**Capstone Project Document**

This document outline shows you the information required for the capstone document.

1. Use this as a template for your own document.
2. Do not remove or re-order the sections listed below.
3. If you do not have content for a specific section, i.e., “current system”, then write a sentence in that section stating so.
4. Remove any comments or explanation (Red or Blue color) in this template before submitting your document
5. Explanations of the sections are in Blue color and should be removed from the document
6. This page should be removed from the submitted document. Your first page in the document will be the next page
7. Bullets are good to use in your document when you are listing ideas

**Title Page**

* *name of system, group members, date, client(s), instructor(s)*

High Five Star

Members – Stephen Church, Robert DeVries, Scott Davis, Benjamin Barnes, Richard Hansel

08 Jan 2022

Instructor – Ali Moussa

## Table of Contents

* *including figures, tables, and appendices*

<https://support.microsoft.com/en-us/office/insert-a-table-of-contents-882e8564-0edb-435e-84b5-1d8552ccf0c0>

**Example:** Right click on the example, then select “Update Field” to update it

Contents

[Table of Contents 3](#_Toc88352508)

[Preface 4](#_Toc88352509)

[Purpose of Document 4](#_Toc88352510)

[Documentation Standards 4](#_Toc88352511)

[User Requirements 5](#_Toc88352512)

[Project Overview 5](#_Toc88352513)

[Statement of the Problem 5](#_Toc88352514)

[Non-Functional Requirements 6](#_Toc88352515)

[System Interface Requirements 6](#_Toc88352516)

[Maintainability and Administration Requirements 6](#_Toc88352517)

[Usability Requirements 6](#_Toc88352518)

[System Design 7](#_Toc88352519)

[Project Management 9](#_Toc88352520)

[Team Configuration 9](#_Toc88352521)

[Project Standards and Procedures 9](#_Toc88352522)

[Glossary 10](#_Toc88352523)

[Index 11](#_Toc88352524)

## Preface

## Purpose of Document

* *What is this document about, what information does it contain that a reader might be interested in?*
* This document is to provide information and reference to “High Five Star” web application development procedure. Found in the following sections are purposes of developing the application and business requirements that it will fill. It contains Use case and design diagrams as well as system requirements. Application architecture are illustrated as well as hardware and software requirements for development and deployment. Finally, project timelines are described in detail.

## Documentation Standards

* *Special notations used in this document, e.g. use of fonts and colors*

## User Requirements

**Business Overview and Objectives**

* *Nature of the client’s business, its mission*
* (Assumption): Client has not been identified, hence recommend a fictitious client Name and potential mission statement, therefore:
  + Client Name: FiveStar
  + Mission: Our mission is to provide the best solution with the intent to surpass customer/service provider expectations in achieving mutual goals!

## Project Overview

## Statement of the Problem

* There currently is no easy method of finding the best quality of workers or companies that provide a reputable service and hire them.

**Project Scope**

* Provide web application that allows customers to search for services.
* Provide a means to compare different service providers.
* Assure the quality of work and payments for both sides.
* Allow service providers to post relevant information about their services and pricing.
* Ability to generate a contract between customer and service provider in order to ensure quality of work and agreed pricing is met.
* Provide a reward system in the form of discounts for high volume users.
* Application fee for service providers with monthly fees and percentage of commission from business obtained from application with a 3-tier account system for service providers: Gold, silver, and bronze. Higher account tiers will show first in searches.

**System Environment**

* The application will be a web-based application providing an easy-to-use interface for both customer, to seek out services, and service provider to advertise and sell services through specific individual, verified accounts using Two-Factor Authentication.

**Current System**

* Current existing systems include web browser searching, word of mouth through personal connections and currently knowing a service provider.

**System Requirements**

**Use Case Diagram**

###### Diagram Description automatically generated

###### 

###### **Use Case Descriptions and Scenarios**

|  |  |  |
| --- | --- | --- |
| Actor | Real Use Cases | Extended Use Cases |
| **Customer –** Any citizen who signs up as a user of the application and that is searching for specific services to hire a provider of said service. | **Login** – Customer will log into app via HTML form with user name and password. App will Authenticate and log into Customer interface  **Search for Services –** Customer will have ability to Enter service and app will perform filtered search for Service provider profiles.  **Receive Incentive Discounts for High Usage -**  Incentive discounts will be awarded for users who hire more service providers through app. | **Contact Service Provider-** will have option and means to contact service provider to initiate negotiations  **Negotiate –** use communication system to negotiate services  **Sign contract –** parties will have means to send electronic copy of service contract and add digital signature  **Submit Review of Service Provider –** customer will have option to submit review of Service provider upon contract completion. |
| **Service Provider –** Any worker or business owner that provides a service and wanting to advertise and sell their services on an efficient platform. | **Login** – Service provider will log into app via HTML form with username and password. App will Authenticate and will log into service provider interface  **Create Profile –** will have means to create profile with description of services, bio  **Choose Tier Level –** choose tier level to increase visibility in searches. Gold, Silver and Bronze | **Receive communications from customers –** means to receive messages from customers to provide more information.  **Negotiate –** use communication system to negotiate services  **Sign contract –** parties will have means to send electronic copy of service contract and add digital signature |

## Functional Requirements

* Clean, easy to use Web application interface.
* Allow users to create a profile as either a customer or a service provider
* One log in for all users but depending on profile status will direct user to appropriate interface (customer or service provider)
* Easy to use search option for customers to search for services
* AI virtual assistant (chatbot) to aid in customer user experience
* Direct messaging feature to message service provider for questions and to negotiate for services.
* Customer usage should be tracked to award discount incentives
* Service providers sign up should allow user to indicate tier level. Tier level upgrade at any time will be available. Tier level will dictate visibility.
* Customers will have ability to submit reviews of service providers if they have hired provider
* Service providers will be able to post description of their services and pictures of their work for advertising purposes
* Easy to use username and password reset should be implemented.

## Non-Functional Requirements

* User log in passwords will be stored in database using hash algorithm
* All HTML forms will be sanitized for exploitations including SQL injection and command injection attacks. Use known good libraries within development frameworks if necessary.
* Tier level of service providers will be a field in service provider table in data base. Search results will be sorted by tier level in the query.
* Reviews submitted by customers will be stored in own table with relationship to the service provider and review can be seen when viewing service provider profile in application. Reviews will be sorted by date submitted.
* Customer incentive discounts will be based on usage **(# of successful hires, not log ins!)** usage will be tracked in database table and incentive field will be calculated based on usage field.

## System Interface Requirements

* Use pre-built A.I virtual assistant to be embedded in application (to be chosen)
* Payment gateway service API (Paypal or similar) to handle transactions between customers and service providers

## Maintainability and Administration Requirements

* Payment methods will have to be monitored and modified on ad-hoc bases to meet PCI requirements, user/industry standards etc.
* U.I will be updated to continue to provide modern user experience based on current industry standards. A modern front-end framework with little chance of deprecation in near future should be chosen to increase lifespan of application without having to perform a complete re-write of codebase.
* Will have administrator log in to handle false profiles or misuse of platform.
* Easy to use username and password reset should be implemented.

## Usability Requirements

* Expected to have varying levels of experience of users
* Focus will be to have a clean and intuitive user interface. Styles and fonts should be easy to see. All functions should be accessible with minimal “clicks”
* For inexperienced users, the AI chatbot should be very intuitive and easy to use to aid in use of the platform.

**Interaction Sequence Diagrams**

* *Create diagrams for* ***at least four*** *non-trivial use cases*
  + *Show problem domain objects only in these diagrams: assume they can communicate directly with actors and are persistent (can store and retrieve data, e.g. in a database)*

**State Machine Diagrams**

* *Identify and diagram at least one complex time-dependent object behavior in your system*

**Activity Diagrams**

* *Identify and diagram at least one complex business activity or object that exhibits complex processing*

## System Design

Layered Architecture

* *Package diagram (do not show classes in the packages, only packages)*

Hardware Architecture

* *textual description*
* *Give an overview of the different computers, etc. that your system will need to function and how they will be related (in UML we use a* deployment diagram *to show this)*
* *How will the different parts of your system will communicate with each other? e.g. TCP/IP, HTTP, XML, JSON, JDBC etc.*

**Hardware Platform**

* *required hardware specs to develop and maintain, e.g. your laptop specs*
* *required hardware specs for* ***production*** *system, e.g. the server(s) the system will eventually be deployed and run on (not your development machine)*

Software Platform

* *required software specs to develop and maintain*
* *required software specs for* ***production*** *system*

Interaction Model

* *style (dialog, direct manipulation)*
* *desired user support (required help, error messages, etc.)*
* *system feedback style (what events require informing the end-user of activity, etc.)*
* *standards (colours, fonts, etc.)*
* *example of each* ***type*** *of interaction (e.g., one data-entry screen, one list-of-items screen, one error warning screen, one summary report, one detail report, etc.). These should be* ***high-quality*** *screenshots or mock-ups; they need to give a clear indication of your proposed UI style*
* *a list of every screen and sub-screen descriptive title*

Persistence Model

* *textual description of mechanism, e.g. you are using JPA, stored procedures, etc.*
* ***Conceptual*** *Schema ERD (as per Systems Analysis and Design course)*
* ***Internal*** *Schema ERD (3NF physical model as per Database courses)* ***and/or*** *file structures*
* *Calculation of expected data size (data size for each persistent class \* expected max number of stored objects of each class)*

Class Diagrams

* *one class diagram* ***per package*** *(see “****Layered Architecture****” section above)*
* *define the attributes, associations and operations for each class in your class diagrams using the full UML notations, e.g. including data types, full operation signatures for operations, etc. UI classes can be complex in structure so you do not need to fully define these*

Interaction Sequence Diagrams

* + ***One diagram per use case in the use case diagram!***

## Project Management

**Schedule**

###### major milestones (Microsoft project is useful here)

## Team Configuration

###### members, roles, reporting relationships, contact information

## Project Standards and Procedures

* *Procedures involve how your development team is going to work together on the project, e.g. regular meeting times, handling communication between team members, etc. Tools and standards might include things like development language, database to use, frameworks to use, industry standards to be followed, etc. Update as they are chosen (this is evolutionary)*

## Glossary

*[Place on a separate page of its own]*

###### definitions of **technical** terms used in this document. Some common terms might have project-specific meaning and so should be described here also, e.g. “manager” has many meanings, but if it means something specific in your project describe it to avoid ambiguity

## Index

*[Place on a separate page of its own]*

###### more than a reformatted table of contents

**Appendix A: Data Dictionary**

*[Place on a separate page of its own]*

*A data dictionary should describe each of the classes in your system, including its attributes, operations, associations, etc. It should include things like valid ranges for attributes, e.g. must not be null, must be between 1 and 10, etc., and a general description of what operations are intended to do. It is intended to be a resource that a developer can go to during project implementation if s/he needs more information on what a specific class is supposed to do in the application (and how), i.e. it’s supposed to reduce ambiguity in the design.*

*There is no standard format for a data dictionary so we recommend using one of two approaches:*

* *Use JavaDoc to create formatted descriptions of your classes, and their contents, e.g. describe the responsibilities of each class, what it’s attributes are (including things like data types, valid ranges for values, etc.), what operations each one will have and roughly what each operation should do. An advantage of using JavaDoc is that it is easy to auto-generate the data dictionary content periodically as the project progresses*
* *Create a table in Word with appropriate columns and enter the data dictionary information into that. This is simple but the content cannot be auto-generated and must be manually update*

**This page includes some links and useful resources for you document understanding. Please remove this page from the submission. It was added to help you.**

[**https://learning.oreilly.com/library/view/software-engineering/9789332558298/xhtml/Chapter002.xhtml#h5-008**](https://learning.oreilly.com/library/view/software-engineering/9789332558298/xhtml/Chapter002.xhtml#h5-008)

[**https://learning.oreilly.com/library/view/software-engineering/9789332558298/xhtml/Chapter006.xhtml#h5-001**](https://learning.oreilly.com/library/view/software-engineering/9789332558298/xhtml/Chapter006.xhtml#h5-001)