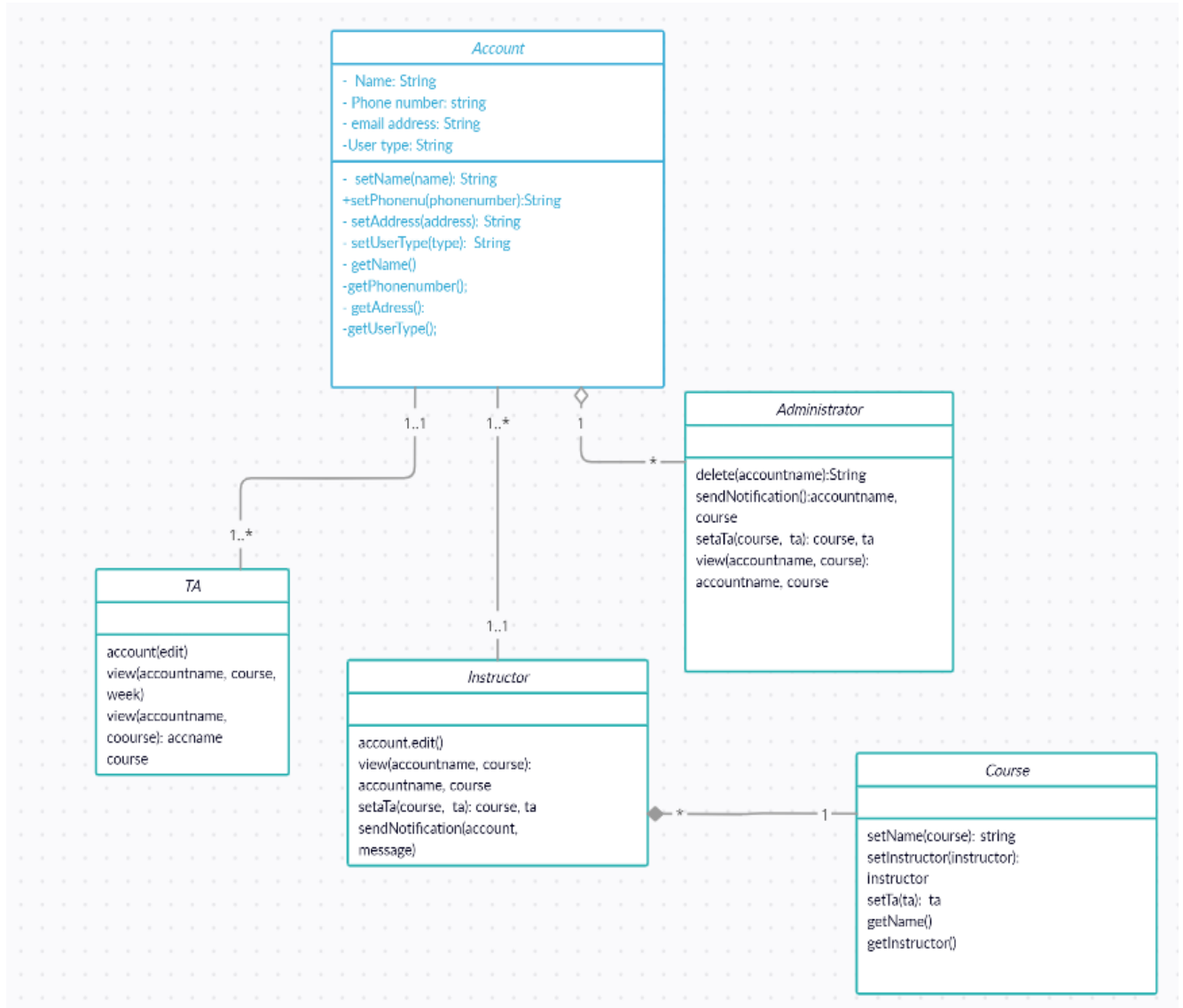
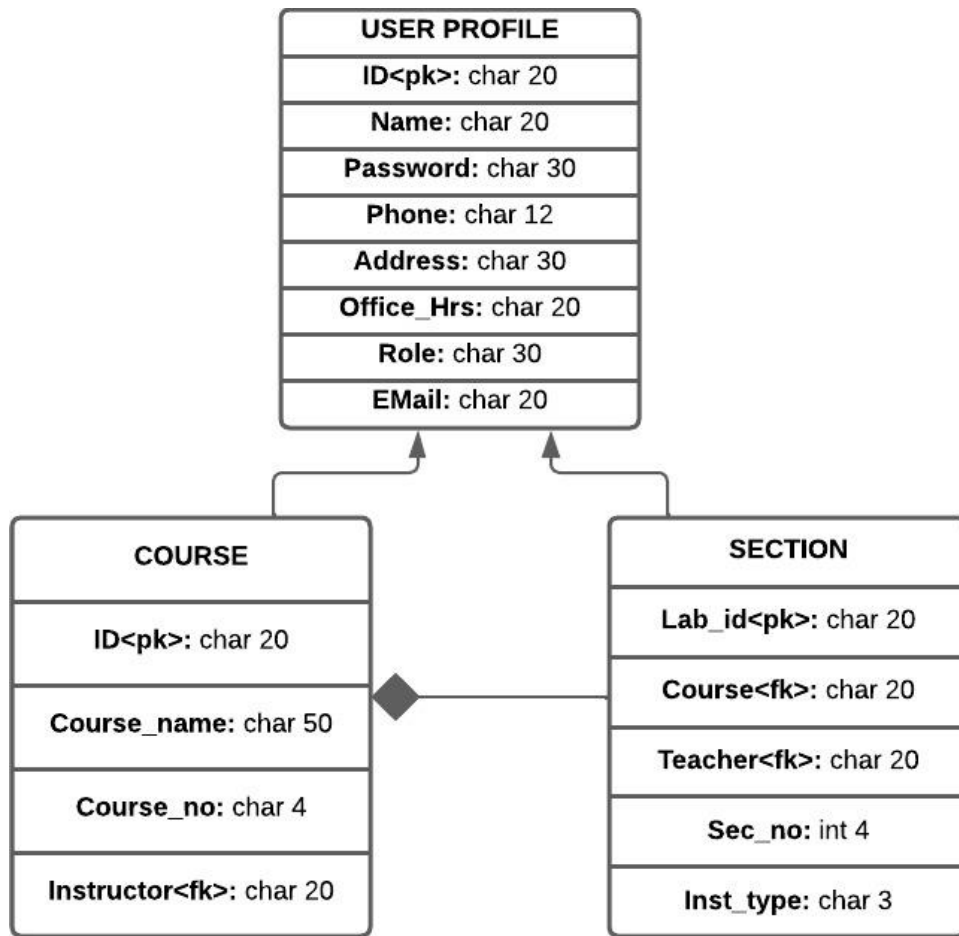


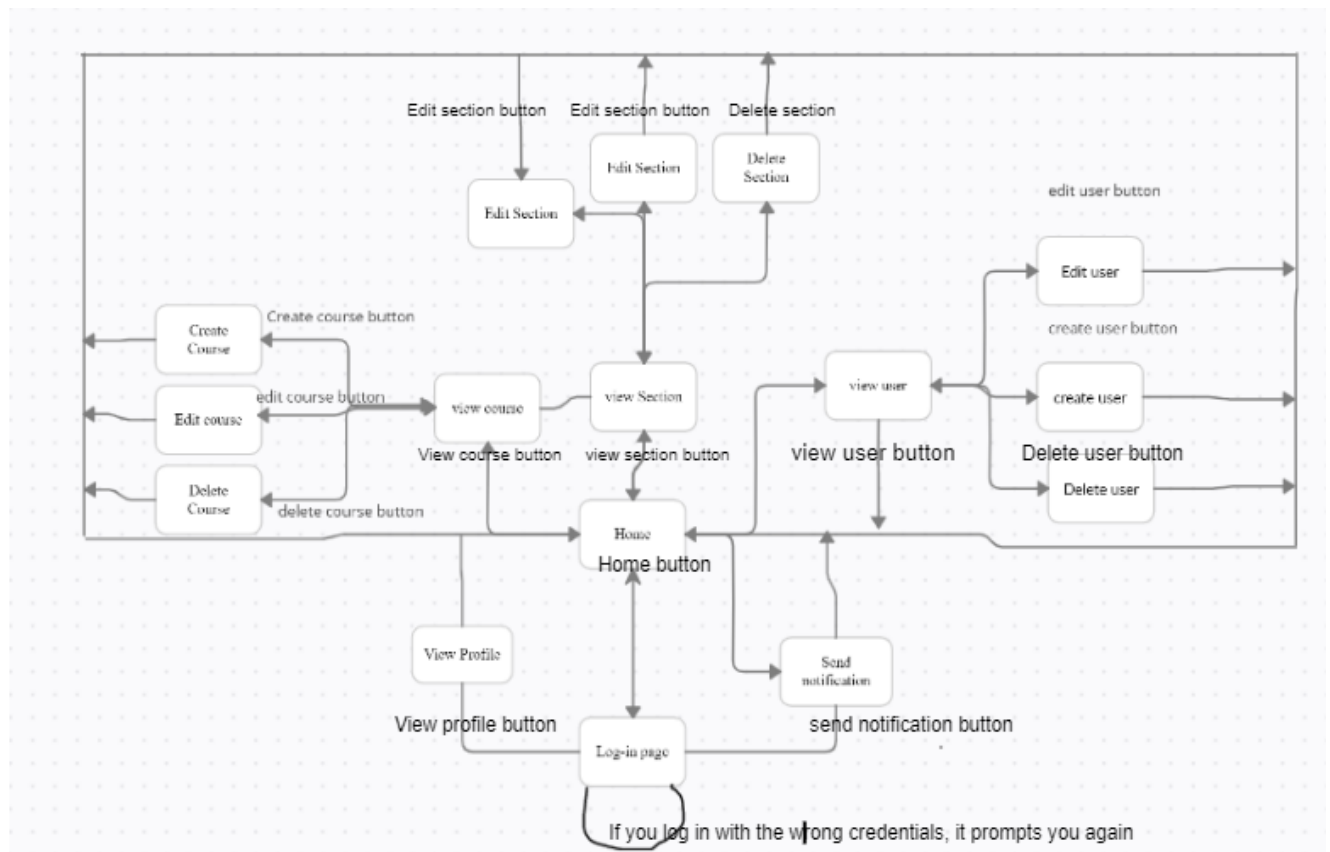
## Class Diagram



*Database Schema:*



## State Machine Diagram:



## Prototypes

# UWM TA Scheduling application

Username:

Password:

Log In

Log in page



## UWM TA Scheduling application

My account

Log out

View users

View courses

View labs

Notify users

Home page



## UWM TA Scheduling application

[Log out](#)[change profile picture](#)

Name: Haitam Chouiekh

Email: [chouiekh@uwm.edu](mailto:chouiekh@uwm.edu)

Phone number: 342780332

Role: Student

Profile page



## UWM TA Scheduling application

[My account](#)[Log out](#)

Search:



New course



Course	Instructor	Location	Hours
CompSci 337	Jay Rock	EMS 420	1:00am - 1:00pm
CompSci 351	John Boyland	Lapham 213	23:00-23:59
CompSci 361	Jayson Rock	EMS 1	Kyle
Infost 1984	Big brother	Online	All the time
Scientology 101	Tom Cruise	Online	12:00pm -

[View courses page](#)



## Create a new Course

New course's name:

New course's Instructor:

New course's location:

New course's time:

Submit

Create course page



## Edit Course

Course name:

Compsci 337

Course Instructor:

Jayson Rock

Course location:

EMS 190

Course time:

MW 11:30am -12:30 am

Save



Edit course page



Are you sure you want to delete this course?  
They will be permanently erased from the  
database.

[Delete course page](#)

Search:

**New Lab**

[View labs page](#)

Lab	Course	Instructor	Location	Hours
802	CompSci 337	Lei Yao	EMS 420	1:00am - 1:00pm
603	CompSci 351	Hosein Barzekar	Lapham 213	23:00-23:59
201	CompSci 361	Apoorv Prasad	EMS 1	Kyle
52	Infost 1984	O'brein	Online	All the time





## Create a new lab

New lab's name:

New lab's TA:

New lab's location:

New lab's time:

Submit

[Create labs page](#)



## Edit Lab

Lab name:

Compsci 361-801

TA name:

Apoorv Prasad

Lab location:

EMS E280

Lab time:

Tuesday 2:30pm-4:30pm

Save



[Home page](#)





## UWM TA Scheduling application

[My account](#)[Log out](#)

Are you sure you want to delete this lab?  
They will be permanently erased from the  
database.

[Delete labs page](#)

## UWM TA Scheduling application

[My account](#)[Log out](#)Search: [+ New User](#)

User	Role	Email	Address	Phone
Jayson Rock	Instructor	jrock@uwm.edu	EMS 420	1:00am - 1:00pm
Apoorv Prasad	TA	apoorvP@uwm.edu	Lapham 213	23:00-23:59
Dwayne Johnson	Singer	theRock@hollywood.com	EMS 1	Kyle
Doug	Douging	doug@doug.doug	Doug	Doug hour

[View users page](#)



## Create a new user

New user's name:

New user's role:

New user's phone number:

New user's address:

Submit

Create user page



## Edit User

User name:

Jayson Rock

User role:

Instructor

User email:

jrock@uwm.edu


User phone number:

352345252

Save




Edit user page


 **UWM TA Scheduling application**

My accountLog out

Are you sure you want to delete this user?  
They will be permanently erased from the database.



Delete user page

 **UWM TA Scheduling application**

My accountLog out

Who would you like to notify:

Instructors

What would you like to notify them about:

New Course

Send

Send notifications page

*Method Descriptions*

Function: createAccount()
Preconditions: account is not already created with user email/name
Postconditions: account is created with appropriate values (email/name)
Side-effects: None
p1: check new account against database of created accounts, if matching keys return error message stating account already exists p2: if not matching keys, information is entered by user input p3: account being created is stored within database, associated email is paired as key/value

Function: deleteAccount()
Preconditions: account is already created with user email/name
Postconditions: account is removed and email/name is removed
Side-effects: None
p1: check account against database of created accounts, if no matching keys return error stating account doesn't exist p2: if matching keys, account is removed from database p3: update database

Function: editAccount()
Preconditions: account is already created with user email/name
Postconditions: account is updated with corresponding information (user email/name)
Side-effects: None
p1: account is found within database by matching key pairs p2: matching account is pulled from database p3: information associated with the account is updated with user input p4: resulting information is then updated within the database

Function: setOfficeHours()
Preconditions: account is already created
Postconditions: set office hours for specific accounts
Side-effects: None
p1: account is found within database p2: matching account is pulled from database p3: set office hours is updated with user input p4: resulting information is then updated within database

Function: deleteOfficeHours()
Preconditions: account is already created and office hours are prefilled
Postconditions: remove office hours on specific account
Side-effects: None
p1: account is found within database by matching key pairs p2: matching account is pulled from database p3: remove set office hours, and update to empty fields p4: resulting information is then updated within database

Function: editContactInfo()
Preconditions: account is already created and contact info is already present
Postconditions: update specific contact info that is edited
Side-effects: None
p1: account is found within database by matching key pairs p2: matching account is pulled from database p3: contact info (email/phone/name/etc.) is updated by user input p4: resulting information is then updated within database

Function: getContactInfo()
Preconditions: account is already created and has contact info already present
Postconditions: return contact info to specific account
Side-effects: None
p1: account is found within database by matching key pairs p2: matching account is pulled from database p3: all value pairs associated with account is returned

Function: sendNotification()
Preconditions: accounts are already created with corresponding emails
Postconditions: send emails to selected accounts
Side-effects: None
p1: account is found within database by matching key pairs p2: matching account is pulled from database p3: value of email from account is saved in variable, variable creates a list of emails p4: resulting list of emails is sent notification (user input/preset notification)

Function: createCourse()
Preconditions: course does not already exist
Postconditions: create course
Side-effects: None
p1: course being added is matched against database of created courses p2: if course exists, error message stating so p3: if course does not exist, take user input and store course within database

Function: deleteCourse()
Preconditions: course already exists
Postconditions: delete course
Side-effects: None
p1: search course to be deleted against database of created courses p2: if course does not exist, error message stating so p3: if course does exist, remove course from database

Function: assignCourse()
Preconditions: course already exists and account to be assigned to already exists, course is not already assigned to account (no duplicates)
Postconditions: course is assigned to specific account, account is updated with course

Side-effects: None
p1: search course to be assigned against database of created courses p2: if course does not exist, error message stating so p3: if course does exist and is already assigned, error message stating so p4: if course does exist and is not assigned to anyone currently, assign course to person p5: update database

Function: assignTACourse()
Preconditions: course already exists and TA account already exists, TA is not already assigned to that specific course
Postconditions: course is assigned to specific TA account
Side-effects: None
p1: search course to be assigned to TA against database of created courses p2: if course does not exist, error message stating so p3: if course does exist and is already assigned to another TA, error message stating so p4: if course does exist and is not assigned to anyone currently, assign course to person p5: update database

Function: viewCourse()
Preconditions: course already exists and instructor account is already created
Postconditions: return instructor course string
Side-effects: None
p1: search course against database of created courses, if not matching keys return error message p2: search account against database of created accounts, return error message if no matching keys p3: return instructor and course information

Function: viewTACourse()
Preconditions: course already exists and TA account is already created
Postconditions: return TA course string
Side-effects: None
p1: search course against database of created courses, return error message if no matching keys p2: search account against database of created accounts, return error message if no matching keys p3: return TA and course information

Function: setContactInfo()
Preconditions: account is already created and does not have contact info entered previously
Postconditions: update account with contact info
Side-effects: None
p1: search account against database of created accounts, return error message if no matching keys p2: enter contact information from user input p3: update created accounts database

*Sprint PBI*

<b>As a Supervisor/Administrator, I want to be able to create accounts for users, so that the accounts can be added to correct group.</b>
<b>M:</b> Need to create database table, write tests, design and implement user interface
GIVEN an account with my unique identifier (possibly email or username) does not exist WHEN I enter my personal information AND a valid password AND the password is confirmed THEN an account is created

<b>As a Supervisor/Administrator, I want to be able to create courses for instructors, so that the instructor can upload appropriate information that they and the TAs can utilize.</b>
<b>M:</b> Need to create database table, write tests, design and implement user interface
GIVEN the course is not already created AND I enter the course information THEN the course is created

<b>As a Supervisor/Administrator, I want to be able to edit accounts for users, so that any information needed to be updated or changed is allowed to do so.</b>
<b>S:</b> Need to update database table with updated information, write tests, implement user interface
GIVEN there is no information entered prior AND the account is already created WHEN the edit information is selected THEN the account is updated with correct information
GIVEN there is information already populated in the fields WHEN the edit information is selected THEN the account is updated with correct information

<b>As a Supervisor/Administrator, I want to be able to access all data in the system, so I am able to see corresponding information</b>
<b>M:</b> Need to create database table, write tests, implement user interface
GIVEN supervisor/administrator's account is already created AND data is being stored in the system WHEN the access all data is selected THEN available data is formatted onto the screen

<b>As a Supervisor/Administrator, I want to be able to assign instructors to courses, so all instructors have their specific courses ready.</b>
<b>S:</b> Need to access database table, write tests, design and implement user interface
GIVEN instructor's account is already created AND appropriate courses are created WHEN an instructor is selected AND a course is selected THEN the selected course is assigned to the selected instructor

<b>As a Supervisor/Administrator, I want to be able to delete accounts for users, so that accounts that are no longer needed will not be using storage space.</b>
<b>S:</b> Need to update database table with removed account, write tests, implement user interface
GIVEN an account with a unique identifier (possibly username) exists AND the unique identifier matches the account to be deleted WHEN the supervisor/administrator selects the delete button THEN the account is removed from the storage space

---

As a Supervisor/Administrator, I want to be able to assign TAs to courses, so each instructor has the appropriate amount of TAs to cover the workload

---

S: Need to access database table, write tests, implement user interface

---

GIVEN the TA account is already created

AND the courses are created

WHEN an TA is selected

AND a course(s) is selected

THEN the selected course is assigned to the selected TA

---