White Volley Girls

CSUN Dashboard

Software Design Document

Name (s):
Christian J.
Nima S
Michael B.
David H.

Section:
Workstation:

Date: (24/October/2022)

Software Design Document

TABLE OF CONTENTS

Software Design Document 1

1.0 INTRODUCTION 3

1.1 Purpose 3

1.2 Scope 3

1.3 Overview 3

2.0 SYSTEM OVERVIEW 3

3.0 SYSTEM ARCHITECTURE 3

3.1 Architectural Design 3

3.2 Interface Design 4

3.3 Decomposition Description 4

3.4 Design Rationale 4

4.0 COMPONENT DESIGN/DETAILED DESIGN 5

5.1 Overview of User Interface 7

5.2 Screen Images 7

Screen Objects and Actions 9

1.0 REQUIREMENTS MATRIX 9

APPENDICES 10

1.0 INTRODUCTION

1.1 Purpose

This software design document describes the architecture and system design of the CSUN Dashboard

1.2 Scope

The defragmentation of information for the consumers is the ultimate end goal of this project will allow them to effectively plan and coordinate their financial and short-term/long-term projections and movement.

1.3 Overview

This document covers the technical philosophy and approach to this product. Such things covered are the entire application layer regarding technologies used and the UI/UX design choices.

2.0 SYSTEM OVERVIEW

There exists information about CSUN in terms of Catalogs, Majors, and Professors that are stored in various schemas across a database in MySQL and JSON documents that are handled and served by Python to a ReactJS built frontend as requested by the user.

3.0 SYSTEM ARCHITECTURE

3.1 Architectural Design

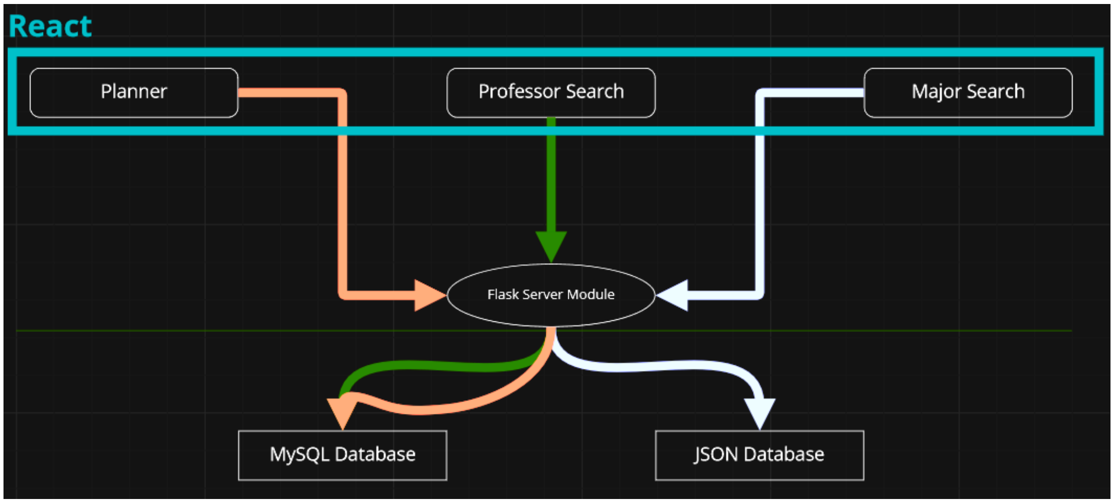


Figure 3.1: Modular System Design

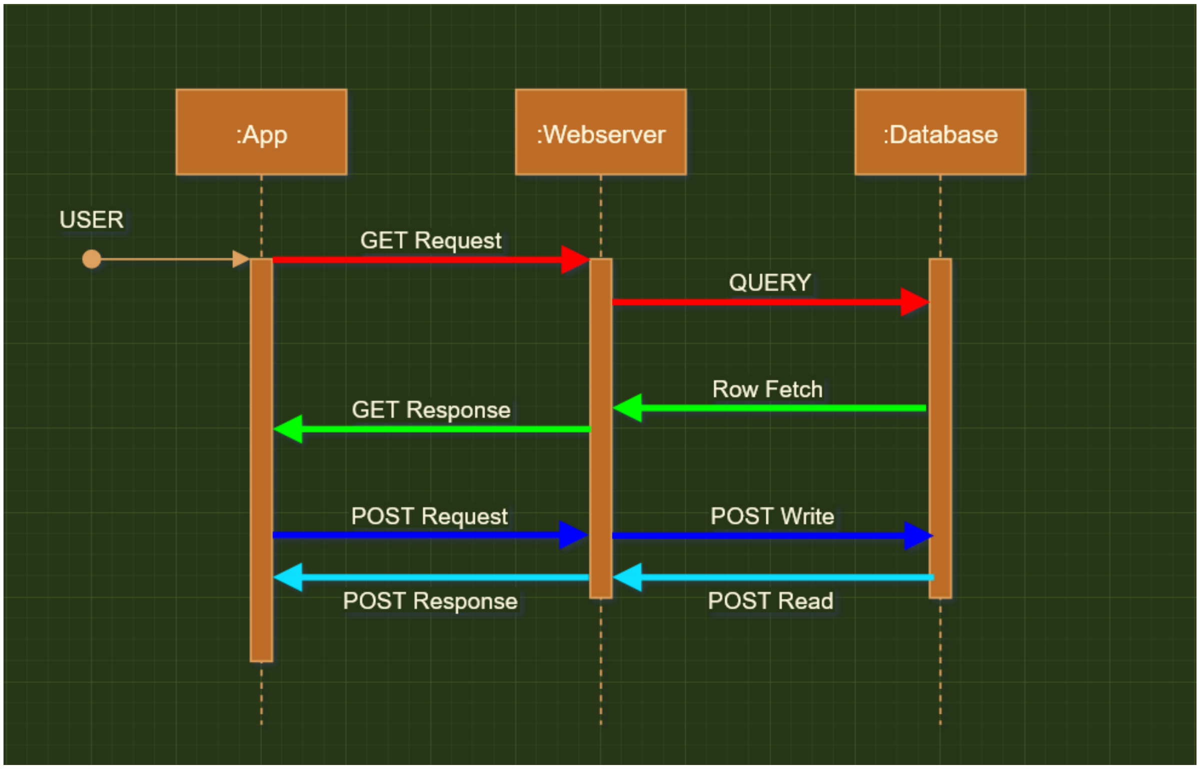
The colors indicate what database each tool is pulling from.

The middle man between the client and server is the Flask Server Module that facilitates communication between the two. The planner and 2 search tools send GET requests to the server module which then accesses the appropriate databases for the information requested.

3.2 Interface Design

Provide textual or image/picture of how the CSCIs interface/relate to each other

3.3 Decomposition Description



Having a middle man between the databases and the client was important for the simplification of client-side operations regarding deserialization of data received from the databases. This reduces the resource usage both client side and server side as the server module is configured such a way to not send unneeded data.

3.4 Design Rationale

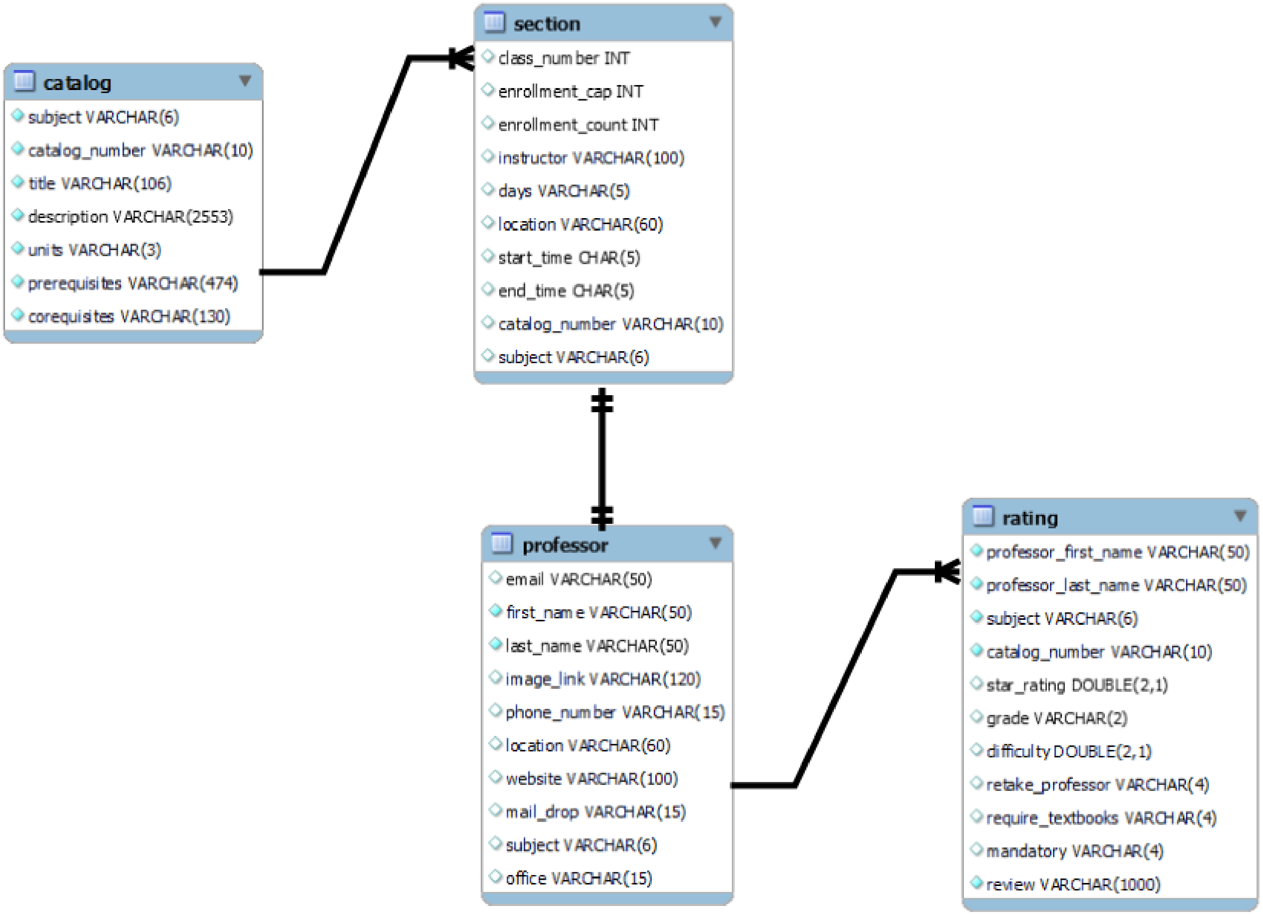
The Flask Server Module placed between the Database and Frontend modules was important for cohesion, compatibility, and security in the entire product.

- Cohesion
  - Having all the data processing happening in one place in one language was important for debugging allowing for faster testing and deployment in the entire development lifecycle
- Compatibility

  - Due to the open source nature of the multitude of libraries that are provided for database access in NodeJS, it would be mildly difficult to keep with the changes that happen in each one.
- Security
  - Since ReactJS is also running on the client, this exposes the query statements to both of the sides of the client-server architecture allowing users to possible proxy between their machine and a request, possibly manipulating queries that are sent to server and revealing sensitive information.

4.0 COMPONENT DESIGN/DETAILED DESIGN

4.1 Class Diagrams



4.1.2 Database Schemas

- MySQL was used to store all subject catalogs and professor information
  - For every subject in the catalog schema, views were created on a per

subject basis

- **Schemas**
    - **Professor Schema (Name: DATATYPE)**
        - **Email: VARCHAR(50)**
        - **First_name: VARCHAR(50)**
        - **Last_name: VARCHAR(50)**
        - **Image_link: VARCHAR(120)**
        - **Phone_number: VARCHAR(15)**
        - **Location: VARCHAR(60)**
        - **Website: VARCHAR(100)**
        - **Mail_drop: VARCHAR(15)**
        - **Subject: VARCHAR(6)**
        - **Office: VARCHAR(15)**
    - **Rating Schema (Name: DATATYPE)**
        - **Professor_First_Name: VARCHAR(50)**
        - **Professor_Last_Name: VARCHAR(50)**
        - **Subject: VARCHAR(6)**
        - **Catalog_number: VARCHAR(10)**
        - **Star_rating: DOUBLE(2,1)**
        - **Grade: VARCHAR(2)**
        - **Difficulty: DOUBLE(2,1)**
        - **Retake_Professor: VARCHAR(4)**
        - **Require_Textbooks: VARCHAR(4)**
        - **Mandatory: VARCHAR(4)**
        - **Review: VARCHAR(1000)**
    - **Catalog Schema (Name: DATATYPE)**
        - **Subject: VARCHAR(6)**
        - **Catalog_Number: VARCHAR(10)**
        - **Title: VARCHAR(106)**
        - **Description: VARCHAR(2553)**
        - **Units: VARCHAR(3)**
        - **Prerequisites: VARCHAR(474)**
        - **Corequisites: VARCHAR(130)**
    - **Section Schema (Name: DATATYPE)**
        - **Class_Number: INT**
        - **Enrollment_Cap: INT**
        - **Enrollment_Count: INT**
        - **Instructor: VARCHAR(100)**
        - **Days: VARCHAR(5)**
        - **Location: VARCHAR(60)**
        - **Start_time: CHAR(5)**
        - **End_time: CHAR(5)**
        - **Catalog_number: VARCHAR(10)**
        - **Subject: VARCHAR(6)**
- JSON Documents were used to store the Major descriptions. They are just strings of explanation about the major stored in an array that is meant to be fetched on load and organized according to what the major is.

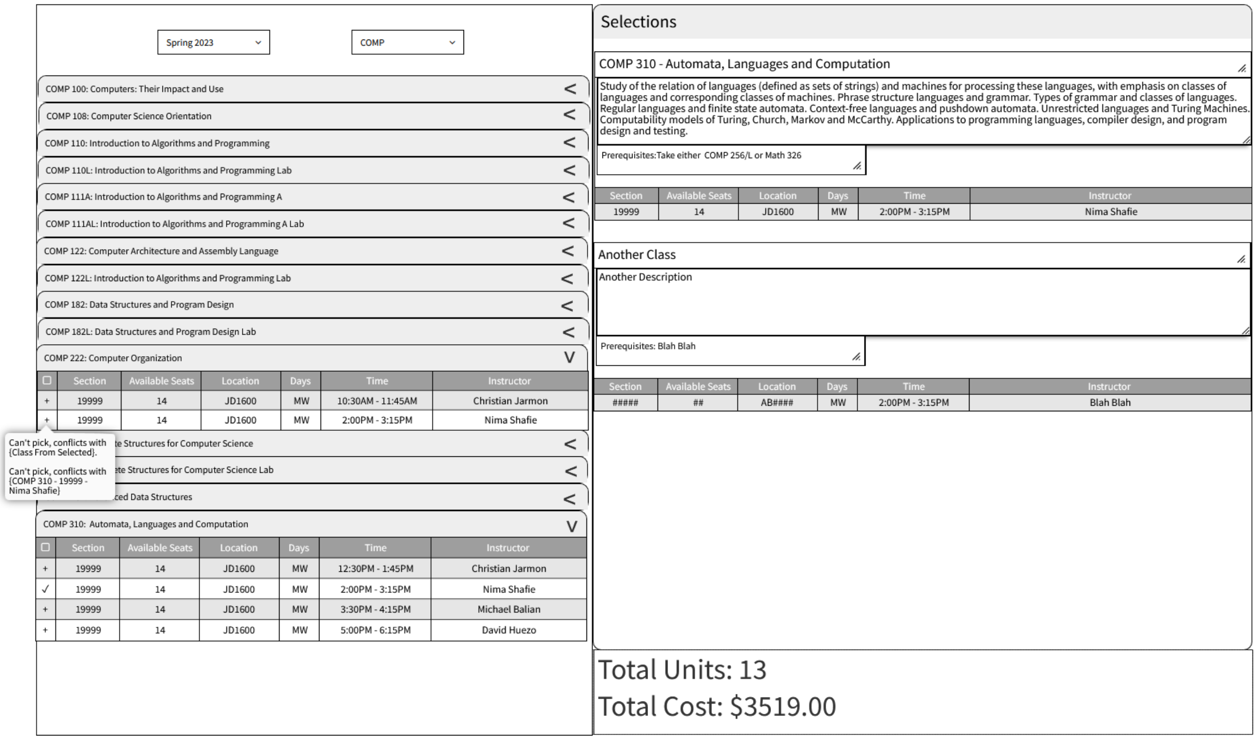
5.0 User Interface

5.1 Overview of User Interface

Each client side module can be imagined as a Domain-Specific search engine.

- In the Planner module, the parameters for every search are as follows
    - Semester (Already present dropdown)
    - Subject (Already present dropdown)
    - Which then displays a list of all the {Subject} courses scheduled in the {Semester} on the left-half of the screen
        - For every course chosen by the user, the course will show up listed on the right-half of the screen
- In the Professor Search, the parameters for every search are as follows
    - Subject (Already present dropdown)
    - Professor (Dropdown that loads on {Subject} choice)
    - In the list of the professors, each will be linked to a ratings page listing reviews made by other students.
- In the Major Search, there will be a list of majors offered by the university that when clicked by the user, will lead to another page listing all the requirements.

5.2 Screen Images



COMP 100: Computers: Their Impact and Use <

COMP 108: Computer Science Orientation <

COMP 110: Introduction to Algorithms and Programming <

COMP 110L: Introduction to Algorithms and Programming Lab <

COMP 111A: Introduction to Algorithms and Programming A <

COMP 111AL: Introduction to Algorithms and Programming A Lab <

COMP 122: Computer Architecture and Assembly Language <

COMP 122L: Introduction to Algorithms and Programming Lab <

COMP 182: Data Structures and Program Design <

COMP 182L: Data Structures and Program Design Lab <

COMP 222: Computer Organization V

□	Section	Available Seats	Location	Days	Time	Instructor
+	19999	14	JD1600	MW	10:30AM - 11:45AM	Christian Jarmon
+	19999	14	JD1600	MW	2:00PM - 3:15PM	Nima Shafie

Can't pick, conflicts with {Class From Selected}.

Can't pick, conflicts with {COMP 310 - 19999 - Nima Shafie}

te Structures for Computer Science <

ete Structures for Computer Science Lab <

ced Data Structures <

COMP 310: Automata, Languages and Computation V

□	Section	Available Seats	Location	Days	Time	Instructor
+	19999	14	JD1600	MW	12:30PM - 1:45PM	Christian Jarmon
✓	19999	14	JD1600	MW	2:00PM - 3:15PM	Nima Shafie
+	19999	14	JD1600	MW	3:30PM - 4:15PM	Michael Balian
+	19999	14	JD1600	MW	5:00PM - 6:15PM	David Huezo

COMP 310 - Automata, Languages and Computation

Study of the relation of languages (defined as sets of strings) and machines for processing these languages, with emphasis on classes of languages and corresponding classes of machines. Phrase structure languages and grammar. Types of grammar and classes of languages. Regular languages and finite state automata. Context-free languages and pushdown automata. Unrestricted languages and Turing Machines. Computability models of Turing, Church, Markov and McCarthy. Applications to programming languages, compiler design, and program design and testing.

Prerequisites:Take either COMP 256/L or Math 326

Section	Available Seats	Location	Days	Time	Instructor
19999	14	JD1600	MW	2:00PM - 3:15PM	Nima Shafie

Another Class

Another Description

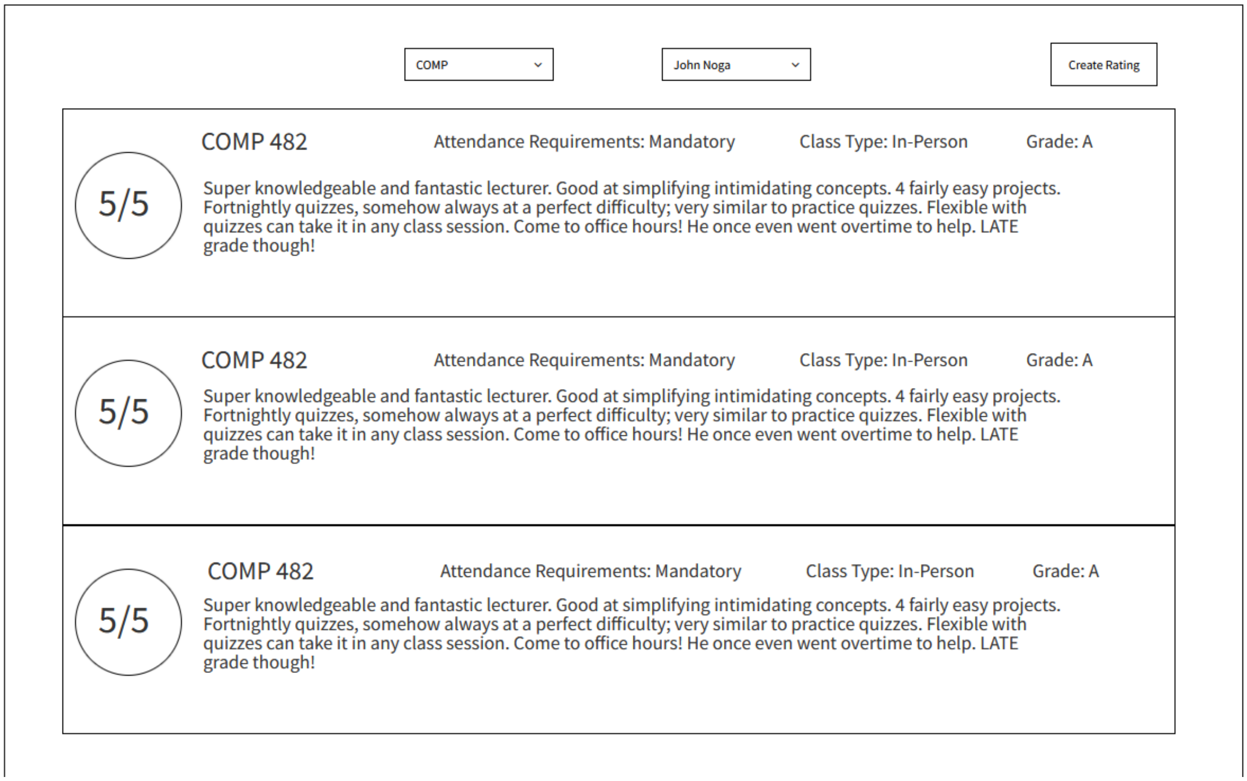
Prerequisites: Blah Blah

Section	Available Seats	Location	Days	Time	Instructor
#####	##	AB####	MW	2:00PM - 3:15PM	Blah Blah

Total Units: 13

Total Cost: $3519.00

Figure 5.1: Planner Page Wireframe



5/5

COMP 482 Attendance Requirements: Mandatory Class Type: In-Person Grade: A

Super knowledgeable and fantastic lecturer. Good at simplifying intimidating concepts. 4 fairly easy projects. Fortnightly quizzes, somehow always at a perfect difficulty; very similar to practice quizzes. Flexible with quizzes can take it in any class session. Come to office hours! He once even went overtime to help. LATE grade though!

5/5

COMP 482 Attendance Requirements: Mandatory Class Type: In-Person Grade: A

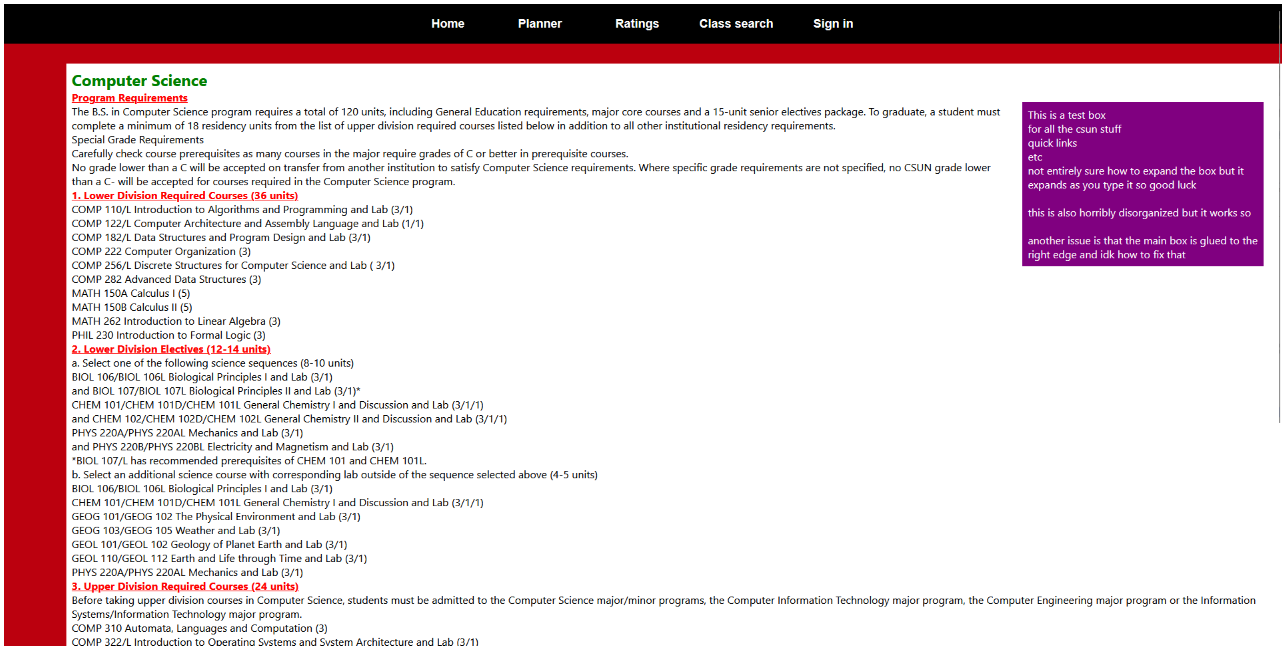
Super knowledgeable and fantastic lecturer. Good at simplifying intimidating concepts. 4 fairly easy projects. Fortnightly quizzes, somehow always at a perfect difficulty; very similar to practice quizzes. Flexible with quizzes can take it in any class session. Come to office hours! He once even went overtime to help. LATE grade though!

5/5

COMP 482 Attendance Requirements: Mandatory Class Type: In-Person Grade: A

Super knowledgeable and fantastic lecturer. Good at simplifying intimidating concepts. 4 fairly easy projects. Fortnightly quizzes, somehow always at a perfect difficulty; very similar to practice quizzes. Flexible with quizzes can take it in any class session. Come to office hours! He once even went overtime to help. LATE grade though!

Figure 5.2: Professor Ratings wireframe



Computer Science

Program Requirements

The B.S. in Computer Science program requires a total of 120 units, including General Education requirements, major core courses and a 15-unit senior electives package. To graduate, a student must complete a minimum of 18 residency units from the list of upper division required courses listed below in addition to all other institutional residency requirements.

Special Grade Requirements

Carefully check course prerequisites as many courses in the major require grades of C or better in prerequisite courses.

No grade lower than a C will be accepted on transfer from another institution to satisfy Computer Science requirements. Where specific grade requirements are not specified, no CSUN grade lower than a C- will be accepted for courses required in the Computer Science program.

1. Lower Division Required Courses (36 units)

COMP 110/L Introduction to Algorithms and Programming and Lab (3/1)
COMP 122/L Computer Architecture and Assembly Language and Lab (1/1)
COMP 182/L Data Structures and Program Design and Lab (3/1)
COMP 222 Computer Organization (3)
COMP 256/L Discrete Structures for Computer Science and Lab (3/1)
COMP 282 Advanced Data Structures (3)
MATH 150A Calculus I (5)
MATH 150B Calculus II (5)
MATH 262 Introduction to Linear Algebra (3)
PHIL 230 Introduction to Formal Logic (3)

2. Lower Division Electives (12-14 units)

a. Select one of the following science sequences (8-10 units)
BIOL 106/BIOL 106L Biological Principles I and Lab (3/1)
and BIOL 107/BIOL 107L Biological Principles II and Lab (3/1)*
CHEM 101/CHEM 101D/CHEM 101L General Chemistry I and Discussion and Lab (3/1/1)
and CHEM 102/CHEM 102D/CHEM 102L General Chemistry II and Discussion and Lab (3/1/1)
PHYS 220A/PHYS 220AL Mechanics and Lab (3/1)
and PHYS 220B/PHYS 220BL Electricity and Magnetism and Lab (3/1)
*BIOL 107/L has recommended prerequisites of CHEM 101 and CHEM 101L.
b. Select an additional science course with corresponding lab outside of the sequence selected above (4-5 units)
BIOL 106/BIOL 106L Biological Principles I and Lab (3/1)
CHEM 101/CHEM 101D/CHEM 101L General Chemistry I and Discussion and Lab (3/1/1)
GEOG 101/GEOG 102 The Physical Environment and Lab (3/1)
GEOG 103/GEOG 105 Weather and Lab (3/1)
GEOL 101/GEOL 102 Geology of Planet Earth and Lab (3/1)
GEOL 110/GEOL 112 Earth and Life through Time and Lab (3/1)
PHYS 220A/PHYS 220AL Mechanics and Lab (3/1)

3. Upper Division Required Courses (24 units)

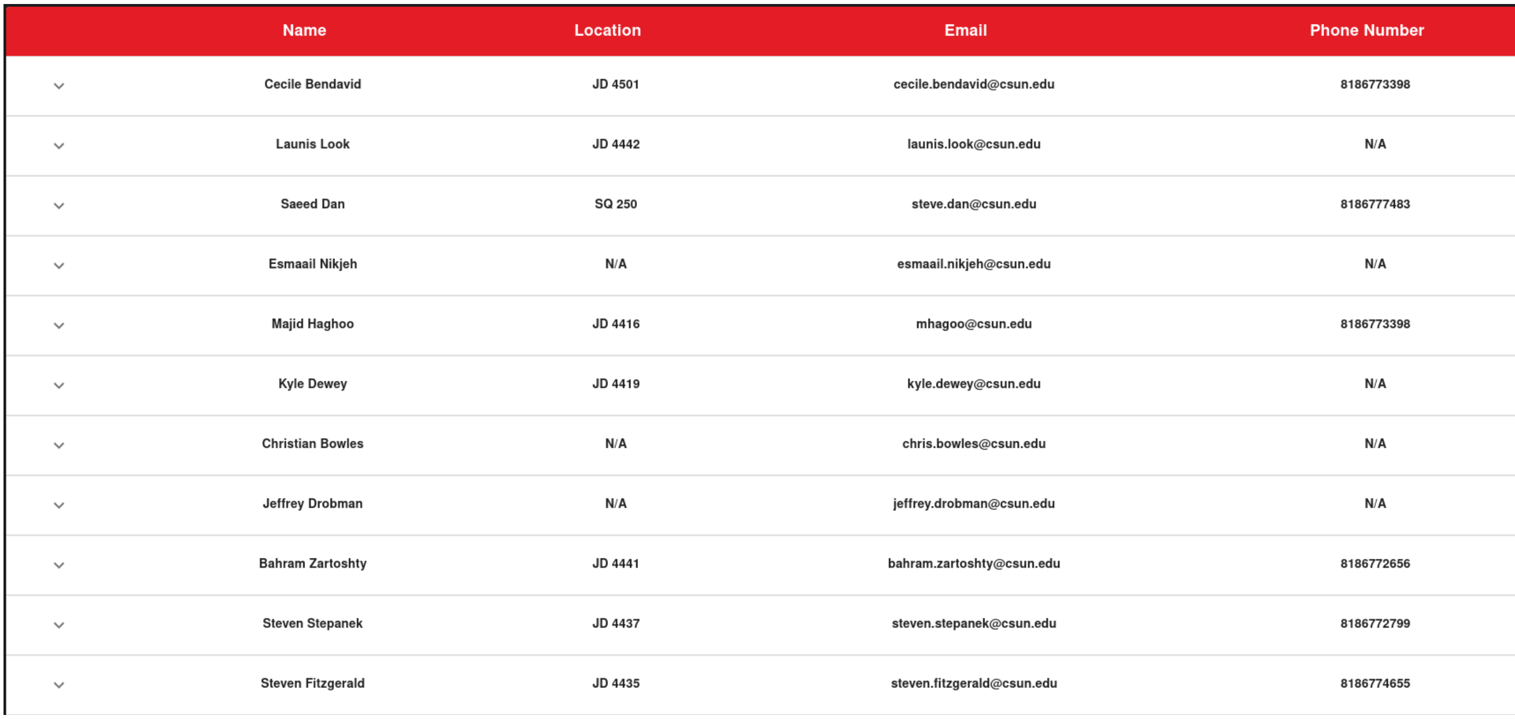
Before taking upper division courses in Computer Science, students must be admitted to the Computer Science major/minor programs, the Computer Information Technology major program, the Computer Engineering major program or the Information Systems/Information Technology major program.
COMP 310 Automata, Languages and Computation (3)
COMP 322/L Introduction to Operating Systems and System Architecture and Lab (3/1)

This is a test box
for all the csun stuff
quick links
etc
not entirely sure how to expand the box but it expands as you type it so good luck

this is also horribly disorganized but it works so

another issue is that the main box is glued to the right edge and idk how to fix that

Figure 5.3: Computer Science Major page Rough Draft



	Name	Location	Email	Phone Number
⌄	Cecile Bendavid	JD 4501	cecile.bendavid@csun.edu	8186773398
⌄	Launis Look	JD 4442	launis.look@csun.edu	N/A
⌄	Saeed Dan	SQ 250	steve.dan@csun.edu	8186777483
⌄	Esmaail Nikjeh	N/A	esmaail.nikjeh@csun.edu	N/A
⌄	Majid Haghoo	JD 4416	mhagoo@csun.edu	8186773398
⌄	Kyle Dewey	JD 4419	kyle.dewey@csun.edu	N/A
⌄	Christian Bowles	N/A	chris.bowles@csun.edu	N/A
⌄	Jeffrey Drobman	N/A	jeffrey.drobman@csun.edu	N/A
⌄	Bahram Zartoshty	JD 4441	bahram.zartoshty@csun.edu	8186772656
⌄	Steven Stepanek	JD 4437	steven.stepanek@csun.edu	8186772799
⌄	Steven Fitzgerald	JD 4435	steven.fitzgerald@csun.edu	8186774655

Figure 5.4: Professor Search

SRS Req. ID	Satisfied (Yes/No)	Satisfaction Component/Proof
FUNC_SRS_(1.0)	**Yes**	Top layer in Figure 3.1
FUNC_SRS_(2.0)	**Yes**	Figure 5.1
FUNC_SRS_(2.1)	**Yes**	Figure 5.1
FUNC_SRS_(2.2)	**Yes**	Figure 5.1
FUNC_SRS_(2.3)	**Yes**	Figure 5.1
FUNC_SRS_(2.4)	**Yes**	Figure 5.1
FUNC_SRS_(3.0)	**Yes**	Figure 5.4
FUNC_SRS_(3.1)	**Yes**	Figure 5.2
FUNC_SRS_(3.2)	**Yes**	Figure 5.4
FUNC_SRS_(4.0)	**Yes**	Figure 5.3

APPENDICES

This section is optional.

Appendices may be included, either directly or by reference, to provide supporting details that could aid in the understanding of the Software Design Document.