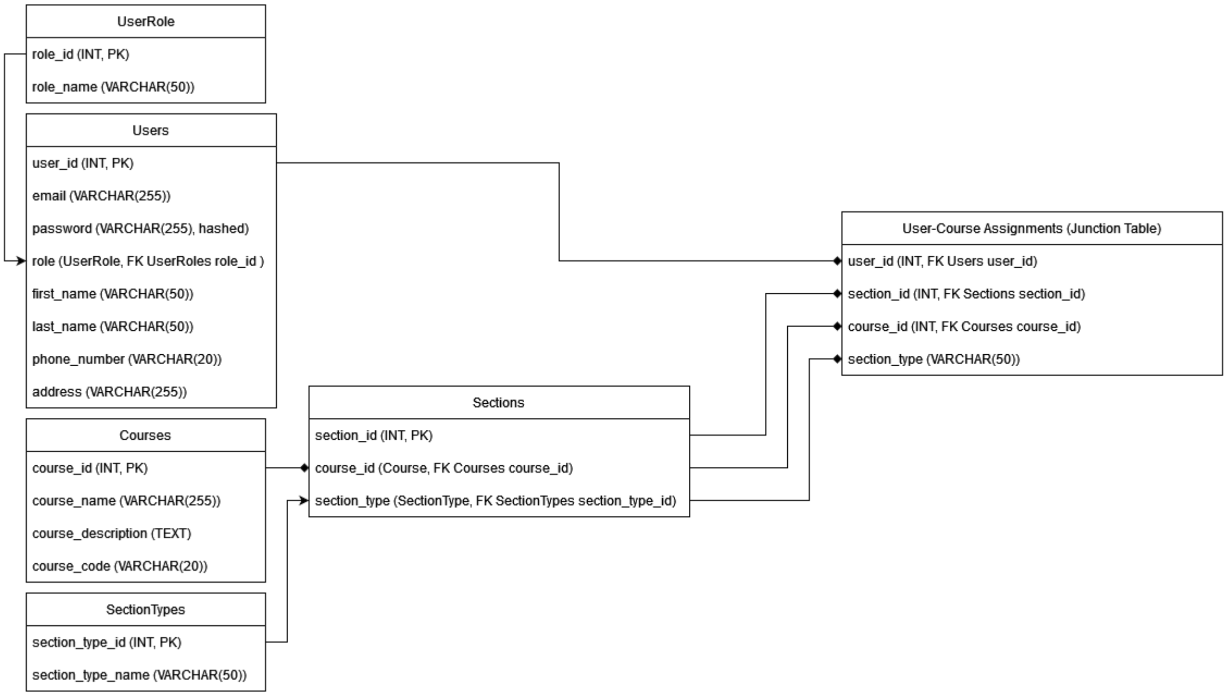
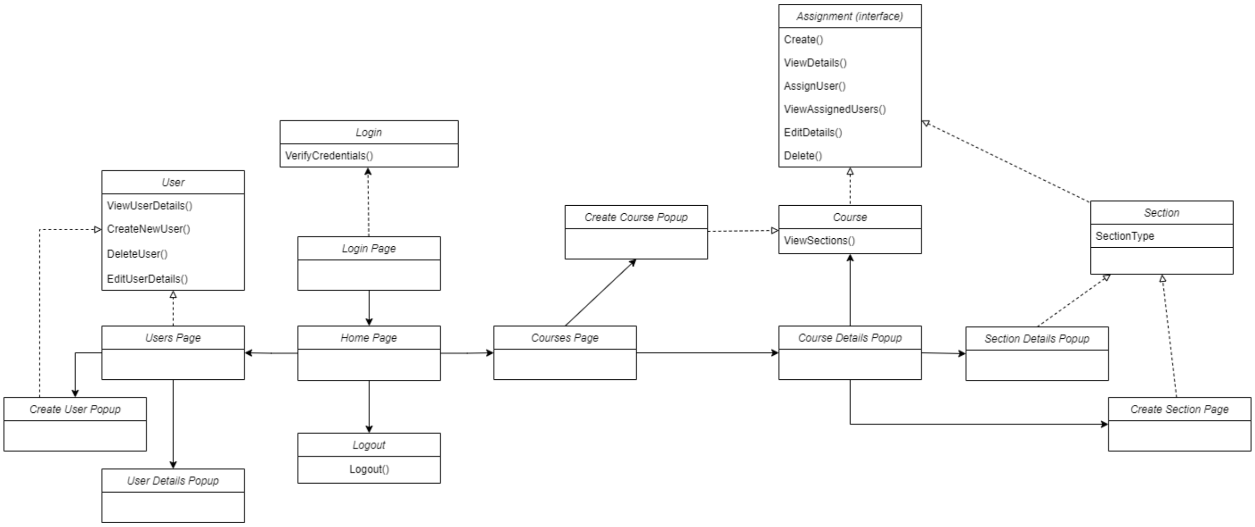
Team supercreative - Design Document

Database Design


UserRole
role_id (INT, PK)
role_name (VARCHAR(50))
Users
user_id (INT, PK)
email (VARCHAR(255))
password (VARCHAR(255), hashed)
role (UserRole, FK UserRoles role_id)
first_name (VARCHAR(50))
last_name (VARCHAR(50))
phone_number (VARCHAR(20))
address (VARCHAR(255))
User-Course Assignments (Junction Table)
user_id (INT, FK Users user_id)
section_id (INT, FK Sections section_id)
course_id (INT, FK Courses course_id)
section_type (VARCHAR(50))
Courses
course_id (INT, PK)
course_name (VARCHAR(255))
course_description (TEXT)
course_code (VARCHAR(20))
Sections
section_id (INT, PK)
course_id (Course, FK Courses course_id)
section_type (SectionType, FK SectionTypes section_type_id)
SectionTypes
section_type_id (INT, PK)
section_type_name (VARCHAR(50))

Software Design Diagram



Method Descriptions

authentication.py

authentication.did_login(p1, p2)
Preconditions: - User has entered a email and a password
Postconditions: - Create session session keys for user data (email, role)
Side-effects: - Creates session
p1: request input
p2: boolean output

authentication.did_logout(p1, p2)
Preconditions: - User requested to logout
Postconditions: - Deletes session keys
Side-effects: - Ends session
p1: request input
p2: boolean output

authentication.active_session_exists(p1, p2)
Preconditions: None
Postconditions: None
Side-effects:
p1: request input
p2: boolean output

authentication.end_session(p1)
Preconditions: - An active session exists
Postconditions: - There is no longer an active session
Side-effects: - Current session is ended - Session keys are deleted
p1: session input

user.py

User.create_user(p1, p2, p3, p4, p5, p6, p7, p8)
Preconditions: - Email is a non-null and properly formatted string (i.e. xyz@uwm.edu) - Password is non-null string and contains each of the following: lowercase, uppercase, number, and special characters - Role is a non-null string and Administrator, Instructor, or TA - First Name is a non-null string - Last Name is a non-null string - Phone Number is non-null string of 10 digits (remove (), - and whitespace) - Address is a non-null string
Postconditions: - Return true if the user information was valid, create an entry in the Users table with the provided information, and go to the User Details Page for the new user. - Return false, display error message that invalid information was provided, and prompt user to re-enter information. (i.e. user already exists or data was input in bad format)
Side-effects: New entry in User table, provided UserID is not usable for any other employee
p1: email input
p2: password input
p3: role input
p4: first input
p5: last input
p6: phone input
p7: address input

p8: string output

User.delete_user(p1, p2)

Preconditions:

- A user selects (clicks on) the Delete User button while viewing the User Details Page of a user from the Users table.

Postconditions:

- Return a true if the selected User account was deleted, remove all entries from the User-Course Assignments table that have the deleted UserID,
- and return to the Users Page
- Return false and display an error message if active user role does is not Administrator
- Return false and go to Users Page if target userID does not exist in Users table

Side-effects: none

p1: userID of target user input

p2: boolean output

User.edit_user(p1, p2, p3, p4, p5, p6, p7,p8)

Preconditions:

- Password is non-null string and contains each of the following: lowercase, uppercase, number, and special characters
- Role is a string and “Administrator”, “Instructor”, or “TA”
- Active user’s role is “Administrator”
- First Name is a string and active user’s role is “Administrator”
- Last Name is a string and active user’s role is “Administrator”
- Phone Number is a string of 10 digits (remove (), - and whitespace)
- Address is a string

Postconditions:

- Return false if active user tried to change target user’s role and active user’s role is not administrator display error message indicating insufficient privileges.
- Return false if user does not exist in User’s table go to Users Page and display error that target user could not be found.
- Return false if any information entered was invalid and display error message indicating invalid information.
- Return true and change values for target user’s entry in the Users table.

Side-effects: none.

p1: user_id input

p2: new_password input (uses existing value if input is null)

p3: new_role input (uses existing value if input is null)

P4:new_first input (uses existing value if input is null)
p5: new_last input (uses existing value if input is null)
p6: new_phone input (uses existing value if input is null)
p7: new_address input (uses existing value if input is null)
P8: string output

User.edit_user_with_skills(p1,p2,p3,p4,p5,p6,p7,p8)
Preconditions: - Password is non-null string and contains each of the following: lowercase, uppercase, number, and special characters - Role is a string and “Administrator”, “Instructor”, or “TA” - Active user’s role is “Administrator” - First Name is a string and active user’s role is “Administrator” - Last Name is a string and active user’s role is “Administrator” - Phone Number is a string of 10 digits (remove (), - and whitespace) - Address is a string
Postconditions: - Return false if active user tried to change target user’s role and active user’s role is not administrator display error message indicating insufficient privileges. - Return false if user does not exist in User’s table go to Users Page and display error that target user could not be found. - Return false if any information entered was invalid and display an error message indicating invalid information. - Return false if skills is not a string - Return true and change values, including skills, for the target user’s entry in the Users table.
p1: user_id input
p2: new_password input
p3: new_role input
p4: new_first input
p5: new_last input
p6: new_phone input
p7: new_address input
p8: skills input

p9: string output

course.py

Course.create_course(p1, p2, p3, p4)
Preconditions: - CourseID is a non-null and unique integer - Course Name is a non-null and unique string - Course Description is a non-null string - Course Code is a non-null, unique, and properly formatted string (i.e. COMPSCI-361) - Active user role is Administrator
Postconditions: - Return false if the CourseID, Course Name, or Course Code already exists in the Courses table or Archived Courses table and indicate duplicated value - Return false if active user role is not Administrator - Return true if input was valid, create an entry in the Courses table with the input information, and go to the Course Details Page for the new course.
Side-effects: none
P1: name input
p2: description input
p3:code input
p4: string output

Course.delete_course(p1, p2)
Preconditions: - A user selects (clicks on) the Delete Course button while viewing the Course Details Page of a course from the Courses table.
Postconditions: - Return false and display an error message if active user role is not Administrator - Return false and go to Courses Page if target CourseID does not exist in Courses table - Return true if the target CourseID exists in the Courses table and the course and its sections were successfully deleted
Side-effects: none
p1: course input
p2: boolean output

Course.edit_course(p1,p2,p3,p4,p5)
Preconditions: - An authorized user selects the edit course button while viewing the course details.
Postconditions: - Return false and display an error message if active user role is not Administrator - Return false and go to Courses Page if target CourseID does not exist in Courses table - Return true if the target CourseID exists in the Courses table and the course and its sections were successfully edited
Side-effects: none
p1: current_course_id input
p2: new_course_name input
p3: new_course_description input
p4: new_course_code input
p5: string output

Course.course_id_to_int(p1,p2)
Preconditions: - Course_id is an existing variable
Postconditions: - Returns None if failed - Returns a int version of course_id
p1: course_id input
p2: int output

section.py

section.create_section(p1, p2, p3)
Preconditions: - Course is non-null and exists in the Courses Table - Section Type is a non-null string and is “Lecture”, “Lab”, or “Discussion”
Postconditions:

- An entry is created in the sections table with the given input information
Side-effects: - A section is created
p1: course input
p2: section_type input
p3: string output

section.delete_section(p1, p2, p3)
Preconditions: - Course_id is non-null and corresponds to a course entry in the courses table - Section_id is non-null and corresponds to a section entry in the sections table
Postconditions: - Corresponding user course assignment for that section is deleted - Section itself corresponding to the section_id is deleted
Side-effects: - User course assignment entry deleted - Section entry deleted
p1: course_id input
p2: section_id input
p3: string output

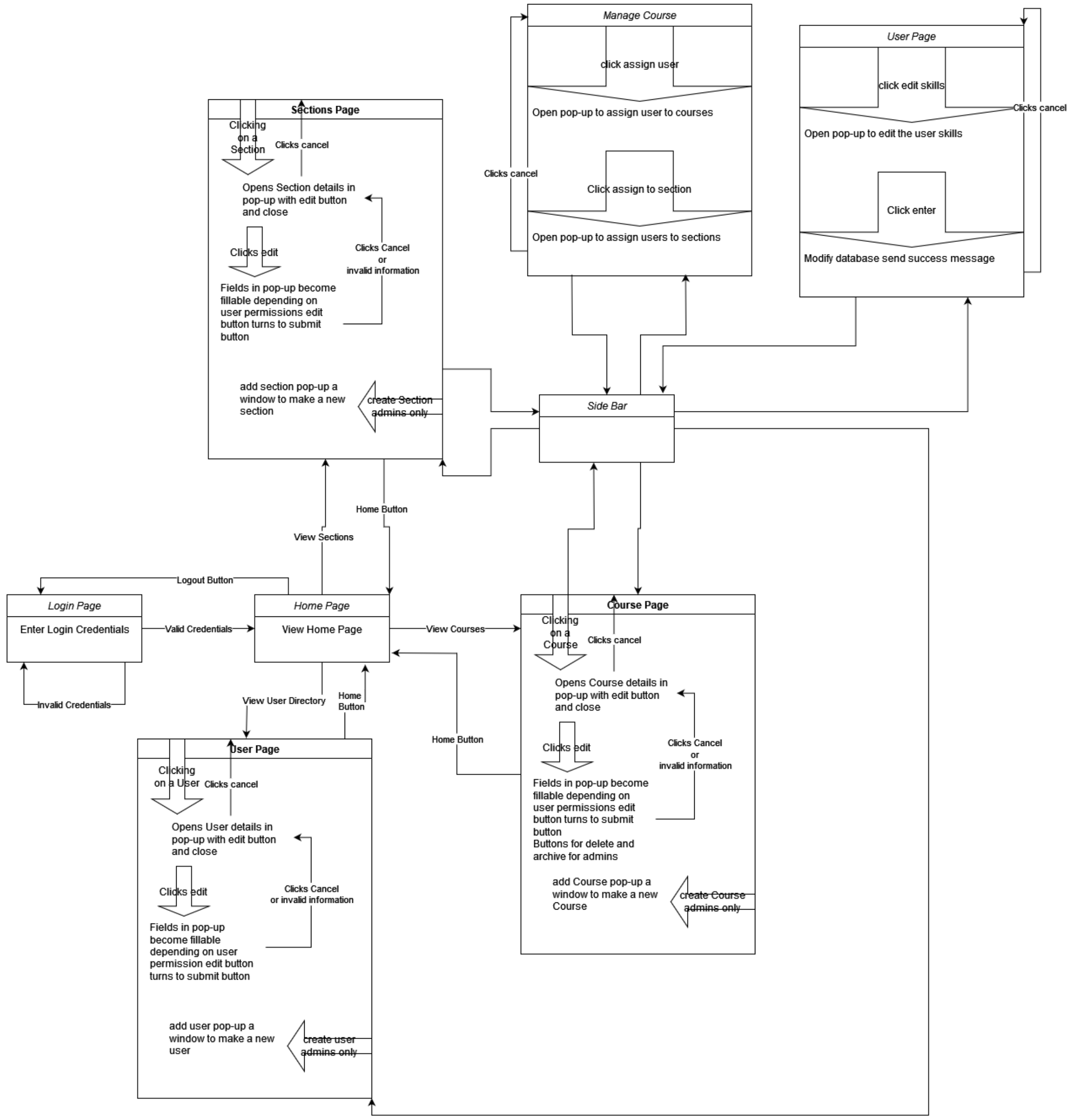
user_assignments.py

user_assignments.assign_user_to(p1, p2, p3, p4)
Preconditions: - Assigned user exists and is in the users table - Assigned course exists and is in the courses table - Assigned section exists in the sections table or is None
Postconditions: - User course assignment is modified if it exists, otherwise a new user course assignment is created
Side-effects: - User course assignment table entry is created or updated
p1: assigned_user input

p2: assigned_course input
p3: assigned_section input
p4: string output

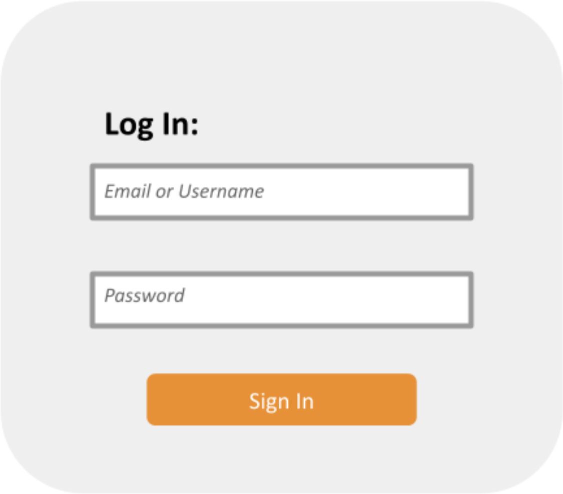
user_assignments.remove_user_from(p1, p2, p3)
Preconditions: - Assigned user exists and is in the users table - Assigned course exists and is in the courses table
Postconditions: - Each user course assignment corresponding to the given user and course is deleted
Side-effects: - User course assignment table entries are deleted
p1: assigned_user input
p2: assigned_course input
p3: string output

State Machine Diagram

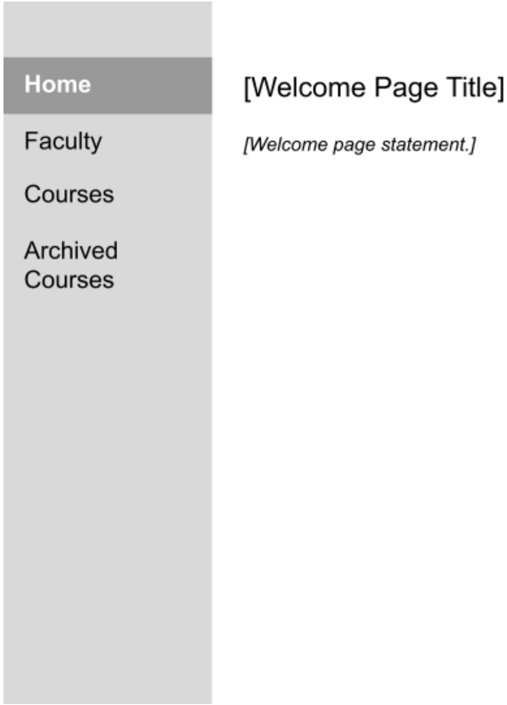

Manage Course
click assign user
Open pop-up to assign user to courses
Click assign to section
Open pop-up to assign users to sections
Clicks cancel
User Page
click edit skills
Clicks cancel
Open pop-up to edit the user skills
Click enter
Modify database send success message
Sections Page
Clicking on a Section
Clicks cancel
Opens Section details in pop-up with edit button and close
Clicks edit
Clicks Cancel or invalid information
Fields in pop-up become fillable depending on user permissions edit button turns to submit button
add section pop-up a window to make a new section
create Section admins only
Side Bar
Home Button
View Sections
Logout Button
Login Page
Enter Login Credentials
Valid Credentials
Invalid Credentials
Home Page
View Home Page
View Courses
View User Directory
Home Button
Home Button
Course Page
Clicking on a Course
Clicks cancel
Opens Course details in pop-up with edit button and close
Clicks edit
Clicks Cancel or invalid information
Fields in pop-up become fillable depending on user permissions edit button turns to submit button
Buttons for delete and archive for admins
add Course pop-up a window to make a new Course
create Course admins only
User Page
Clicking on a User
Clicks cancel
Opens User details in pop-up with edit button and close
Clicks edit
Clicks Cancel or invalid information
Fields in pop-up become fillable depending on user permission edit button turns to submit button
add user pop-up a window to make a new user
create user admins only

Prototypes

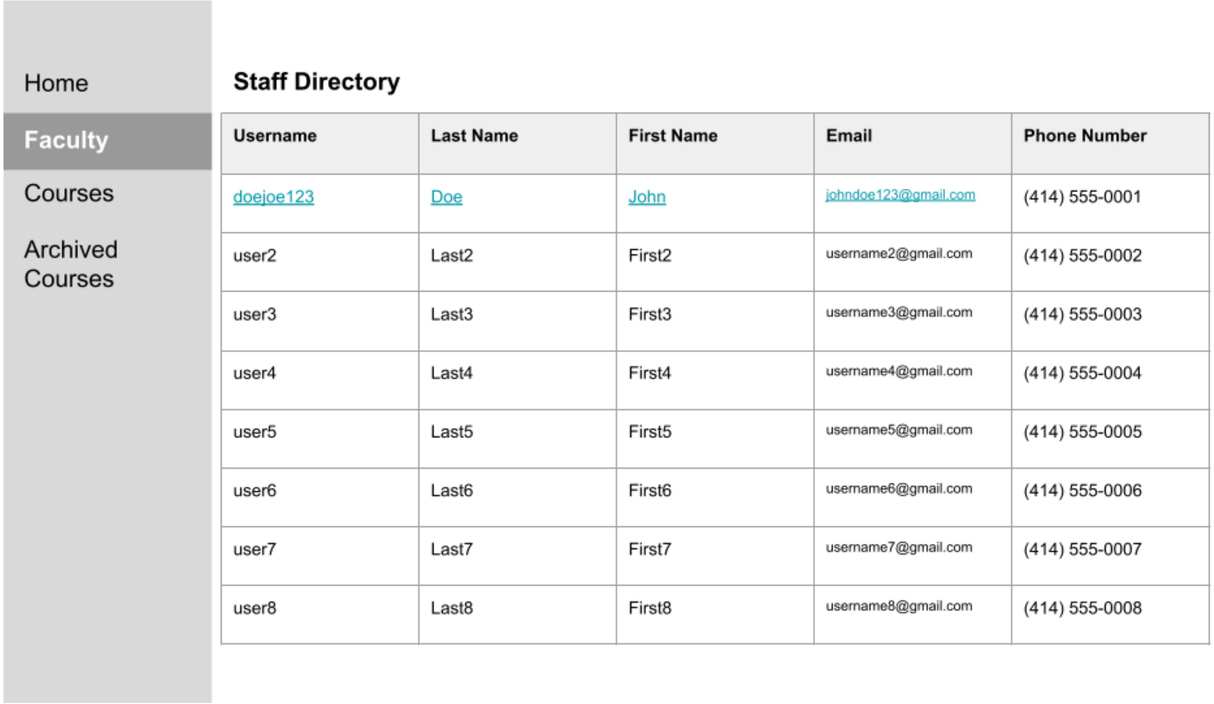
Login Page



Home Page



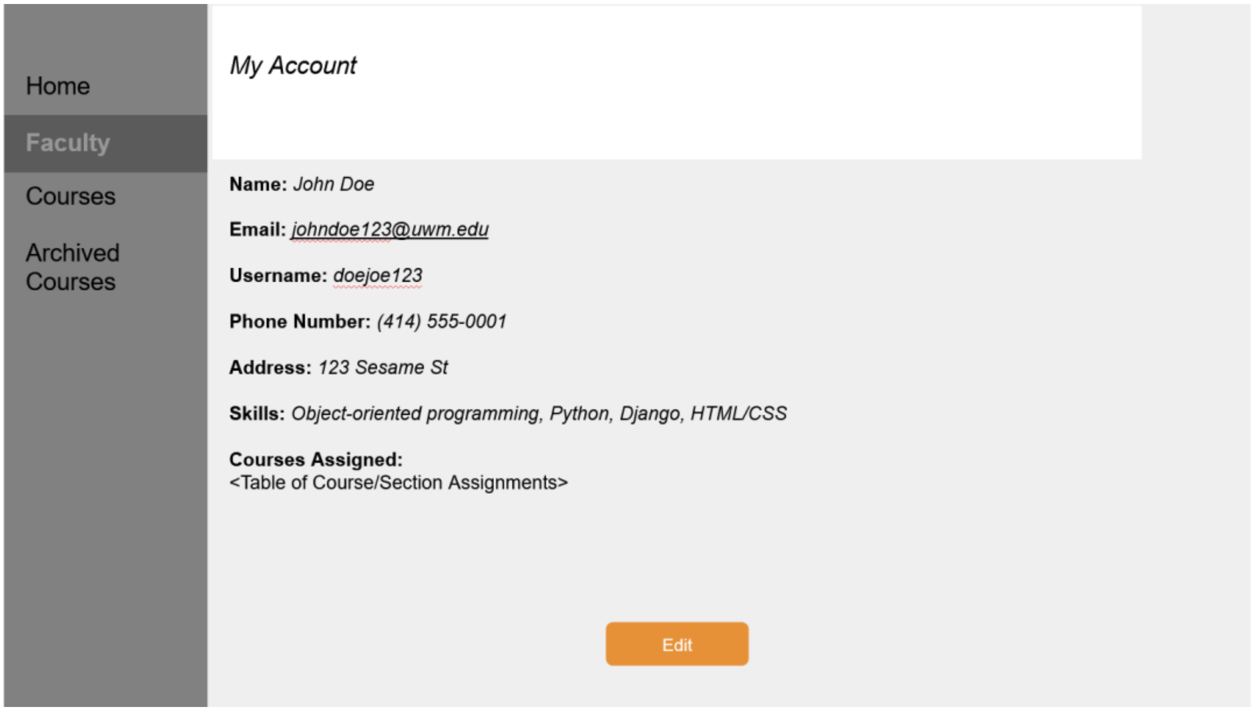
Users Page



Staff Directory

Username	Last Name	First Name	Email	Phone Number
doejoe123	Doe	John	johndoe123@gmail.com	(414) 555-0001
user2	Last2	First2	username2@gmail.com	(414) 555-0002
user3	Last3	First3	username3@gmail.com	(414) 555-0003
user4	Last4	First4	username4@gmail.com	(414) 555-0004
user5	Last5	First5	username5@gmail.com	(414) 555-0005
user6	Last6	First6	username6@gmail.com	(414) 555-0006
user7	Last7	First7	username7@gmail.com	(414) 555-0007
user8	Last8	First8	username8@gmail.com	(414) 555-0008

User Details Page



My Account

Name: *John Doe*

Email: *johndoe123@uwm.edu*

Username: *doejoe123*

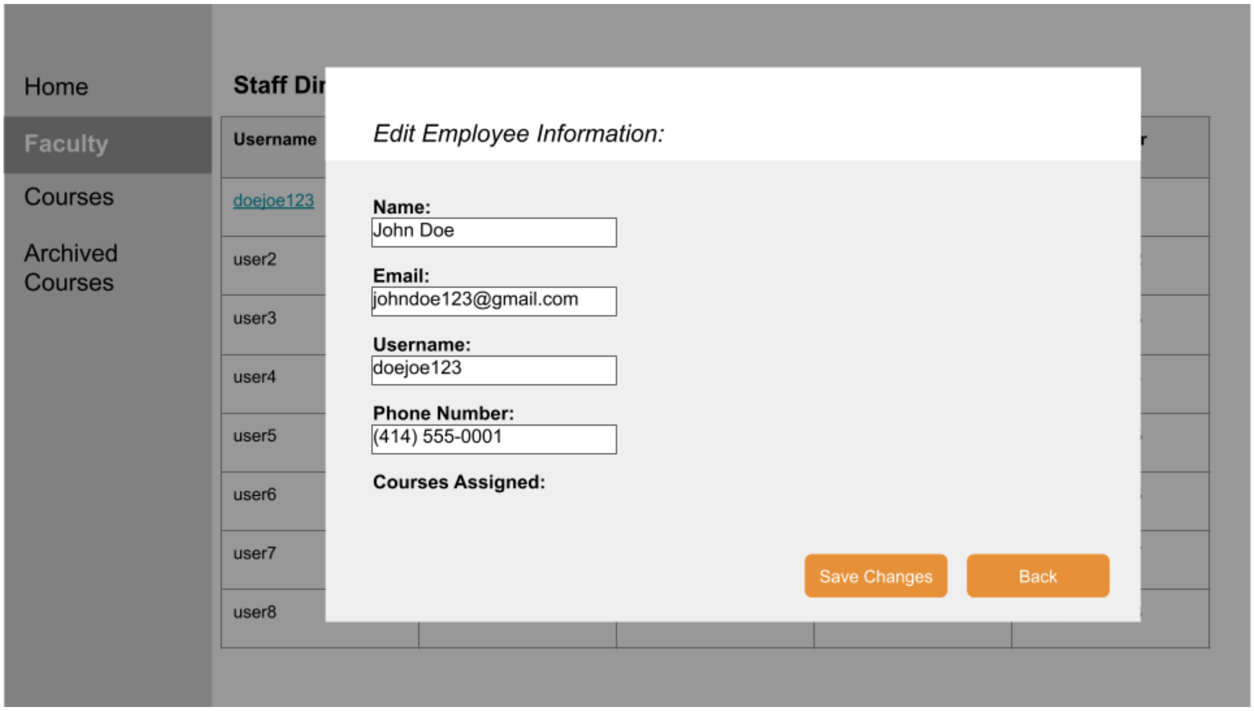
Phone Number: *(414) 555-0001*

Address: *123 Sesame St*

Skills: *Object-oriented programming, Python, Django, HTML/CSS*

Courses Assigned:
<Table of Course/Section Assignments>

Edit User Details Page



Username
doejoe123
user2
user3
user4
user5
user6
user7
user8

Edit Employee Information:

Name: John Doe

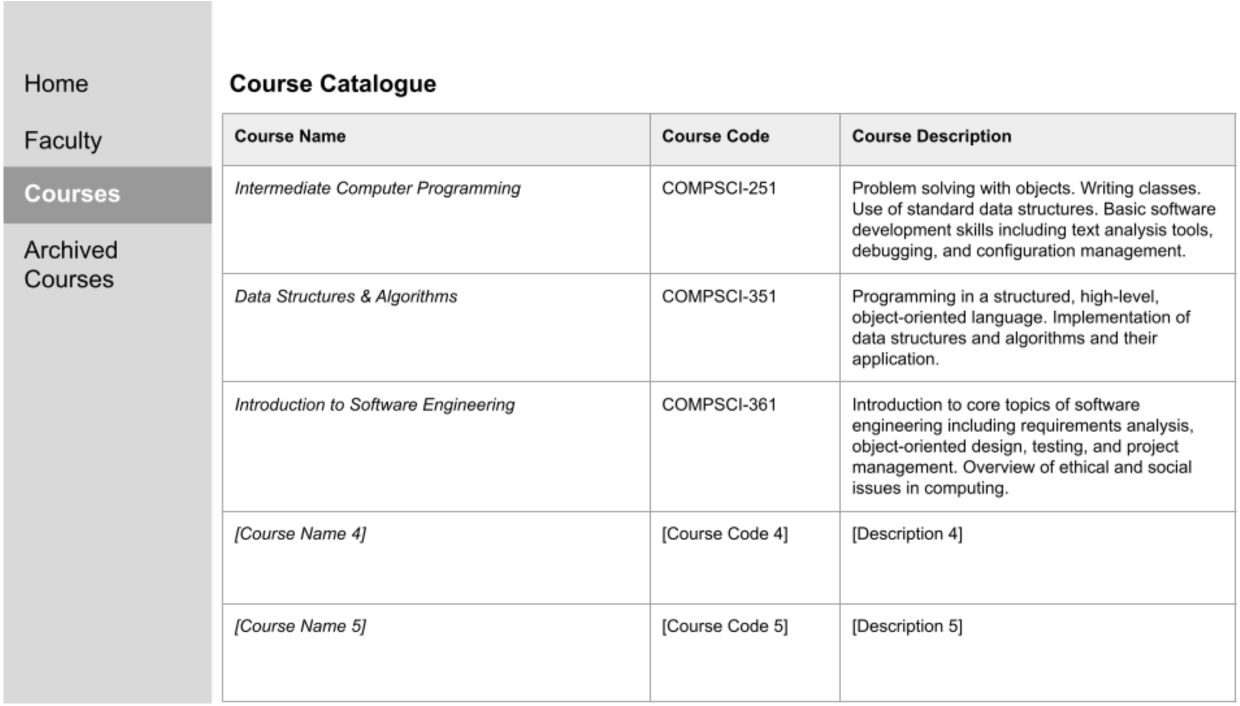
Email: johndoe123@gmail.com

Username: doejoe123

Phone Number: (414) 555-0001

Courses Assigned:

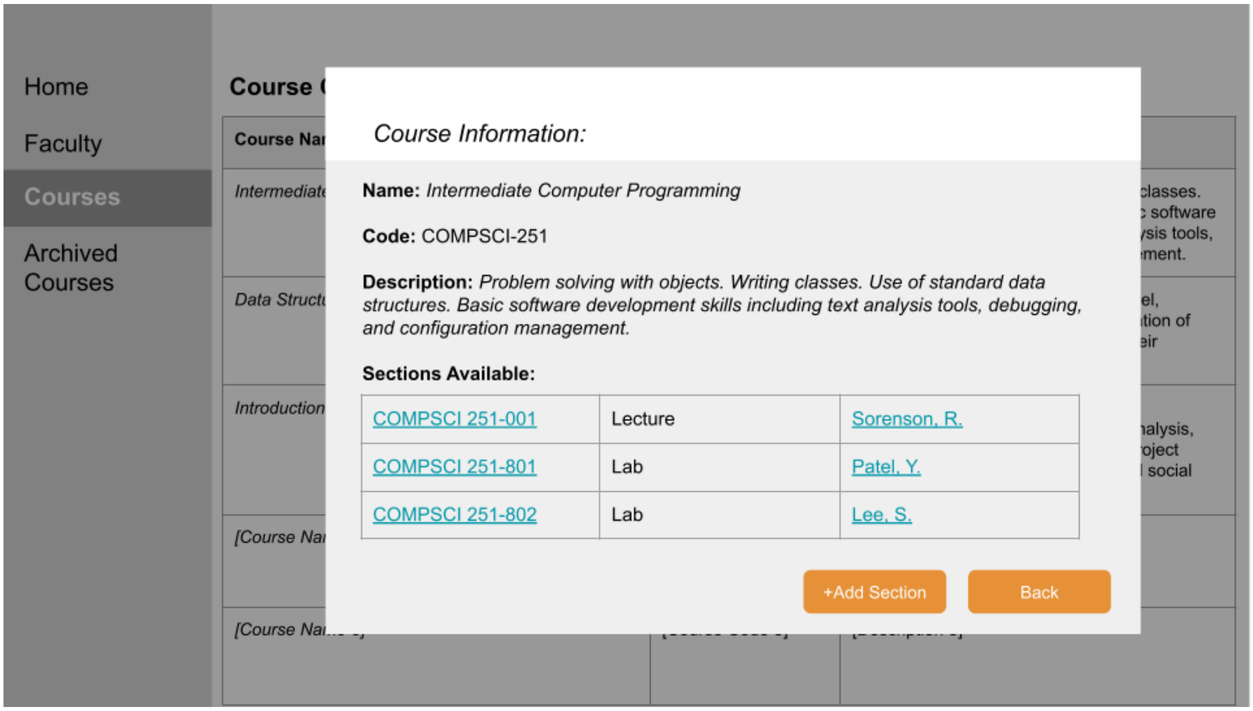
Courses Page



Course Catalogue

Course Name	Course Code	Course Description
Intermediate Computer Programming	COMPSCI-251	Problem solving with objects. Writing classes. Use of standard data structures. Basic software development skills including text analysis tools, debugging, and configuration management.
Data Structures & Algorithms	COMPSCI-351	Programming in a structured, high-level, object-oriented language. Implementation of data structures and algorithms and their application.
Introduction to Software Engineering	COMPSCI-361	Introduction to core topics of software engineering including requirements analysis, object-oriented design, testing, and project management. Overview of ethical and social issues in computing.
[Course Name 4]	[Course Code 4]	[Description 4]
[Course Name 5]	[Course Code 5]	[Description 5]

Course Details Page



Course Information:

Name: *Intermediate Computer Programming*

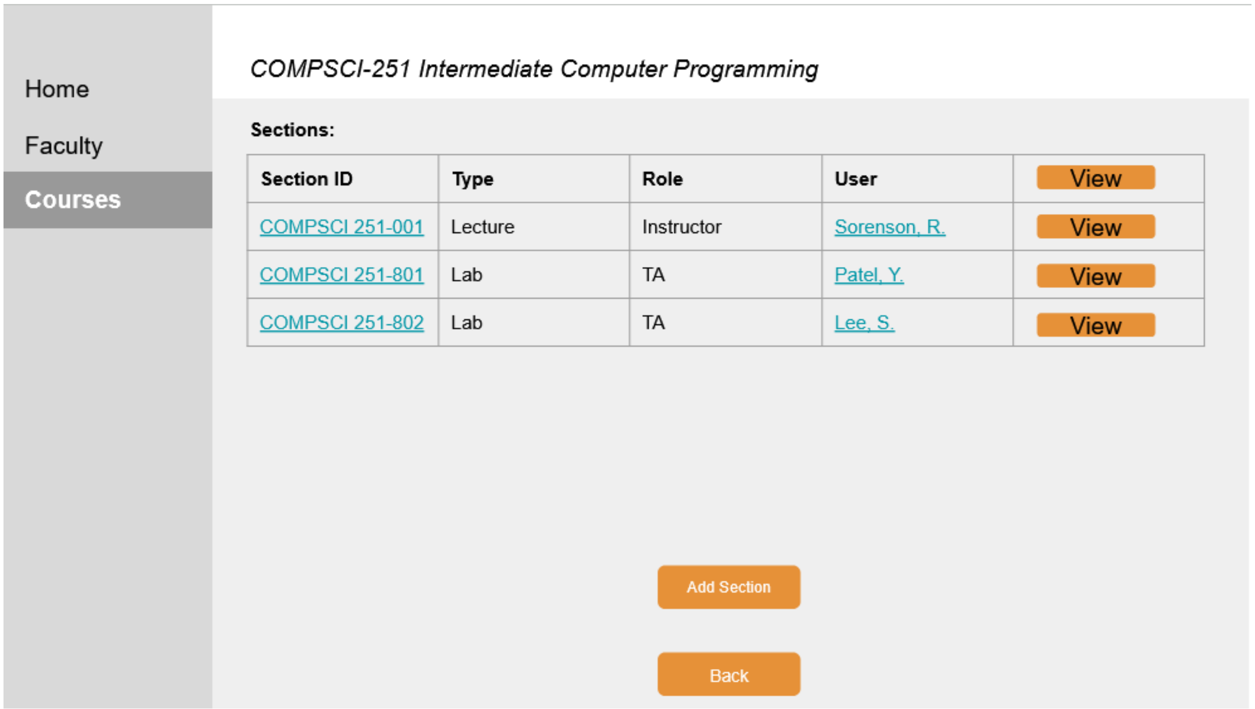
Code: COMPSCI-251

Description: *Problem solving with objects. Writing classes. Use of standard data structures. Basic software development skills including text analysis tools, debugging, and configuration management.*

Sections Available:

COMPSCI 251-001	Lecture	Sorenson, R.
COMPSCI 251-801	Lab	Patel, Y.
COMPSCI 251-802	Lab	Lee, S.

Manage Course Page

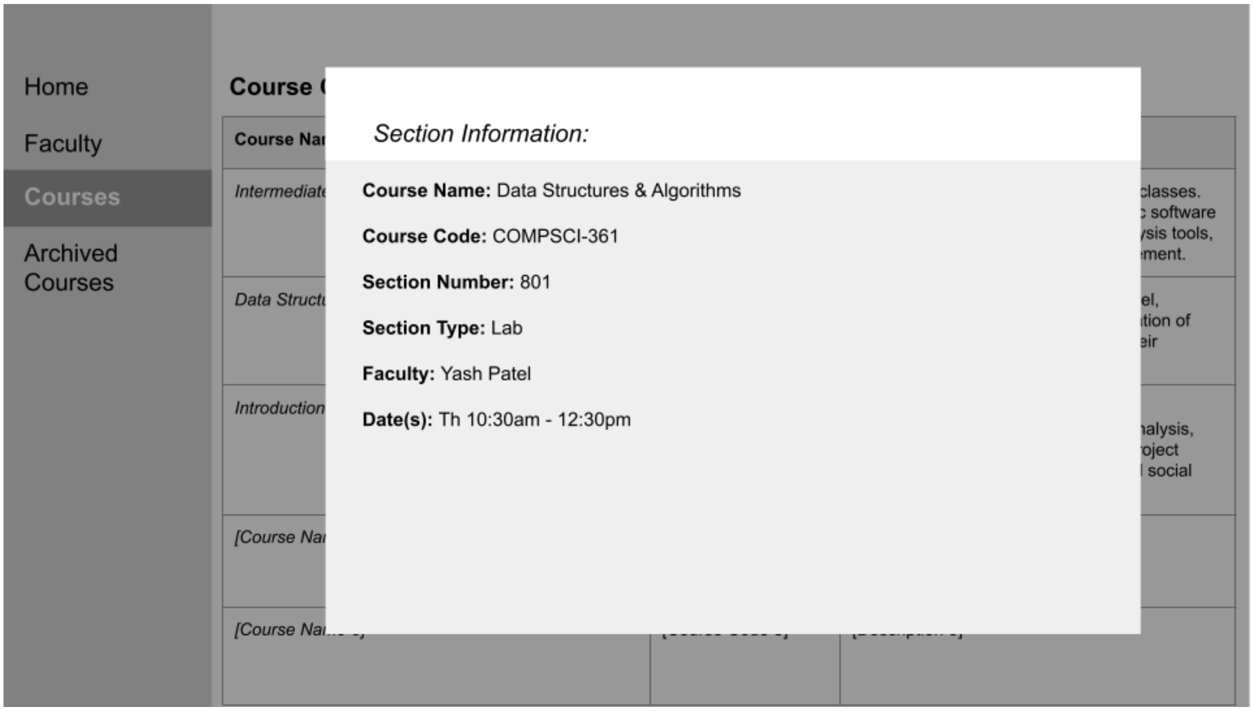


COMPSCI-251 Intermediate Computer Programming

Sections:

Section ID	Type	Role	User	View
COMPSCI 251-001	Lecture	Instructor	Sorenson, R.	View
COMPSCI 251-801	Lab	TA	Patel, Y.	View
COMPSCI 251-802	Lab	TA	Lee, S.	View

Section Details Page



Section Information:

Course Name: Data Structures & Algorithms

Course Code: COMPSCI-361

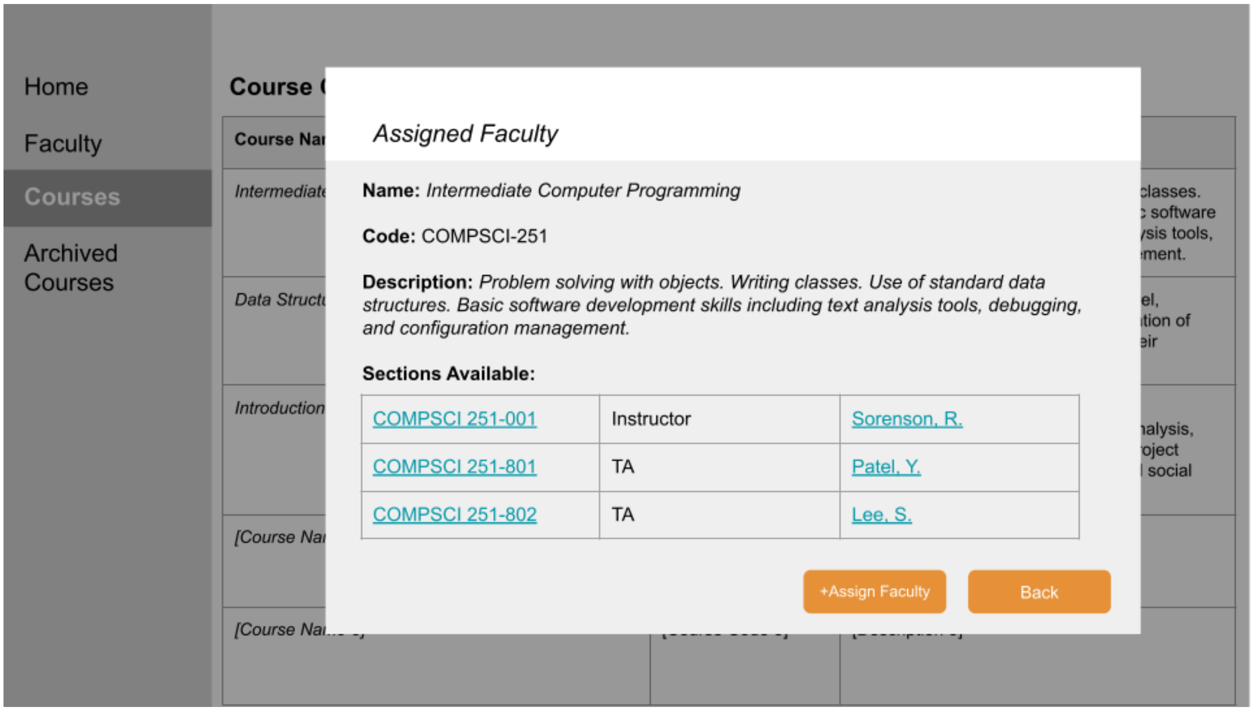
Section Number: 801

Section Type: Lab

Faculty: Yash Patel

Date(s): Th 10:30am - 12:30pm

Assigned Users Page



Assigned Faculty

Name: *Intermediate Computer Programming*

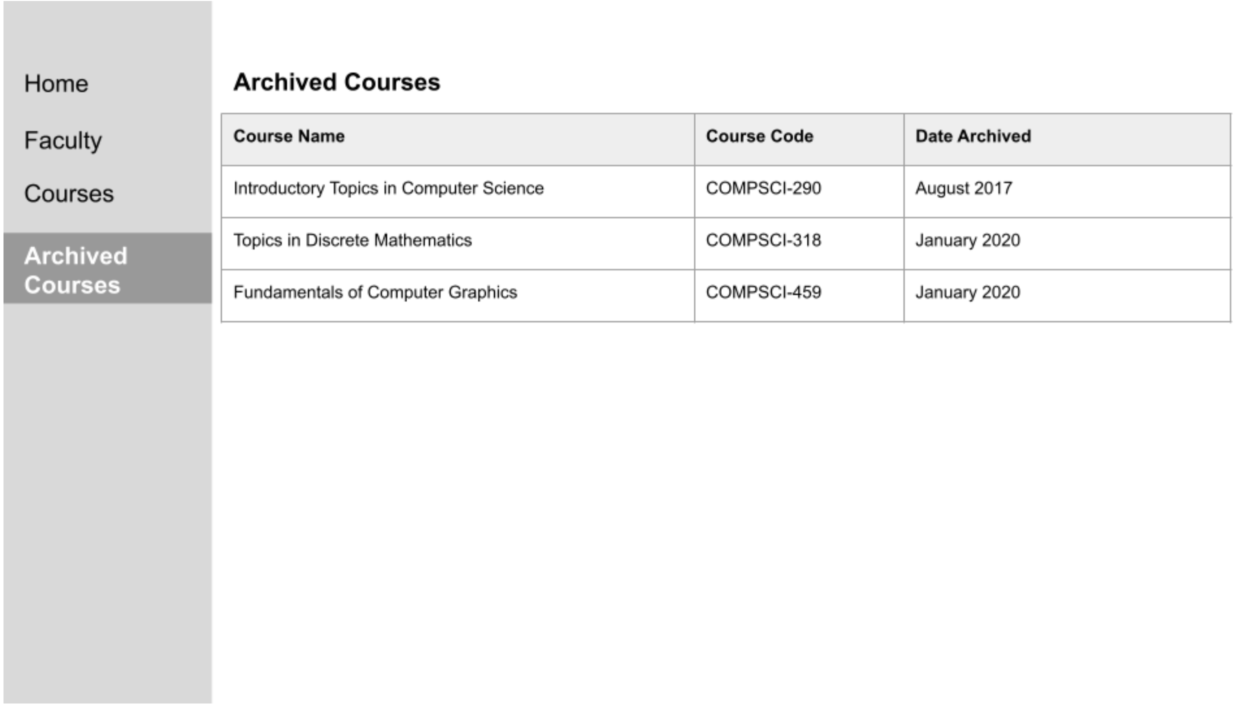
Code: COMPSCI-251

Description: *Problem solving with objects. Writing classes. Use of standard data structures. Basic software development skills including text analysis tools, debugging, and configuration management.*

Sections Available:

COMPSCI 251-001	Instructor	Sorenson, R.
COMPSCI 251-801	TA	Patel, Y.
COMPSCI 251-802	TA	Lee, S.

Archived Courses Page (defunct)



Archived Courses

Course Name	Course Code	Date Archived
Introductory Topics in Computer Science	COMPSCI-290	August 2017
Topics in Discrete Mathematics	COMPSCI-318	January 2020
Fundamentals of Computer Graphics	COMPSCI-459	January 2020