*Florida International University*

*School of Computing and Information Sciences*

CIS 4911 Senior Capstone Project – **Software Engineering Focus**

**Format of Deliverable 3 - Design Document (DD)**

**The bullets represent the sections before chapter 1 “Introduction”, numbers on the left represent the corresponding chapters and sections.**

· Cover page – Name of course and section, name of system, project team number, group member names, date, and name of professor.

· Copyright and trademark notices, restrictions on copying or distributing the documentation, information for contacting the issuing organization (reader’s comments), warranties, contractual obligations or disclaimers, and general warnings and cautions.

· Abstract – one or two paragraphs giving a brief overview of the document.

· Table of Contents

1. Introduction

Introduce the introduction (one or two paragraphs)

1.1. Problem definition (very similar to RD).

1.2. Design methodology used e.g., identify software process model, ease of creating a design from the systems requirements, types of models (UML models) used to represent the design.

1.3. Definitions, acronyms, and abbreviations.

1.4. Overview of document

2. System Design (i.e., overall system design)

Introduce the system decomposition chapter (one or two paragraphs).

2.1. Overview – high-level description of the system design (architecture) e.g., provides a package diagram showing the major subsystems and briefly describes each subsystem. Relate the system decomposition to the requirements of the system. Use at least two (2) architectural patterns.

Our application will use a server-client architecture and all of the implementation will be on the server side, where we will use a MVC architecture

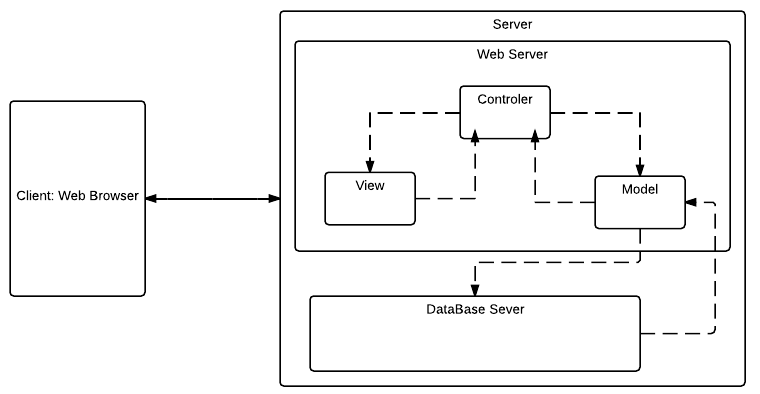
2.2. Subsystem Decomposition – provide a detailed description for each of the major subsystems. Identify the requirements associated with each subsystem.

The server subsystem will be implemented using a Model-View-Controller (MVC) architecture.

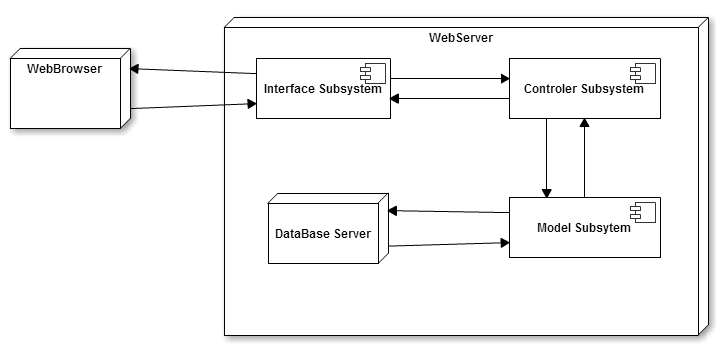
The Controller subsystem will handle the logic of the system and the interactions between the other Model and View subsystems. The Controller will receive the requests from the web browser, send that request for data to the Model subsystem, the data received from the Model is used to create the View which is presented in HTML via the web browser the user is using. In essence the Controller subsystem is responsible for behaving like an intermediary between the user and the data storage

The View subsystem contains and formats the information presented to the user. This information is received from the Controller subsystem and in turn sends user input to the Controller.

The Model subsystem handles the data storage, which will be done via a MySQL database



2.3. Hardware and Software Mapping – map subsystems to h/w and s/w. The h/w and s/w are for the systems to be implemented. May include a deployment diagram showing the associations between the subsystems and hardware.



2.4. Persistent Data Management – identify data that needs to be stored and the structure of the data. Use a data dictionary to represent the initial data extracted from the use cases.

The system will store two kinds of data: a profile for each user containing his name, rank, agency and Tier the user belongs to; and all the information pertinent to each BOLO created: type of BOLO, description of the subject, officer who created, date created, etc

2.5. Security/Privacy – describe user authentication processes, encryption of data, and use of firewalls or security servers.

SQL Injection The system handles SQL Injection attacks with the Active Record class from the CodeIgniter framework.

Data Encryption: the system encrypts the data from each session using the Encryption class from CodeIgniter

Cross-site scripting:CodeIgniter comes with a Cross Site Scripting Hack prevention filter that filters all POST and COOKIE data

Cross-site request forgery: CSRF is also addressed by enabling the Cross Site Scripting Hack prevention filter

3. Detailed Design

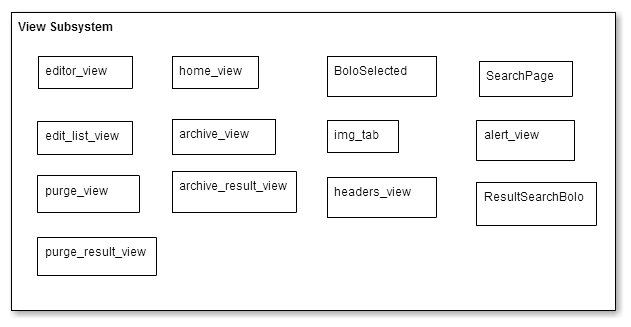
Introduce the detailed design chapter (one or two paragraphs)

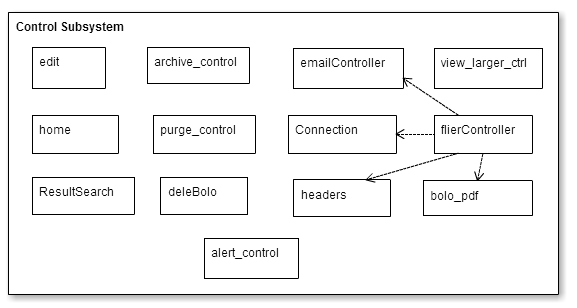
3.1. Overview – briefly describe the behavior and structure of each subsystem.

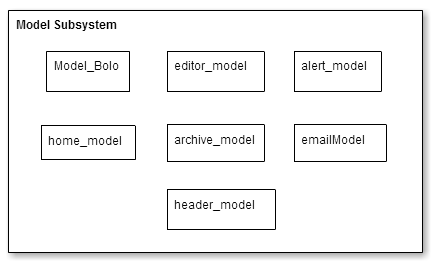
The system is decomposed into 3 subsystems: Model, Controller and View Subsystems. The Model Subsystem is responsible for the controlling access to the system’s database; the View Subsystem is responsible for drawing the user interface and handling the user’s interactions; the Controller Subsystem handles all the logic, it acts as an intermediary between the View and the Model and processes and validates the data the user inputs

3.2. Static model – detailed description of the structure for each subsystem. May include detailed class diagrams. Place diagrams (e.g., minimal class diagram, detailed class diagram per subsystem) inline. Use at least four (4) design patterns.

The following images list the classes that compose each subsystem. The classes are organized by functionality: for example, all the model, view and control classes needed to implement the Edit functionality are grouped in one folder; the ones needed for the homepage are in another folder, and so on. In the case of the Model subsystem, some of the most commonly used functions are grouped into one big class called Model\_Bolo.php. *NOTE:* classes in a subsystem are not always related to each other, but to classes in the other subsystems

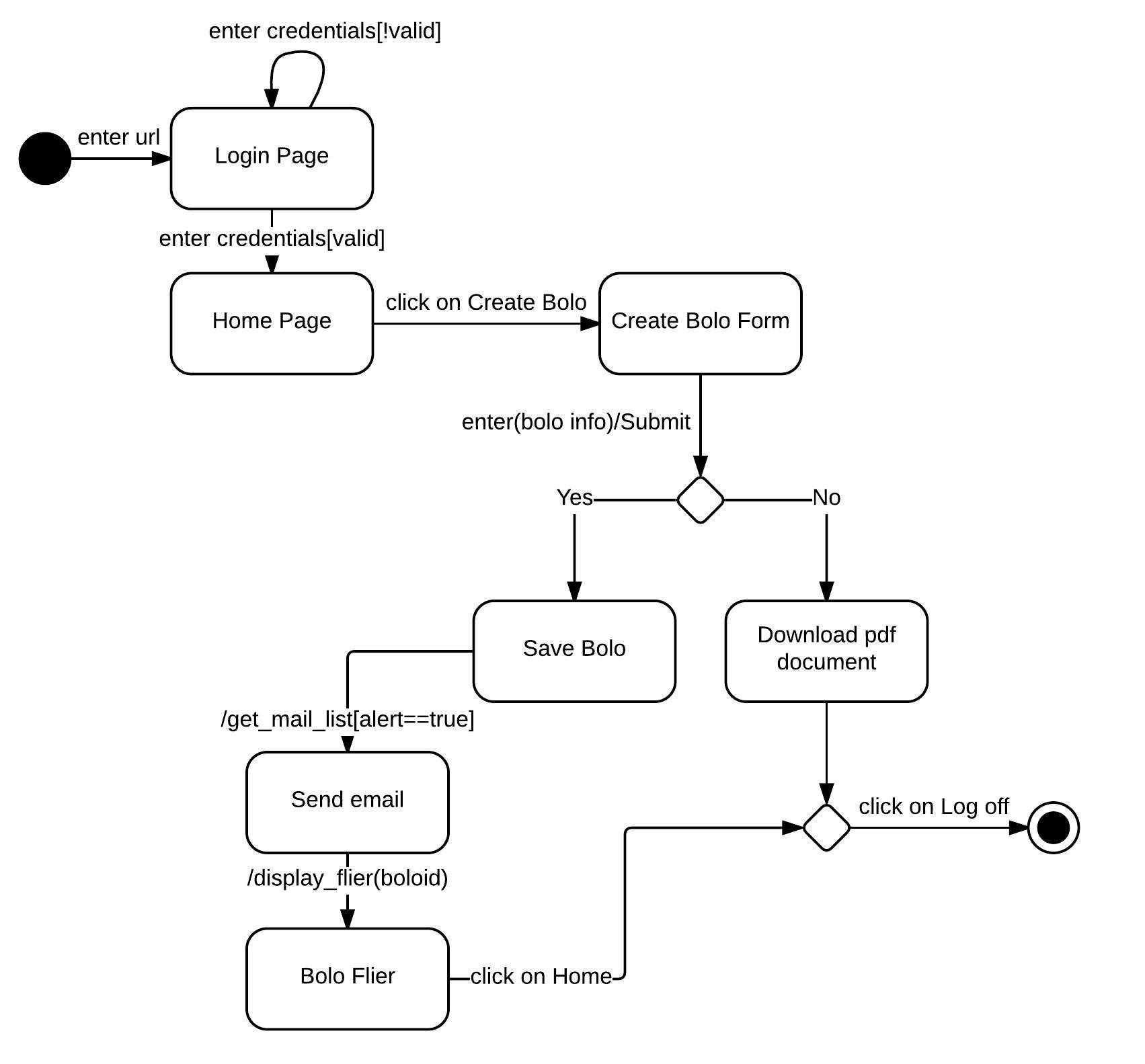






3.3. Dynamic model – state machine diagram for the main control object in each subsystem. Include the design of the ***main algorithms*** used in the problem solution. Refinement of the sequence diagram from the analysis model. Place diagrams inline.

The following state machine shows the process of login in and out, submitting a BOLO and creating BOLO as a pdf document



3.4. Code Specification - describe the class interfaces (attributes and method signatures) and constraint (invariants, precondition and post-conditions) for the main control object in each system. Code should be in Appendix C.

4. Glossary - define terms used in document, especially domain specific terms.

5. Appendix

5.1. Appendix A - Use case diagram for use cases being implemented.

5.2. Appendix B - Use cases being implemented (from the RD).

5.3. Appendix C – Documented class interfaces (code) for the subsystem(s) you will implement and the constraints.

5.4. Appendix D - Diary of meeting and tasks.

6. References