12/04/2024

CU DEGREE PLANNER

TEAM MEMBERS [GITHUB USERNAME]: TRUMAN DAVIS [TRUST-E], HANNAH MURTHA [HANNAHMURTHA], DARALYNN RHODE [DRHODE03], NICK TISHKOWSKI [SALLYGATOR], AND MARCUS WINTON [MWINTON05506]

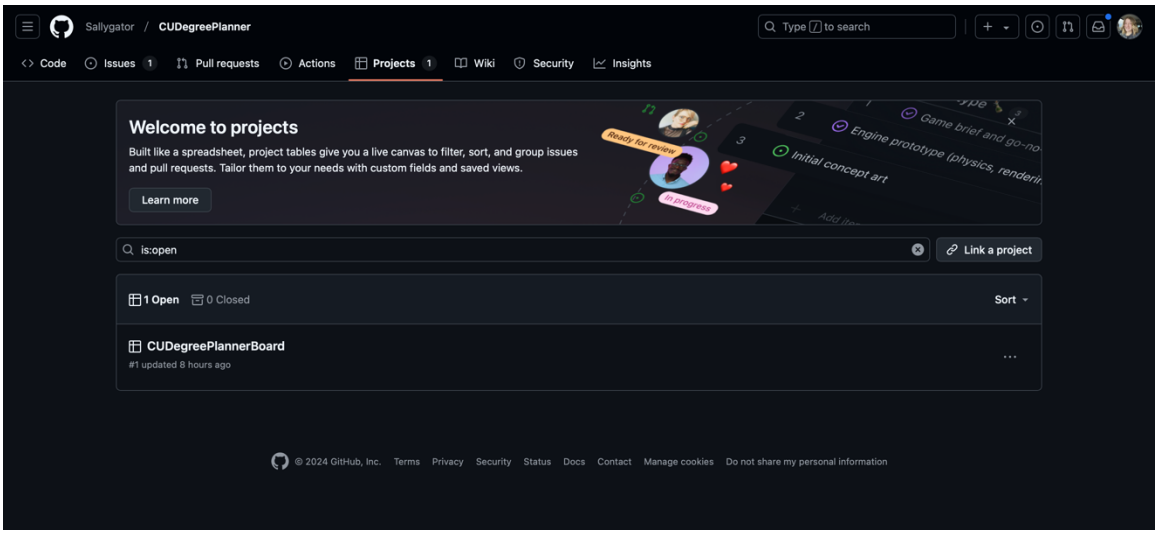
PROJECT DESCRIPTION

An interface to assist CU Boulder Computer Science Undergraduate Students with planning their degree coursework. Students will be given a list of classes which they can then drag and drop into their semesters. Needed pre-reqs will be highlighted for them. This application is meant to take away the stress of class selection by streamlining the process. A course catalog navigator allows students to search for needed classes and credits.

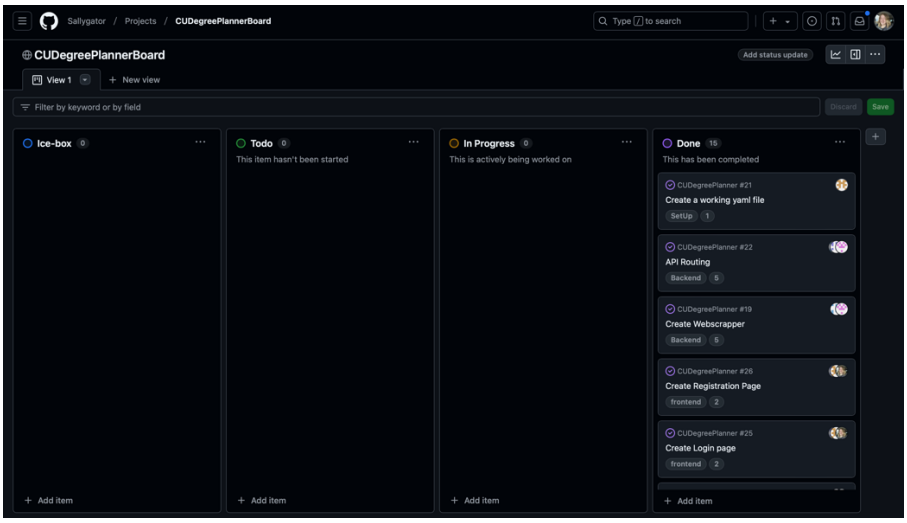
PROJECT TRACKER

Link to Project Board on GitHub: https://github.com/users/Sallygator/projects/1

Project Board Screenshot:



Project Board Screenshot:



VIDEO

YouTube Link: https://www.youtube.com/watch?v=pG2IOp2gnGA

VCS

GitHub Repository Link: https://github.com/Sallygator/CUDegreePlanner

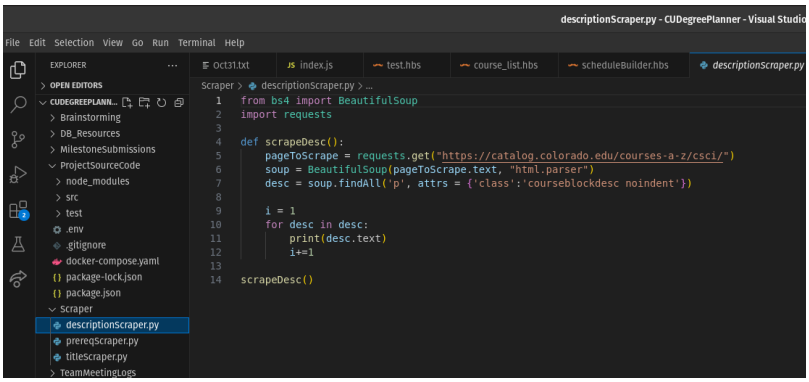
CONTRIBUTIONS

The following is a summary of contributions and screenshots made by each member.

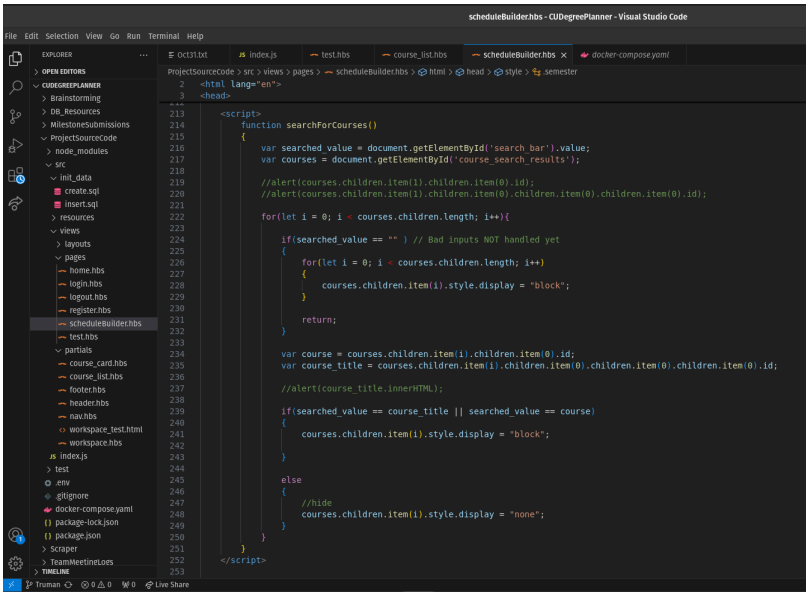
Truman Davis

i. My contributions to the project were in the initial data collection and backend work. I wrote the web scrapers we used to make our API. I helped in formatting the database tables to meet our goals and make function implementation easier. Lastly, I implemented the search function on the site. I used Python for the scrapers and scraped the CU CS courses list to gather all the names and details of the courses essential to the project. We used Postgresql for the database and Jquery to put together the sortables that made the search function work.

ii. Scrapper:



Course Search:



Hannah Murtha

i. My main contribution to the project was User Interface design and implementation, where I styled the Home, Login, Register, Navbar and Schedule Builder pages, aiming for a user-friendly experience. Besides that, I integrated the frontend and backend components together, dealing with merge conflicts for the team and adapting to scripts as needed. I wrote the APIs for logging in, registering, logging out, and the YAML file. I also performed user-focused testing on the website itself, trying to identify issues by intentionally breaking the website. Lastly, I documented meeting notes for the team. Technology wise, for the frontend I used HTML, CSS, Handlebars, and Bootstrap. For backend I used GitHub, JavaScript and Node.Js.

ii. Styling:

```
<style>
    body /* background stuff */
    {
        margin: 0;
        display: flex;
        flex-direction: column;
        height: 100%;
        width: 100%;
        background: linear-gradient(135deg, rgb(171, 143, 107), #648cc4); /* Gradient background */
    }

    .container /* holds all content */
    {
        display: auto;
        overflow: hidden;
        height: fit-content;
    }

    .card
    {
        width: 90%;
        height: fit-content;
        background-color: #cfc08dba;
    }

    .sidebar /* sidebar main */
    {
        width: 30%;
        height: 100%; /* changes the heihht of the sidebar, very broken*/
        box-shadow: 0 4px 10px rgba(0, 0, 0, 0.9);
        background-color: rgba(0, 0, 0, 0.731);
        overflow-y: auto;
        display: auto;
        border: 5px solid #605b5b00;
        border-radius: 5px;
    }

    /* Flexbox for input and button container */
    .sidebar .search-container
    {
        display: flex;              /* Align children horizontally */
        align-items: center;        /* Vertically center the input and button */
        margin: 2% 7%;              /* Adjust the margin for better spacing */
```

Index Register:

```
app.post('/register', async (req, res) =>
{
  const { username, password, degree } = req.body; //getting request info
  if (!username || !password || !degree) {
    return res.status(400).json({ message: 'Missing required fields.' });
  }
  try
  {
  //hash the password using bcrypt library
  const hash = await bcrypt.hash(req.body.password, 10);

  // To-DO: Insert username and hashed password into the 'users' table
  await db.query('INSERT INTO users(username, password, degree) VALUES ($1, $2, $3);', [username, hash, degree]);
  console.log("User was inserted into the database");
  return res.redirect('/login'); //it worked, redirect to login route
  }

  catch(error)
  {

    // Log the error details for debugging
    console.error("Error during registration:", error);

    // Return a 500 Internal Server Error with a message
    return res.status(500).json({ message: 'Internal server error.' });

    return res.redirect('/register'); //didnt work, go back to the register page
  }
});
```

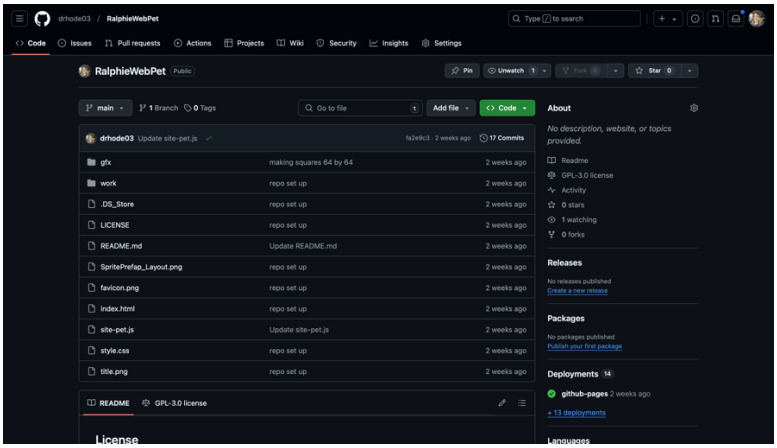
Register:

```
<body>
  <div class="content-wrapper">
    <form method="POST" action="/register">
      <div class="form-group px-5 mt-3">
        <h1>Register</h1>

        <label for="username" class="form-label">Username</label>
        <input name="username" class="form-control" id="username" required>

        <label for="password" class="form-label">Password</label>
        <input name="password" class="form-control" id="password" type="password" required>

        <label for="degree" class="form-label">Degree</label>
        <select name="degree" class="form-control" id="degree" required>
          <option value="">Select Degree</option>
          <option value="ComputerScience">Computer Science</option>
        </select>
      </div>

      <div class="form-group mb-3 px-5 mt-3">
        <button type="submit" class="btn btn-secondary">Register</button>
      </div>

      <div class="form-group px-5">
        <p>Already have an account? <a href="/login">Login</a> </p>
      </div>
    </form>
  </div>
</body>
```

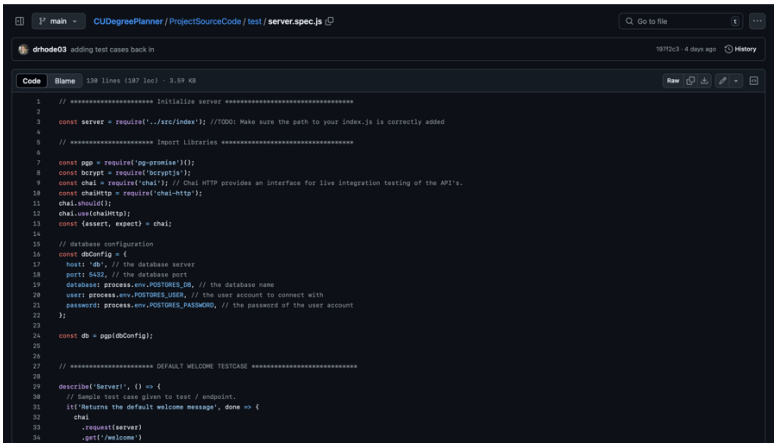
Daralynn Rhode

i. My main contributions to the project were website design/frontend development, test implementation, and application hosting. I also did most of the "glue work" (documentation, maintaining boards, lab-work, etc.) and team organization related to maintaining and updating the GitHub repo. The addition of the Web Pet Ralphie was also my work, I created a separate repo to house the functionality. Technology wise, I worked with Chai and Mocha for the testing, Node.Js, Javascript, Handlebars, and HTML for the frontend portions, and with Render for the cloud hosting.

ii. Pet Ralphie Repo:



Test Cases:



Nick Tishkowski

i. My contributions to the project included populating individual class cards from our database; each containing a class key, description and a list of prerequisites. I then implemented the drag-and-drop functionality of the class cards in the Schedule Builder page, the ability to search for classes based on what requirement they make in the search bar, and lastly to detect if all pre-requirements were met when the user dropped a class into their schedule. If all

requirements were not met, the requirement text would turn red, otherwise it would turn green. I used HTML, Handlebars, Boostrap. JavaScript and Node.Js.

ii. Populate Cards:

```
{{!-- Displays class info when a class card is clicked --}}
<script>
    function displayclassInfo(id){
        document.getElementById().innerHTML;
    }
</script>

{{!-- Search & Prereq Functions Below --}}
<script>
    function createFooterDescription(classcode, name, description, prerecs){
        document.getElementById("footer-class-title").innerHTML = name;
        document.getElementById("footer-class-code").innerHTML = classcode;
        document.getElementById("footer-class-description").innerHTML = description;
    }
</script>

{{! Handles moving cards programatically}}
<script>
    function moveClassCard(classid, targetid){
        var card = document.getElementById(classid);
        document.getElementById(targetid).append(card.parentElement);
    }

    //function setupTemplate(){
        //moveClassCard("CSCI1300", "Year 1_Fall");
        //moveClassCard("CSCI1200", "Year 1_Fall");
        //moveClassCard("MATH1300", "Year 1_Fall");
        //moveClassCard("PHYS1110", "Year 1_Fall");
    //}
</script>
```

Check PreReqs:

```
{{! Logic for checking pre-recs}}
<script>
    function checkPrerecs(classid, yearsem){
        var card = document.getElementById(classid);
        var prerecs = document.getElementById(classid + "_required");
        var current_table = yearsem;
        if (prerecs != null) {
            var year = parseInt(current_table.id.charAt(5));
            var semester = current_table.id.charAt(7);

            var prerecs_parsed = prerecs.innerHTML.slice(9).split(",");
            var satisfied = new Array(prerecs_parsed.length).fill(false);

            for (let i = 1; i <= year; i++){
                var check_year_fall = document.getElementById("Year " + i + "_Fall")
                for (let j = 0; j < check_year_fall.childElementCount; j++) {
                    var check_card = check_year_fall.children.item(j).children.item(0);
                    if (check_card != null) {var check_countsfor = document.getElementById(check_card.id + "_countsfor");}
                    if (check_countsfor != null) {
                        var check_countsfor_parsed = check_countsfor.innerHTML.slice(11);
                        for (let k = 0; k < prerecs_parsed.length; k++){
                            if (check_countsfor_parsed == prerecs_parsed[k]){
                                satisfied[k] = true;
                            }
                        }
                    }
                }
                if (semester == "S"){
                    var check_year_fall = document.getElementById("Year " + year + "_Fall")
                    for (let j = 0; j < check_year_fall.childElementCount; j++) {
                        var check_card = check_year_fall.children.item(j).children.item(0);
                        if (check_card != null) {var check_countsfor = document.getElementById(check_card.id + "_countsfor");}
                        if (check_countsfor != null) {
                            var check_countsfor_parsed = check_countsfor.innerHTML.slice(11);
                            for (let k = 0; k < prerecs_parsed.length; k++){
                                if (check_countsfor_parsed == prerecs_parsed[k]){
                                    satisfied[k] = true;
                                }
                            }
                        }
                    }
                }
```

Marcus Winton

i. My main contribution to the project was the design and creation of the database used to manage all of the classes. This included solving the problem of classes having multiple prerequisites, as well as handling multiple classes that could fulfill a given prerequisite. Additionally, I cleaned the data gathered from the web scraper. I then worked on implementing the technologies used, including Python for web scraping, PostgreSQL for database management, and Google Sheets for exporting data as a CSV. This process also involved the painstaking task of converting the prerequisite codes provided by the scraper into our "counts_for" column.

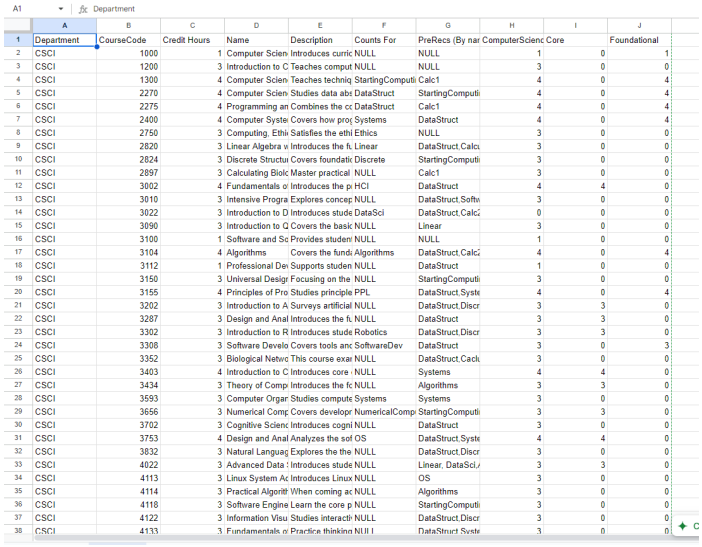
ii. Database Population:

```
INSERT INTO courseregistry (ClassCode, CreditHours, Description, Name, PreRecs, CountsFor, Core, Foundational, ComputerScience)

VALUES
('CSCI1000', 1, 'Introduces curriculum, learning techniques, time management and career opportunities in Computer Science. Includes presentations from alumni and others with relevant educational and professional experience.', 'Computer Science as a Field of Work and Study', NULL, NULL, 0, 1, 1),

('CSCI1200', 3, 'Teaches computational thinking and techniques for writing computer programs using the Python programming language. Intended for students who realize that computational skills are beneficial to all fields of study, but who have little or no experience in programming or are not Computer Science majors. Studen

('CSCI1300', 4, 'Teaches techniques for writing computer programs in higher level programming languages to solve problems of interest in a range of application domains. Appropriate for students with little to no experience in computing or programming.', 'Computer Science 1: Starting Computing', 'Calc1', 'StartingComputing',

('CSCI2270', 4, 'Studies data abstractions (e.g., stacks, queues, lists, trees, graphs, heaps, hash tables, priority queues) and their representation techniques (e.g., linking, arrays). Introduces concepts used in algorithm design and analysis including criteria for selecting data structures to fit their applications. Knowl

('CSCI2275', 4, 'Combines the content in CSCI 1300 and CSCI 2270 and is intended for students with experience with at least one object oriented programming language. Assumes knowledge of programming constructs- data types, conditionals, loops and classes. Students must pass a programming competency exam administered by the

('CSCI2400', 4, 'Covers how programs are represented and executed by modern computers, including low-level machine representations of programs and data, an understanding of how computer components and the memory hierarchy influence performance.', 'Computer Systems', 'DataStruct', 'Systems', 0, 4, 4),

('CSCI2750', 3, 'Satisfies the ethics requirement for computer science BA and BS majors. This course is intended to provide students with perspectives which help them deal with ethical and societal implications in their careers as computing professionals. Examines ethical ramifications of current and future computing syste

('CSCI2820', 3, 'Introduces the fundamentals of linear algebra in the context of computer science applications.  Includes vector spaces, matrices, linear systems, and eigenvalues.  Includes the basics of floating point computation and numerical linear algebra.', 'Linear Algebra with Computer Science Applications', 'DataStru

('CSCI2824', 3, 'Covers foundational materials for computer science that is often assumed in advanced courses. Topics include set theory, Boolean algebra, functions and relations, graphs, propositional and predicate calculus, proofs, mathematical induction, recurrence relations, combinatorics, discrete probability. Focuses

('CSCI2897', 3, 'Master practical mathematical techniques for representing and analyzing biological quantities of different kinds. Develop mathematical intuition about biological calculations. Learn to model and solve simple feedback processes. Learn to model and solve simple accumulation processes. Learn to model and decom

('CSCI3002', 4, 'Introduces the practice and research of human-computer interaction, including its history, theories, the techniques of user-centered design, and the development of interactive technologies. Covers computing in society at large with respect to domains such as health, education, assistive technology, ethics,

('CSCI3010', 3, 'Explores concepts and techniques for design and construction of larger, reliable, and maintainable software systems in the context of object-oriented programming. Covers various topics including: object-oriented programming paradigms, scope, inheritance, program structure and design, practical use of versio

('CSCI3022', 3, 'Introduces students to the tools methods and theory behind extracting insights from data. Covers algorithms of cleaning and munging data, probability theory and common distributions, statistical simulation, drawing inferences from data, and basic statistical modeling.', 'Introduction to Data Science with Pr

('CSCI3090', 3, 'Covers the basics of quantum computation, including the basics of quantum information; axioms of quantum mechanics; quantum circuits and universality; the relationship between quantum and classical complexity classes; simple quantum algorithms such as the quantum Fourier transform; Shor factoring algorithm;

('CSCI3100', 3, 'Provides students with an understanding of the professional, ethical, legal and social issues and responsibilities of software developers, as well as providing them with the ability to analyze the local and global impacts of computing on individuals, organizations and society.  Required for, and restricted

('CSCI3104', 4, 'Covers the fundamentals of algorithms and various algorithmic strategies, including time and space complexity, sorting algorithms, recurrence relations, divide and conquer algorithms, greedy algorithms, dynamic programming, linear programming, graph algorithms, problems in P and NP, and approximation algori

('CSCI3112', 1, 'Supports students in developing professional skills and practices in computing, including: preparing for technical and behavioral interviews, professional networking, mastering new technologies not addressed in the curriculum, presenting work, the role of graduate study, and exploring career and research di

('CSCI3150', 3, 'Focusing on the concepts of universal design and Web Standards, this course will address issues that occur at the nexus of web standards, Universal Design and the needs of persons with disabilities. Students will gain the expertise and skills to create media and web sites which are accessible, usable and ef

('CSCI3155', 4, 'Studies principles governing the design and analysis of programming languages and their underlying execution models. Explores values, scoping, recursion, higher-order functions, type systems, control structures, and objects. Introduces formal semantics as a framework for understanding programming features.

('CSCI3202', 3, 'Surveys artificial intelligence techniques of search, knowledge representation and reasoning, probabilistic inference, machine learning, and natural language. Knowledge of Python strongly recommended.', 'Introduction to Artificial Intelligence', 'DataStruct,Discrete,Probability,DataSci', NULL, 3, 0, 3),

('CSCI3287', 3, 'Introduces the fundamental concepts of database requirements analysis, database design, and database implementation with emphasis on the relational model and the SQL programming language. Introduces the concepts of Big Data and NoSQL systems.', 'Design and Analysis of Database Systems', 'DataStruct', NULL,

('CSCI3302', 3, 'Introduces students to fundamental concepts in autonomous robotics: mechanisms, locomotion, kinematics, control, perception and planning. Consists of lectures and lab sessions that are geared toward developing a complete navigation stack on a miniature mobile robotic platform.', 'Introduction to Robotics',

('CSCI3308', 3, 'Covers tools and techniques for successful software development with a strong focus on best practices used in industry. Students work in small teams to complete a semester-long application development project. Students learn front-end design and construction using HTML & CSS, back-end database design and co

('CSCI3352', 3, 'This course examines the computational representation and analysis of biological phenomena through the structure and dynamics of networks, from molecules to species. Attention focuses on algorithms for clustering network structures, predicting missing information, modeling flows, regulation, and spreading-p

('CSCI3403', 4, 'Introduces core concepts in cybersecurity including confidentiality, integrity, authentication, risk management, and adversarial thinking. The concepts will be applied to both traditional information technology (IT) systems and cyber physical systems (CPS). At the conclusion of the course students should ha

('CSCI3434', 3, 'Introduces the foundations of formal language theory, computability, and complexity. Shows relationship between automata and various classes of languages. Addresses the issue of which problems can be solved by computational means, and studies complexity of solutions.', 'Theory of Computation', 'Algorithms',
```

Database Creation:

```
-- -----------------------------------------------------
-- Table courseregistry
-- -----------------------------------------------------
DROP TABLE IF EXISTS courseregistry;

CREATE TABLE IF NOT EXISTS courseregistry (
  ClassCode CHAR(8),
  CreditHours INT NOT NULL,
  Description TEXT NOT NULL,
  Name VARCHAR(200) NOT NULL,
  PreRecs TEXT NULL,
  CountsFor VARCHAR(45) NULL,
  Core INT NOT NULL,
  Foundational INT NOT NULL,
  ComputerScience INT NOT NULL
);

-- -----------------------------------------------------
-- Table degreerequirements
-- -----------------------------------------------------
DROP TABLE IF EXISTS degreerequirements;

CREATE TABLE IF NOT EXISTS degreerequirements (
  MajorCode SERIAL PRIMARY KEY,
  TotalCredits INT NOT NULL,
  Requirements TEXT NOT NULL
);

-- -----------------------------------------------------
-- Table user
-- -----------------------------------------------------
DROP TABLE IF EXISTS users;

CREATE TABLE IF NOT EXISTS users (
  userId SERIAL PRIMARY KEY,
  username VARCHAR(200),
  email VARCHAR(200),
  password BYTEA,
  degree VARCHAR(45),
  major VARCHAR(45),
  majorCode INT
);
```

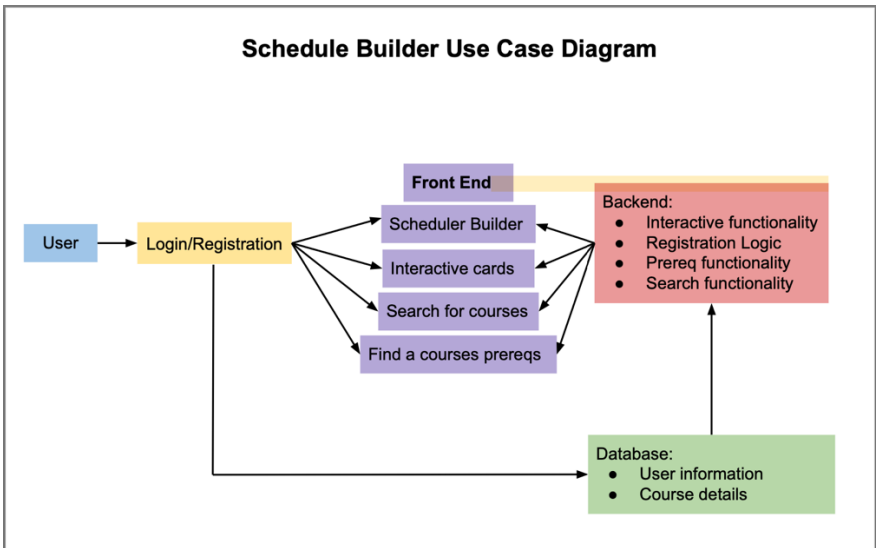
Course Information:



Department	CourseCode	Credit Hours	Name	Description	Counts For	PreRecs (By nar	ComputerScienc	Core	Foundational
CSCI	1000	1	Computer Scien	Introduces curric	NULL	NULL	1	0	1
CSCI	1200	3	Introduction to C	Teaches comput	NULL	NULL	3	0	0
CSCI	1300	4	Computer Scien	Teaches techniq	StartingComputi	Calc1	4	0	4
CSCI	2270	4	Computer Scien	Studies data abs	DataStruct	StartingComputi	4	0	4
CSCI	2275	4	Programming an	Combines the cc	DataStruct	Calc1	4	0	4
CSCI	2400	4	Computer Syste	Covers how proc	Systems	DataStruct	4	0	4
CSCI	2750	3	Computing, Ethi	Satisfies the ethi	Ethics	NULL	3	0	0
CSCI	2820	3	Linear Algebra w	Introduces the fu	Linear	DataStruct,Calcu	3	0	0
CSCI	2824	3	Discrete Structu	Covers foundatic	Discrete	StartingComputi	3	0	0
CSCI	2897	3	Calculating Biolc	Master practical	NULL	Calc1	3	0	0
CSCI	3002	4	Fundamentals o	Introduces the pi	HCI	DataStruct	4	4	0
CSCI	3010	3	Intensive Progra	Explores concep	NULL	DataStruct,Softw	3	0	0
CSCI	3022	3	Introduction to D	Introduces stude	DataSci	DataStruct,Calc2	0	0	0
CSCI	3090	3	Introduction to Q	Covers the basic	NULL	Linear	3	0	0
CSCI	3100	1	Software and Sc	Provides studen	NULL	NULL	1	0	0
CSCI	3104	4	Algorithms	Covers the funda	Algorithms	DataStruct,Calc2	4	0	4
CSCI	3112	1	Professional Dev	Supports studen	NULL	DataStruct	1	0	0
CSCI	3150	3	Universal Desigr	Focusing on the	NULL	StartingComputi	3	0	0
CSCI	3155	4	Principles of Pro	Studies principle	PPL	DataStruct,Syste	4	0	4
CSCI	3202	3	Introduction to A	Surveys artificial	NULL	DataStruct,Discr	3	3	0
CSCI	3287	3	Design and Anal	Introduces the fu	NULL	DataStruct	3	3	0
CSCI	3302	3	Introduction to R	Introduces stude	Robotics	DataStruct,Discr	3	3	0
CSCI	3308	3	Software Develo	Covers tools anc	SoftwareDev	DataStruct	3	0	3
CSCI	3352	3	Biological Netwo	This course exar	NULL	DataStruct,Calc	3	0	0
CSCI	3403	4	Introduction to C	Introduces core	NULL	Systems	4	4	0
CSCI	3434	3	Theory of Comp	Introduces the fc	NULL	Algorithms	3	3	0
CSCI	3593	3	Computer Organ	Studies compute	Systems	Systems	3	0	0
CSCI	3656	3	Numerical Comp	Covers develop	NumericalCompu	StartingComputi	3	3	0
CSCI	3702	3	Cognitive Scienc	Introduces cogni	NULL	DataStruct	3	0	0
CSCI	3753	4	Design and Anal	Analyzes the sof	OS	DataStruct,Syste	4	4	0
CSCI	3832	3	Natural Languag	Explores the the	NULL	DataStruct,Discr	3	0	0
CSCI	4022	3	Advanced Data I	Introduces stude	NULL	Linear, DataSci,A	3	3	0
CSCI	4113	3	Linux System Ac	Introduces Linux	NULL	OS	3	0	0
CSCI	4114	3	Practical Algorit	When coming ac	NULL	Algorithms	3	0	0
CSCI	4118	3	Software Engine	Learn the core p	NULL	StartingComputi	3	0	0
CSCI	4122	3	Information Visu	Studies interacti	NULL	DataStruct,Discr	3	0	0
CSCI	4133	3	Fundamentals o	Practice thinking	NULL	DataStruct,Syste	3	0	0

USE CASE DIAGRAM

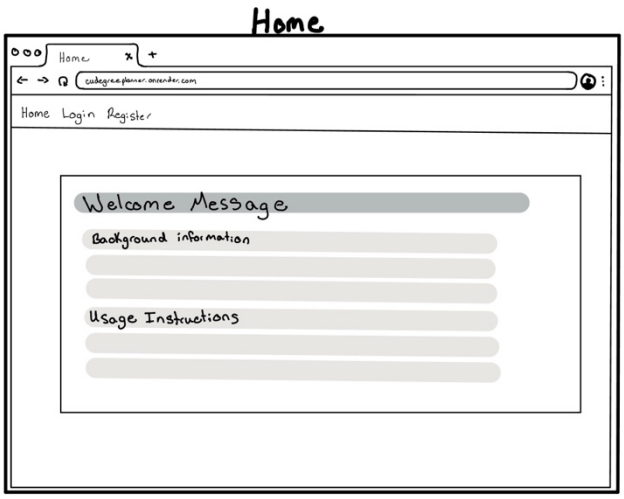
The following is an example of the mockup wireframes we used for the website design.



WIREFRAMES

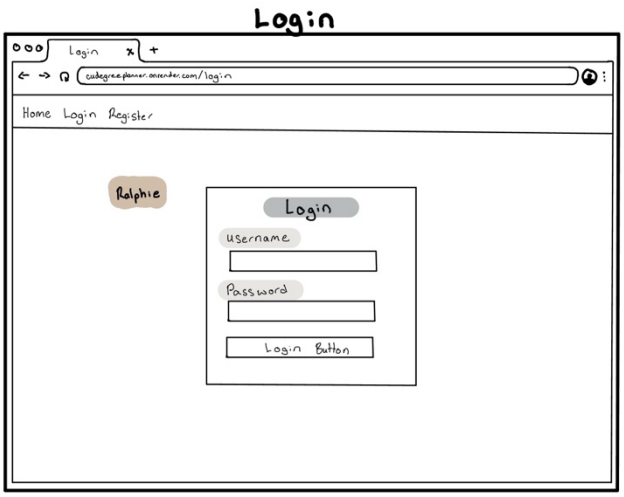
The following is an example of the mockup wireframes we used for the website design. They are all the final-ish designs we used.

Home Page

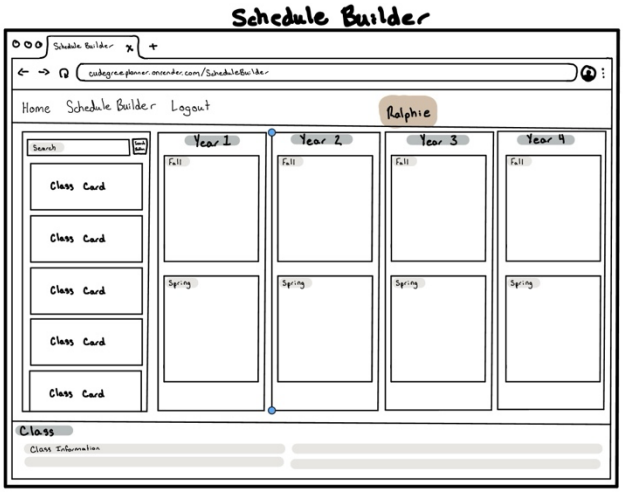


Register Page

Login Page



Schedule Builder



TEST RESULTS

The following is a summary of results and observations made by our end testing users. The test users themselves were all undergrad CU students studying computer science in some capacity.

Results

i. Overall, the application was received very well by all the students who tested it. The highlighting of pre-reqs and co-reqs for classes was very popular and received praise over its functionality. The website usage was simple and easy for the users to follow. Ralphie was a favorite and most users commented on their addition. On the basic level, many users felt like the application was functioning well. All users had suggestions concerning further development and what could be done to improve the site. Small bugs were found and noted for further addressing.

Observations

i. Users mainly focused on dragging and dropping classes in different orders to test the pre-req highlighting. All users also tested the search functionality and what phrases and codes could be used to return a result. Most users seemed focused on bug testing and seeing if they could 'break' the application in any way. In all cases but one, the users behaved as expected and the application responded correctly. The one case that was unaccounted for was when a user tried to use a password manager to create a password for the site and the login functionality would not accept it. This requires further debugging.

DEPLOYMENT

Link to Render Hosted Application: https://cudegreeplanner.onrender.com

*It will take a little bit for the page to load initially, shouldn't be more than a minute or so.