**A.   Write a comprehensive summary that addresses the following requirements:**

|  |
| --- |
| **The business problem or opportunity you are solving for, including a description of the customers and why this application will fulfill their needs** |
| Not a Real Company (NaRC) is currently looking for a scalable solution for managing interactions with their customers. NaRC provides SEO services to local businesses looking to improve their online presence. The customers that are currently using NaRC’s services provides NaRC with content regarding their local business. This requires frequent interaction between NaRC and their customers.  This application fulfills NaRC’s needs by allowing their SEO Specialists keep track of their customers, create appointments with reminders, log interactions with customers, as well as log/create work cases related to customers. NaRC’s SEO specialists will have individual accounts and be able to hold more accountability when working with customers and provide deliverables to customers with work and communication logs to give them piece of mind or to settle disputes when/if customers claim no work is done or no communication was made. |
| **Existing gaps in the software application you are replacing or modifying (if applicable)** |
| NaRC currently uses an appointment scheduler that requires a wider range of function. Implementing the ability to log communication and work cases in the current software application being used will satisfy their additional needs. |
| **The software development life cycle methodology you use to guide and support software development activities** |
| The Waterfall Agile hybrid SDLC methodology will be used to guide and support development activities throughout the project. This is to decrease the number of bugs introduced into the application going through incremental modifications whilst taking customer input into consideration.  First is analyzing the problem the client has presented. Understanding this assist with planning a solution. Next is gathering requirements for the application. High-level requirements are already provided by NaRC. The primary actors that will be working directly with the application will provide user stories alongside developers. The user stories will then be translated to functional and non-functional requirements. We can then begin story boarding to get an idea of how the application will work. We’ll start putting together a document for unit tests while designing the application interface referencing back to the requirements document. Users will give their input on the design to ensure that the user interface is simple and intuitive. After the design phase, we can continue building on the document for unit tests for another day. Now the coding development process will begin in the form of sprints to quickly build, test, debug, test, and present to the user for acceptance testing. Once the application is complete and fulfills NaRC’s needs, all deliverables will be packaged prior to delivery to the customer. Maintenance for the application is not included in the contract. |
| **Deliverables associated with the applied software development life cycle methodology** |
| Deliverables associated with the applied SDLC include a brief on the purpose of the application, functional requirements, and use cases. |
| **The plan for implementation of your software solution, including the anticipated outcomes from this development** |
| Implementing this software solution requires a migration of the current database that NaRC is using to track appointments and customer information to the new application. Additional tables will be created within the database to accommodate the functions in the new application.  The expected outcome is a seamless transition from the old application to the new application. Allowing immediate use of the application for logging communication and work with customers. |
| **The methods for validating and verifying that the developed software application meets the requirements and subsequently the needs of the customers** |
| Black box testing methods will be used to validate and verify the software application to meet the requirements/needs of the customer prior to sending the application to the customer for acceptance testing. This includes boundary value analysis, equivalent partitioning, and cause-effect graphing. |
| **The programming environments and any related costs, as well as the human resources that are necessary to execute each task in the development of the software application** |
| Developing this software application requires:   * IntelliJ IDE - $0.00 * JDK 20 - $0.00 * JavaFX - $0.00 * MySQL - $0.00 * Computer - $0.00 (already supplied) * Cloud server - $0.00 (temporary trial service) |
| **A projected timeline including milestones, start and end dates, duration for each milestone, dependencies, and resources assigned to each task** |
| i. Planning and Design: 10  ii. Development: 40  iii. Documentation: 5 |

**B.   Design and develop a fully functional software application that addresses your identified business problem or organizational need. Include each of the following attributes as they are the *minimum required elements* for the application:**

|  |
| --- |
| **one of the following application types: mobile, web, or stand-alone application** |
| Stand-Alone Application |
| **code including inheritance, polymorphism, and encapsulation** |
| Factory design pattern for customer interactions, persons, and locations. |
| **search functionality with multiple row results and displays** |
| Ability to search customers through customer names. |
| **a database component with the functionality to securely add, modify, and delete the data** |
| JDBC MySQL with all functions withing the DAO package |
| **ability to generate reports with multiple columns, multiple rows, date-time stamp, and title** |
| Reports are generated and viewed within the Reports screen. |
| **exception controls** |
| Multiple IO, SQL, and general exceptions throughout the application. |
| **validation functionality** |
| Form validation for appointments, customers, and interactions. |
| **industry appropriate security features** |
| User verification on the Login screen. |
| **design elements that make the application scalable** |
| Factory design method for people, interactions, and locations. Allows the creation of different variants of people, interactions, and locations. |
| **a user-friendly, functional GUI** |
| Designed with end user in mind. |

**C.   Create *each* of the following forms of documentation for the application you have developed:**

|  |
| --- |
| **A design document including a class diagram and design diagram** |
| Design: ApplicationDesign.docx file in ProjectDocument folder  Class: CapstoneCRM\_UML\_Diagram.png image file in ProjectDocument folder |
| **A test plan for a unit test, including screenshots** |
| **Requirement to be tested:**  Customer search function within the Customers window.  **Preconditions: Conditions that must be present before test case can successfully run**  The customers table must be populated with customers.  **Steps: The steps the tester must execute to test the feature.**   1. Login to the application from the login screen using the existing username John and password nguyen. 2. Once on the Appointments screen navigate to the bottom right corner and click on the customer’s button. 3. Once on the Customers screen, there with be a prepopulated list of customers within the customers’ table. 4. Within the top right corner of the screen there with be a search box and button for searching customers by name. 5. Enter a name within the list of customers then click the search button.   **Expected results: Expected results and any side effects such as updating a database, writing to a file, etc.**  The expected result is the customers table only displaying the customers that have a matching name. |
| **The results of the unit test based on the provided test plan, including screenshots** |
| Test Pass: |
| **Source code and executable file** |
| Available in the CapstoneCRM folder \C868\CapstoneCRM\src |
| **User guide for setting up and running the application for maintenance purposes** |
| Author:  Name - John Nguyen  Email - jngu102@wgu.edu  Phone - (425) 770 - 9705  Title: Appointment Tracker  Purpose: A scheduling desktop application for keeping track of appointments with customer  IDE: IntelliJ IDEA 2022.2.1  JDK: GraalVM 20.0.1  JavaFX: javafx-fxml 19.0.2.1 set as a dependency from Maven2 repository  javafx-controls 19.0.2.1 set as a dependency from Maven2 repository  javafx-maven-plugin 0.0.8 set as a plugin from Maven2 repository  Direction To Run: To run the program, you must first set up the RDBS with MySQL. Then open the project within an IDE, build the project, then compile the program.  Additional Report: The additional report displays a record of all login attempts in the event suspicious activity occurs  MySQL Connector Driver version 8.0.33 - mysql-connector-j-8.0.33 |
| **User guide for running the application from a user perspective** |
| User\_Guide.docx in ProjectDocument folder |