nodebucket

TECHNICAL DESIGN DOCUMENT

|  |  |
| --- | --- |
| **Author** | Justin Singleton |
| **Last Updated** | 3/30/2020 3:24 PM |
| **Version Number** | 1.0.0 |

**Version History**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Date | Reason for Change | Version |
| Singleton, Justin | 3/3/2020 | Initial Creation | 1.0.0 |
| Singleton, Justin | 3/14/2020 | Add ORD, NoSQL, and Unit Tests | 1.0.0 |
| Singleton, Justin | 3/20/2020 | Add QA Test Plan and Unit Tests | 1.0.0 |
| Singleton, Justin | 3/22/2020 | Add three Unit Tests | 1.0.0 |
| Singleton, Justin | 3/28/2020 | Add three Unit Tests | 1.0.0 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[SECTION 1: INTRODUCTION 3](#_Toc22630702)

[Purpose 3](#_Toc22630703)

[Terminology 3](#_Toc22630704)

[User Personas 3](#_Toc22630705)

[User Stories 3](#_Toc22630706)

[Time Estimations 4](#_Toc22630707)

[SECTION 2: Process Design 4](#_Toc22630708)

[Prototypes 4](#_Toc22630709)

[Object Relational Diagram(s) 4](#_Toc22630710)

[NoSQL Document Diagram 5](#_Toc22630712)

[NoSQL Data Structure 5](#_Toc22630713)

[SECTION 3: QA Testing 5](#_Toc22630711)

[QA Test Plan 5](#_Toc22630714)

[Unit Tests 5](#_Toc22630714)

[SECTION 4: REFERENCES 6](#_Toc22630715)

# SECTION 1: INTRODUCTION

## Purpose

This application is to be a Task Manager for the company to use, to create, edit, delete, and complete tasks. This is for all employees to use and update at their will, so that they may have a tracker for their personal tasks. This will also include a login page to authenticate users before using.

## Terminology

|  |  |
| --- | --- |
| **Name** | **Comments** |
| *MEAN Stack* | *MongoDB, Express, Angular, and Node.js* |
| *VS Code* | *Development IDE* |
| *SoapUI* | *API testing tool* |
| *MongoDB Compass* | *Database development IDE* |
| *Angular Material* | [*https://material.angular.io/*](https://material.angular.io/) |
| *Flex-layout* | [*https://github.com/angular/flex-layout*](https://github.com/angular/flex-layout) |

## User Personas



John Oldman

He is an older gentleman that works as a mid-level employee at the business. His skills with computers are minor at best. He checks his E-mail once a day, but for the most part likes to do his work the ‘old-fashioned’ way. He would more than likely use this application rarely, unless he needs to. That being said, a simpler and intuitive UI would be a necessity for John to respond positively to the application.



John Youngman

He is a younger man who is tasked as the admin for the company’s IT department. He is very knowledgeable in computer technologies. His daily tasks are almost always focused around the computer, and would be the one to oversee the applications processes. He doesn’t care much for UI, but is more interested in the functionality that the admin requires.



Sarah Youngwoman

She is a younger woman that works as a mid-level employee at the company. She is moderately knowledgeable in computers. She checks her E-mail twice a day, and can get around minor troubleshooting issues with her computer. She is always on the go, and is very busy. She requires fast load times and mobile usage for the application, so that she can keep her busy schedule moving.

## User Stories

As an employee, I need a simple and intuitive UI, so that I can find what I need without getting lost.

As an employee, I need to create and update tasks, so that I can know exactly the task that are required.

As an employee, I need to be able to easily find all tasks, so that I can find what I’m looking for, even if I don’t know exactly what I’m looking for.

As an employee, I need an easy place to sign in, so that I can access my account and make tasks.

As an employee, I need to be able to drag and drop items within the manager, so that I can more easily interface with the application.

As an admin, I need to have authorization controls, so that I can access options for the application.

As an admin, I need to have access to user information, so that I can track misuse in the system.

As an admin, I need to have cookies, so that I can track user usages.

As an admin, I need to be able to add/delete users for the application, so that I can keep personnel up to date.

As an admin, I need to give/take away authorization, so that we can have better security.

As an employee, I need to be able to find completed tasks, so that I know what further needs to be done.

As an employee, I need to be able to access the application on mobile, so that I can use it anywhere I’m at.

As an employee, I need to be able to access the application fast, so that I don’t have to slow down my schedule for the slow loading times.

As an employee, I need to have a listing of completed tasks that I have done, so that I can keep track of my progress in case I forget what I’ve done.

As an employee, I need to be able to edit a task, so that if I make mistake or something changes, I can mark that.

**Time Estimations**

|  |  |  |
| --- | --- | --- |
| **Create Login Page** | | |
| **Task** | **Hours** | **Points** |
| **Reactive UI** | **3** | **1** |
| **Implement authentication** | **4** | **2** |
| **Create authorization** | **4** | **3** |

|  |  |  |
| --- | --- | --- |
| **Create Task Manager Page** | | |
| **Task** | **Hours** | **Points** |
| **Reactive UI** | **3** | **1** |
| **Setup Database** | **4** | **2** |
| **Setup Create/Delete** | **6** | **5** |
| **Setup Update** | **4** | **3** |

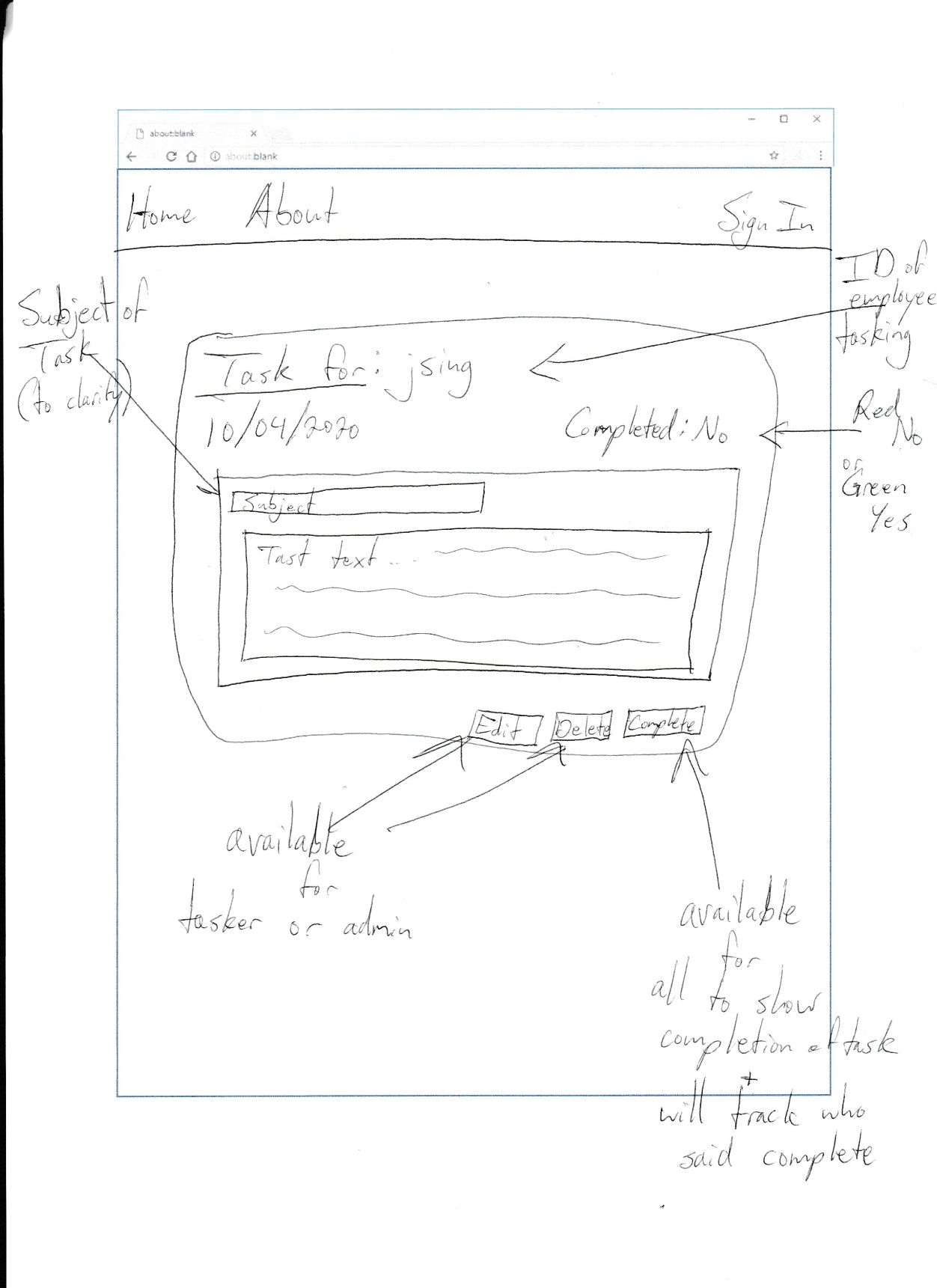
|  |  |  |
| --- | --- | --- |
| **Create Personal Completed List** | | |
| **Task** | **Hours** | **Points** |
| **Reactive UI** | **3** | **1** |
| **Setup Database** | **4** | **2** |
| **Create authorization** | **4** | **3** |

|  |  |  |
| --- | --- | --- |
| **Create Admin Utilities** | | |
| **Task** | **Hours** | **Points** |
| **Implement authorization** | **4** | **2** |
| **Create cookies** | **2** | **1** |

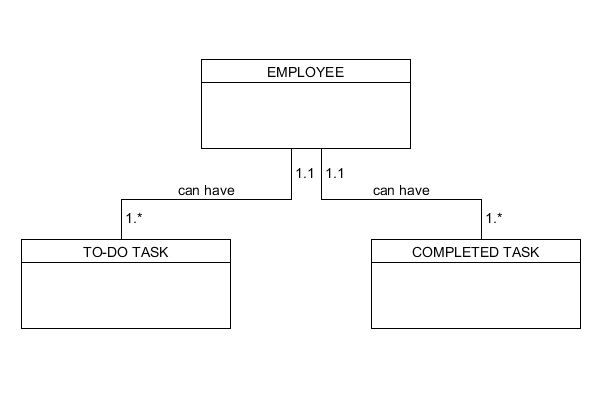
# SECTION 2: Process Design

## Prototypes

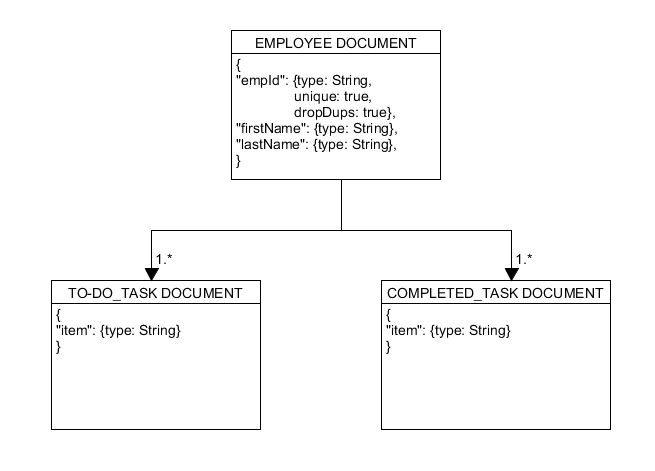
## 



## Object Relational Diagram(s)



## NoSQL Document Diagram



## NoSQL Data Structure



# SECTION 3: QA Testing

## QA Test Plan

**Project Name:** https://github.com/Friend0fMisery/nodebucket

**Test Plan**

|  |  |
| --- | --- |
| **Tests Executed by**: Justin Singleton | **Test Execution date:** March 2020 |
| **Description:** The tests that will be conducted will be small Unit Tests, to make sure that the functionality of the site works as intended. That is to include rendering of all pages, submitting tasks, editing tasks, deleting tasks, signing in, and cookie creation. | |

## Unit Tests

**Test Case Document #1**

|  |  |
| --- | --- |
| **Test Executed by**: Justin Singleton | **Test Execution date:** 3/15/2020 |
| **Description:** This is the test for page rendering. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Test Steps** | **Expected** | **Actual Results** | **(Pass/Fail)** | **Notes** |
| 1 | Navigate to localhost:4200/ | Home page display | Home page display | Pass |  |
| 2 | Navigate to sign-in page | Sign-in page display | Nothing appeared | Fail | Forgot to add to app.module.ts |
| 3 | Navigate to sign-in page | Sign-in page display | Sign-in page display | Pass |  |
| 4 | Navigate to About page | About page display | About page display | Pass |  |
| 5 | Navigate to New task page through button | New task page display | New task page display | Pass |  |

|  |
| --- |
| Comments: This was the initial testing of trying to view the pages in the application. The initial rendering of the home page and the sign-in page worked as intended. Only a small hiccup where the router needed to be updated. After that all other pages displayed upon request through navigation of the UI. |

**Test Case Document #2**

|  |  |
| --- | --- |
| **Test Executed by**: Justin Singleton | **Test Execution date:** 3/15/2020 |
| **Description:** This is the test for the FindEmployeeById API. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Test Steps** | **Expected** | **Actual Results** | **(Pass/Fail)** | **Notes** |
| 1 | Run node app.js | Application starts on port 3000 | Application Starts on port 3000 | Pass |  |
| 2 | Execute GET /api/employees/1007 | Sends ID requested | Continually loads with no end | Fail | Need to add /api to the URI in API |
| 3 | Execute GET /api/employees/1007 | Sends ID requested | 404 | Fail | Logic under API request too complicated. |
| 4 | Execute GET /api/employees/1007 | Sends ID requested | Sends ID requested | Pass | ID requested populates in response. |

|  |
| --- |
| Comments: This was the testing for the API on request of employee ID. There were some issues ran into here. I have little experience building APIs, especially when needing to communicate to the database. After some trial and error of working with the code, the test came back as a pass. The results expected were the employee document showing in SOAPUI, and it did. |

**Test Case Document #3**

|  |  |
| --- | --- |
| **Test Executed by**: Justin Singleton | **Test Execution date:** 3/21/2020 |
| **Description:** This is the test for the FindAllTasks API. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Test Steps** | **Expected** | **Actual Results** | **(Pass/Fail)** | **Notes** |
| 1 | Execute GET /api/employees/1007/tasks | Sends only tasks requested | Sends whole employee document | Fail | Changed logic to try to only get tasks back. |
| 2 | Execute GET /api/employees/1007/tasks | Sends only tasks requested | 404 | Fail | Missing brackets. |
| 3 | Execute GET /api/employees/1007/tasks | Sends only tasks requested | Sends tasks requested | Pass |  |

|  |
| --- |
| Comments: This was the testing for the API on request of all tasks for a specific Employee. Initially I tried recreating the same API as FindEmployeeById with minor changes. This worked a little, but the specification I provided was not good enough. I had to add ‘empId todo done’ within the request to get the information desired back. |

**Test Case Document #4**

|  |  |
| --- | --- |
| **Test Executed by**: Justin Singleton | **Test Execution date:** 3/22/2020 |
| **Description:** This is the test for the CreateTask API. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Test Steps** | **Expected** | **Actual Results** | **(Pass/Fail)** | **Notes** |
| 1 | Execute POST /api/employees/1007/tasks | Creates new task | 404 | Fail | Used .add instead of .push |
| 2 | Execute POST /api/employees/1007/tasks | Creates new task | 404 | Fail | Forgot to change to POST in SOAPUI. |
| 3 | Execute POST /api/employees/1007/tasks | Creates new task | Creates new task | Pass |  |

|  |
| --- |
| Comments: This was the testing for the API on request to create new task. This was the easiest of them all, since the logic from the previous APIs helped a ton. The mistake of not using .push was minor. The SOAPUI mistake was also easily fixable. After this, you can now create new tasks with ease. |

**Test Case Document #5**

|  |  |
| --- | --- |
| **Test Executed by**: Justin Singleton | **Test Execution date:** 3/22/2020 |
| **Description:** This is the test for the UpdateTask API. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Test Steps** | **Expected** | **Actual Results** | **(Pass/Fail)** | **Notes** |
| 1 | Execute PUT /api/employees/1007/tasks | Edits task | Continuously loads | Fail | I did not set the employee todo and done objects. |
| 2 | Execute PUT /api/employees/1007/tasks | Edits task | 404 | Fail | I had employee.todo.push still in the code, which gave issues. Had to remove. |
| 3 | Execute PUT /api/employees/1007/tasks | Edits task | Edits task | Pass | It edited but deleted the rest of the array. Had to change the logic to be able to identify the task being edited. |
| 4 | Execute PUT /api/employees/1007/tasks/:taskId | Edits task without deleting | Edits task without deleting | Pass |  |

|  |
| --- |
| Comments: This was the testing for the API on request to edit a task. Using the logic from the previous API, made this easier. There were a couple hiccups in identifying certain areas of the code, such as todo and done. The test ran as expected, however the entire todo array was deleted for the updated item. Therefore with identifying the specific item, allowed us to only change that specific item. |

**Test Case Document #6**

|  |  |
| --- | --- |
| **Test Executed by**: Justin Singleton | **Test Execution date:** 3/22/2020 |
| **Description:** This is the test for the DeleteTask API. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Test Steps** | **Expected** | **Actual Results** | **(Pass/Fail)** | **Notes** |
| 1 | Execute DELETE /api/employees/1007/tasks/:taskId | Deletes specific task. | 404 | Fail | .toString() was not recognized, had to change the code there. |
| 2 | Execute DELETE /api/employees/1007/tasks/:taskId | Deletes specific task. | Deletes specific task | Pass |  |

|  |
| --- |
| Comments: This was the testing for the API on request to delete task. This was the simplest test of the APIs. With the code from Update task available, I was able to just make minor tweaks. The .toString() method did not work so I had to make minor adjustments there. After, I was able to target a task and delete without affecting the rest of the array. |

**Test Case Document #7**

|  |  |
| --- | --- |
| **Test Executed by**: Justin Singleton | **Test Execution date:** 3/28/2020 |
| **Description:** This is the test for the Sign-in cookie store. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Test Steps** | **Expected** | **Actual Results** | **(Pass/Fail)** | **Notes** |
| 1 | Input employee ID and submit | Populates cookie with ID as value. | Blank cookie | Fail | Cookie store value was not set to input field. |
| 2 | Input employee ID and submit | Populates cookie with ID as value. | Populates cookie with ID as value. | Pass |  |
| 3 | Input employee ID and submit | Reroutes to Home with employees list of tasks | Reroutes to home, no tasks. | Fail | [] brackets were supposed to be (), |
| 4 | Input employee ID and submit | Reroutes to Home with employees list of tasks | Reroutes to Home with employees list of tasks | Pass |  |

|  |
| --- |
| Comments: This was the testing for the cookie usage in the sign-in page, to reroute to home, and to populate the tasks of the ID specified. Cookies were being set initially, but with no value. Had to connect the input field to the cookie population. After that, rerouting to home was easy, but finding the mistake of the [] was extremely difficult since there was no real way of knowing. After some help from a class mate, the mistake was fixed. |

**Test Case Document #8**

|  |  |
| --- | --- |
| **Test Executed by**: Justin Singleton | **Test Execution date:** 3/28/2020 |
| **Description:** This is the test for the Create Task Dialog. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Test Steps** | **Expected** | **Actual Results** | **(Pass/Fail)** | **Notes** |
| 1 | Click New Button | Opens Dialog. | Opens Dialog | Pass |  |
| 2 | Input Field to create new task. | Closes Dialog and new task appears. | Closes Dialog, no new task. | Fail | Error connected with dependencies not being updated. |
| 3 | Input Field to create new task. | Closes Dialog and new task appears. | Closes Dialog and new task appears. | Pass |  |

|  |
| --- |
| Comments: This was the testing for the Create Task Dialog, if it opens, closes, and creates the task specified. Opening was no problem, and so was closing. Though, there were some issues in troubleshooting why the task wasn’t being populated. The error given was vague, but after some research, I just needed to update my dependencies. After that, the new task was created. |

**Test Case Document #9**

|  |  |
| --- | --- |
| **Test Executed by**: Justin Singleton | **Test Execution date:** 3/28/2020 |
| **Description:** This is the test for the Drag and Drop functionality. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Test Steps** | **Expected** | **Actual Results** | **(Pass/Fail)** | **Notes** |
| 1 | Drag task to new column. | New task is moved. | New task is not moved, new blank task is created. | Fail | The update function was not routed correctly to the API. |
| 2 | Drag task to new column | New task is moved. | New task is moved. | Pass |  |

|  |
| --- |
| Comments: This was the testing for the Drag and Drop functionality. The ability to pick up an item was working correctly but when trying to drop, the task would shift back to where it was at, and create a blank item in the todo field. This was because the Update function wasn’t routed correctly, with :taskId at the end. After removing this, the drag and drop worked back and forth between columns. |

# SECTION 4: REFERENCES

Depositphotos, Inc. (n.d.). ᐈ Cartoon oops stock images, Royalty Free oops cartoon pictures

vectors: download on Depositphotos®. Retrieved March 25, 2020, from

https://depositphotos.com/vector-images/cartoon-oops.html