|  |  |
| --- | --- |
| **Assessment 2 – Software Design Specifications** | |
| **Project Name** | Driving License Management System |
| **Group #** | 1 |
| **Group Members Names** | Carlos Armando Escalona Montes de Oca  Jayawarne Arachchige Dewmi Nihara Sakunali  Srijana Aryal  Sudarshan Udash |
| **Version** | 1.0 |
| **Date Created** | November 11th, 2022 |

Contents

[1. Introduction](#_heading=h.gjdgxs) **3**

[1.1 Purpose of Document](#_heading=h.30j0zll) 3

[1.2 Scope of the Project](#_heading=h.1fob9te) 3

[1.3 System Overview](#_heading=h.3znysh7) 3

[2. Design Considerations](#_heading=h.2et92p0) **3**

[2.1 Assumptions and Dependencies](#_heading=h.tyjcwt) 3

[2.2 General Constraints](#_heading=h.3dy6vkm) 4

[3. Use Interface Design](#_heading=h.1t3h5sf) **4**

[4. Architectural Strategies](#_heading=h.4d34og8) **4**

[5. System Architecture](#_heading=h.2s8eyo1) **5**

[6. Detailed System Design](#_heading=h.17dp8vu) **5**

[7. Glossary](#_heading=h.3rdcrjn) **6**

[8. References](#_heading=h.26in1rg) **6**

# Introduction

## Purpose of Document

This Software Design Specification Document will define the design of a Driver’s License Management System. It contains specific information about the architecture, expected inputs, outputs and interaction between classes and functions to meet the desired requirements.

## Scope of the Project

The scope of the project is to develop a Driving License Management System to allow users to schedule dates for their license trials.

## System Overview

This system aims to make it easier for public entities and applicants for a driver's license to obtain an appointment for their assessment/trial through a simple and user-friendly website that contains their personal details and allows them to select a date for their assessment.

It will have a GUI supported for the most common browsers and the user will be able to register and login to the system to apply. All this information is stored in a database. The administrator’s part of the public entity can reject or approve the evaluation requests after a manual review of the requirements and record the results to be displayed in the system.

# Design Considerations

This section describes many of the issues which need to be addressed or resolved before attempting to devise a complete design solution.

## Assumptions and Dependencies

Assumptions and dependencies for the license management system are listed below:

* Only booking and processing functionality will be developed on the system.
* Applicants can request booking only after registration and proper login.
* Each applicant is unique in the system.
* Each applicant has only one trial appointment in the system.
* Evaluation of application is manually set by the staff
* Compatible web browser is used.
* Payment is not processed on the system.

## General Constraints

A list of a general constraints are listed below:

* Limited time for the development.
* Communication limitations due to geographical distribution of members.
* Lack of programming skills on the team.
* System will use the Django framework as a web server.
* The Django framework will be used in conjunction with Python, Jinja2, CSS and HTML languages.
* Versioning control will use Git on a public repository (GitHub).
* Database to store information will be SQLite. Which is not a large-scale intended database but works as a base for the prototype.

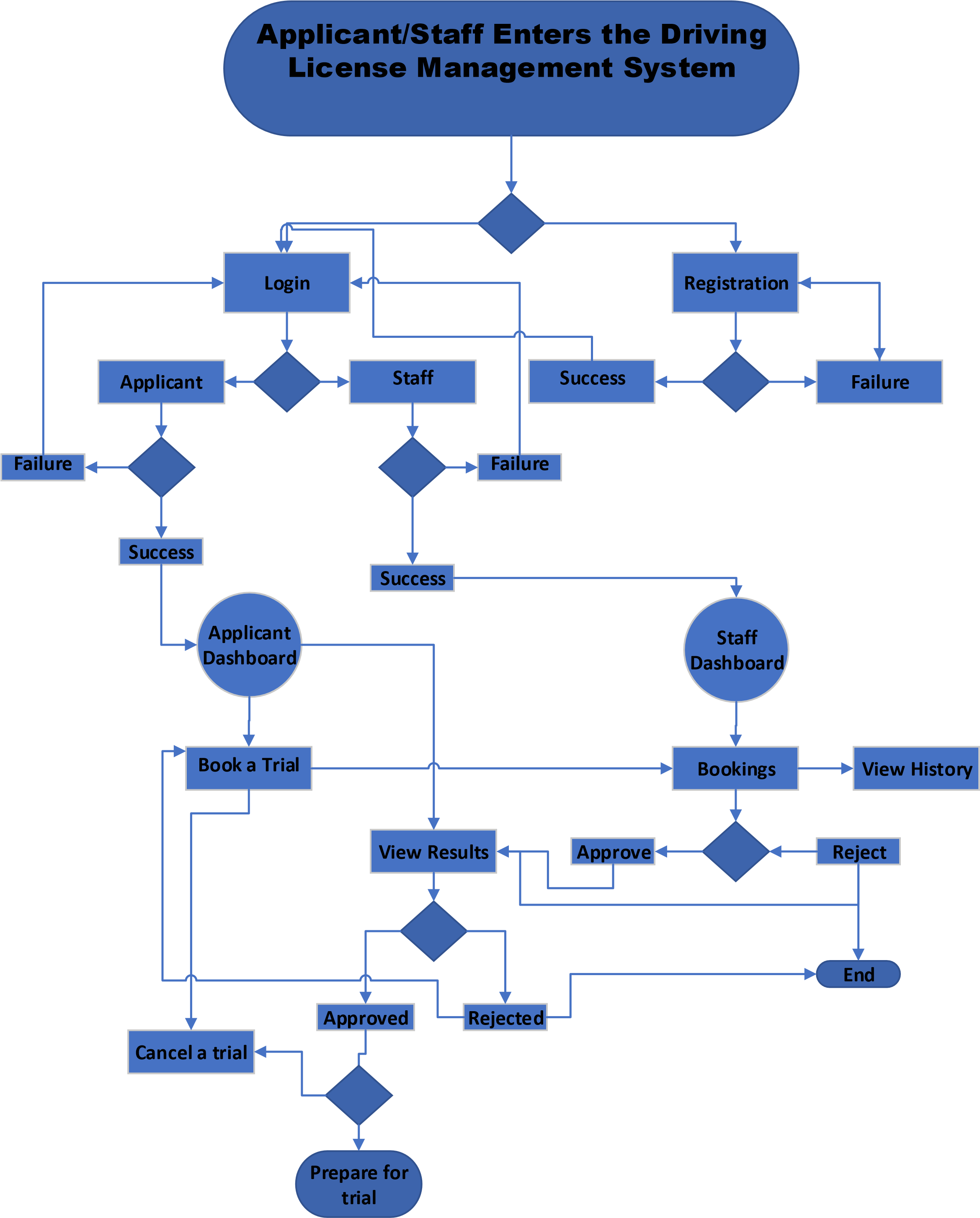
# Use Interface Design

HTML web page with buttons and users form to introduce information with the logo and official identification of the public entity.

* Personas: System is designed with two types of users in design Applicant and Staff.
  + - * **Applicant:** Male or Female. +18 years old. Working or studying any profession. Goal is to get a driver's license. Expecting the system to give them instruction and guide them through the process of getting the appointment.
      * **Staff:** Male or Female. +25 years old. Working for the public entities. Responsible for reviewing, approving or rejecting applications and recording the results in the system. Expecting the system to be a useful tool to simplify their jobs.
    - User Scenario:

George is a 18 years old student that just received a car for his birthday. He wants to use this car but first he has to get his driver's license. His parents guide them in the process of collecting the information and make the payments in the bank so the next step is to ingress the Drivers License Management System and set an appointment for the trial. George ingress the Drivers License Management System and register with their personal data and then login to the home web page where he can select the option of making an appointment and see the confirmation. The day of the appointment George goes to the trial center and a Staff will record his performance in the test and register the test result.

* + - Task-flows



* + - Swimlane diagrams
    - Storyboard for the applicant:

Graphical user interface, application

Description automatically generated

* + - Storyboard for Staff:

Graphical user interface, application

Description automatically generated

# Architectural Strategies

*<Describe any design decisions and/or strategies that affect the overall organization of the system and its higher-level structures. These strategies should provide insight into the key abstractions and mechanisms used in the system architecture. Describe the reasoning employed for each decision and/or strategy (referring to previously stated design goals and principles) and how any design goals or priorities were balanced or traded-off. Such decisions might concern (but are not limited to) things like the following:*

* *Use of a particular type of product (programming language, database, library, etc. ...)*
* *Reuse of existing software components to implement various parts/features of the system*
* *Future plans for extending or enhancing the software*
* *User interface paradigms (or system input and output models)*
* *Hardware and/or software interface paradigms*
* *Error detection and recovery*
* *Memory management policies*
* *External databases and/or data storage management and persistence*
* *Distributed data or control over a network*
* *Generalized approaches to control*
* *Concurrency and synchronization*
* *Communication mechanisms*
* *Management of other resources*

*Each significant strategy employed should be discussed in its own subsection, or (if it is complex enough) in a separate design document (with an appropriate reference here of course). Make sure that when describing a design decision that you also discuss any other significant alternatives that were considered, and your reasons for rejecting them (as well as your reasons for accepting the alternative you finally chose). Sometimes it may be most effective to employ the "pattern format" for describing a strategy.>*

# System Architecture

*<This section should provide a high-level overview of how the functionality and responsibilities of the system were partitioned and then assigned to subsystems or components. Don't go into too much detail about the individual components themselves. The main purpose here is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together to provide the desired functionality.>*

**DLMS Architecture Diagram**

DLMS

Database

Applicants

Staffs

server

Presentation

Layer

Logical Layer

Data Layer

Registration

and Login

Web Elements

Booking

Viewing

Results

Web Elements

Registration

and Login

Accepting

Rejecting

*A nice and clear software architecture diagram can be a good idea to be shown here along with some description.*

# Detailed System Design

*<Most components described in the System Architecture section will require a more detailed discussion. Other lower-level components and subcomponents may need to be described as well. Each subsection of this section will refer to or contain a detailed description of a system software component. The discussion provided should cover the following software component attributes:*

* *Classifications*
* *Attributes*
* *Responsibilities*
* *Composition/Aggregation/Generalization*

*A class diagram showing all of the above with some description should suffice.*

# Glossary

* Django - Django is a free and open-source, Python -based web framework that follows the model–view–Template (MVT) architectural pattern.

# References

*<A list of referenced and/or related publications.>*