotal Tracker) to manage your progress while working on tasks.	• To get started with Pivotal Tracker, use Pivotal Tracker quick start. • Here is a sample template for creating Pivotal Tracker user stories.
Version control with GIT	Using GIT to manage and track changes in your project.	• Use the recommended Git Workflow, Commit Message and Pull Request (PR) standards.
HTTP & Web services	Creating API endpoints that will be consumed using Postman	• Guide to Restful API design • Best Practices for a pragmatic RESTful API
Test-driven development	Writing tests for functions or features.	
Continuous Integration	Using tools that automate build and testing when the code is committed to a version control system.	
Databases	Using a database to store data.	• Node-postgres • Node.js postgresql tutorial

Self / Peer Assessment Guidelines

Use this as general guidelines to assess the quality of your work. Peers, mentors, and facilitators should use this to give **feedback** on areas that should be improved on.

Andela

Criterion	Does not Meet Expectation	Meets Expectations	Exceed Expectations
Project management	Fails to break down modules into smaller, manageable tasks. Cannot tell the difference between chores, bugs, and features	Breaks down each module into smaller tasks and classifies them. Constantly updates the tool with progress or lack of it	Accurately, assigns points to the tasks. Informs stakeholders of project progress/blockers in a timely manner
Version Control with Git	Does not utilize branching but commits to master branch directly instead.	Utilizes branching, pull-requests, and merges to the develop branch. Use of recommended commit messages.	Adheres recommended GIT workflow and uses badges.
Programming logic	The code does not work in accordance with the ideas in the problem definition.	The code meets all the requirements listed in the problem definition.	The code handles more cases than specified in the problem definition. The code also incorporates best practices and optimizations.
Test-Driven development	Unable to write tests.	Writes tests that fail.	Writes tests that pass.
HTTP & Web Services	Fails to develop an API that meets the requirements specified	Successfully develops an API that gives access to all the specified endpoints	Handles a wide array of HTTP error codes and the error messages are specific
Databases	Unable to create database models for the given project	Has a database design that is normalized and can store, update and query records from the database	Creates table relationships
Token-Based Authentication	Does not use Token-Based authentication	Makes appropriate use of Token-Based authentication and secures all private endpoints.	
Security	Fails to implement authentication and authorization in given project	Successfully implements authentication and authorization in the project	creates custom and descriptive error messages
Test Coverage	The solution did not attempt to use TDD	60% test coverage	> 60% test coverage

Andela

Challenge 4: Implement front-end app

Timelines

- **Expected Length to Complete:** 1 week
- **Due Date:** Friday, 30th November 2018

Challenge Summary

You are expected to power your HTML templates or front-end pages from **Challenge 1** with data from the API built in **Challenge 3.** This challenge requires that you implement the frontend part of the application using vanilla Javascript.

NB:

- *Ensure that Challenge 3 is completed and merged to the **develop** branch before you get started.*
- *You are to make use of **Fetch API** for making requests to the backend*
- ***JQuery can be used only for aesthetics.***
- *Do **not** to use frameworks or libraries like Angular, Vue or React.*

Guidelines

1. **On Pivotal Tracker create stories to build out your frontend with vanilla Javascript.**
2. **On Pivotal Tracker create stories to capture any other tasks not captured above. The tasks can be feature, bug or chore in this challenge**
3. **Create a new folder or repo in which you will develop your front end.**
4. **Setup linting library and ensure you configured the style guide properly**
5. **Implement your front-end.**
6. **Deploy your front-end to Heroku or Github-Pages**

Target skills

After completing this challenge, you should have learned and also be able to demonstrate the following skills.

Skill	Description	Helpful Links
Project management	Using a project management tool (Pivotal Tracker) to manage your progress while working on tasks.	• To get started with Pivotal Tracker, use Pivotal Tracker quick start. • Here is a sample template for creating Pivotal Tracker user stories.

Andela

Version control with GIT	Using GIT to manage and track changes in your project.	• Use the recommended Git Workflow, Commit Message and Pull Request (PR) standards.
UI/UX	Creating good ui interface and user experience	• See rules for good UI design here • See this article for More guide • For color palettes, see this link

Self / Peer Assessment Guidelines

Use this as general guidelines to assess the quality of your work. Peers, mentors, and facilitators should use this to give **feedback** on areas that should be improved on.

Criterion	Does not Meet Expectation	Meets Expectations	Exceed Expectations
Project management	Fails to break down modules into smaller, manageable tasks. Cannot tell the difference between chores, bugs, and features	Breaks down each module into smaller tasks and classifies them. Constantly updates the tool with progress or lack of it	Accurately, assigns points to the tasks. Informs stakeholders of project progress/blockers in a timely manner
Version Control with Git	Does not utilize branching but commits to master branch directly instead.	Utilizes branching, pull-requests, and merges to the develop branch. Use of recommended commit messages.	Adheres recommended GIT workflow and uses badges.
Programming logic	The code does not work in accordance with the ideas in the problem definition.	The code meets all the requirements listed in the problem definition.	The code handles more cases than specified in the problem definition. The code also incorporates best practices and optimizations.
UI/UX	The page is non-responsive, elements are not proportional, the color scheme is not complementary and uses alerts to display user feedback	The page is responsive (at least across mobile, tablet and desktops), the color scheme is complementary, and uses properly designed dialog boxes to give the user feedback	