Heavenly Medina
CSC 415-01
March 26, 2020
Assignment 3 - Open Source Software: Analysis and Design

VM: student1@csc415-server10.hpc.tcnj.edu, sysadmin@csc415-server10.hpc.tcnj.edu
GitHub:

- repository: Creeport
- full pathname: https://github.com/medinah2/Creeport/

1. UML Diagrams:

- **Use Case Descriptions** for the key use cases.
- key cases: file a report, view all reports, view recent reports, receive alerts, authenticate email

Use Case	file a report
Primary Actor	student/individual filing a report
Goal in Context	To allow users to file a report of an incident of harassment.
Preconditions	User must have previously created or logged in with an account verified through email confirmation and a valid password.
Trigger	A user wishes to file a report anonymously and online rather than in person.
Scenario	1. The user logs onto the creeport website 2. The user enters their email and password 3. After successfully logging in the user can select “file a new report” 4. User will fill out a form that has loaded specifying specific details about the incident. 5. After the report is submitted the user will receive a message saying that their report has successfully be submitted. 6. The user will be redirected to the user home page.
Exceptions	1. Report not filled out in entirety: report will not be submitted, user redirected to incomplete field 2. User does not finish submitting report: draft to be saved. (possibly will implement in future iterations)
Priority	high priority, to be implemented early on

When Available	second prototype
Frequency of Use	frequent, primary function of web application
Channel to Actor	via webpage
Secondary Actors	Title IX administrator
Channels to Secondary Actors	via report stored on database
Open Issues	1. Should verified Title IX members be able to file reports as well, and have them classified as verified reports? 2. Should there be a way to save report drafts in the event that the user does not want to complete a report?

Use Case	view all reports
Primary Actor	Title IX administrator
Goal in Context	To allow Title IX members to view all reports filed through the webpage.
Preconditions	Title IX administrator has an account that has been verified for access status.
Trigger	Title IX member decides to view all reports reported through Creeport.
Scenario	1. The Title IX administrator logs onto the creeport website 2. The administrator enters their email and password 3. After successfully logging in the administrator can select "view all reports" 4. All reports ever filed will be displayed on the screen. (default) 5. The administrator can view most recent reports based on a menu selection (based on type of report, based on reports by specific offenders)
Exceptions	1. No reports have been filed yet: administrator will view a message that no reports have been filed yet
Priority	high
When Available	3rd prototype
Frequency of Use	frequent

Channel to Actor	via web application
Secondary Actors	-
Channels to Secondary Actors	-
Open Issues	1. Should users be able to view previous reports that they have submitted? 2. How should reports be ordered?

-

Use Case	View recent reports
Primary Actor	Title IX administrator
Goal in Context	To allow Title IX members to view all reports filed through the webpage.
Preconditions	Title IX administrator has an account that has been verified for access status.
Trigger	Title IX member decides to view recent reports reported through Creeport.
Scenario	The user logs onto the creeport website 1. The user enters their email and password 2. After successfully logging in the administrator will select the tab “view recent reports” 3. All reports filed from that day will be displayed on the screen. (default) 4. The administrator can view most recent reports based on a menu selection (that day, that week, that month)
Exceptions	1. No reports have been filed yet: administrator will view a message that no reports have been filed yet 2. No new reports have been filed: If no reports were filed that day there will be a message that states “no reports were filed today” and the administrator can use the menu to view older reports.
Priority	medium priority
When Available	3rd prototype iteration
Frequency of Use	high, may be checked frequently
Channel to Actor	via web application

Secondary Actors	-
Channels to Secondary Actors	-
Open Issues	1. How will recent reports be ordered? In terms of time submitted or in terms of severity?

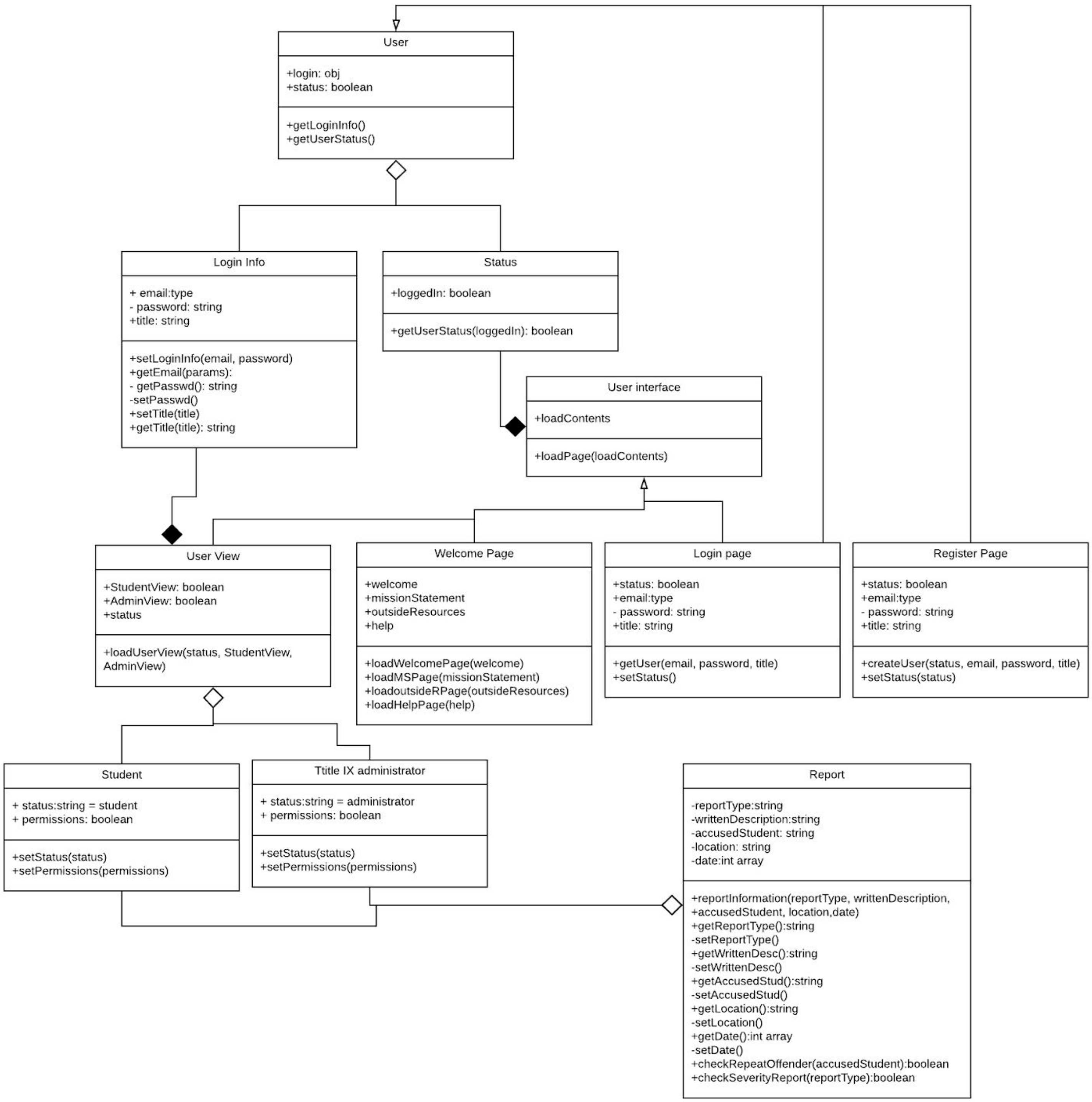
use case	receive alerts
Primary Actor	Title IX administrator
Goal in Context	To alert Title IX members of serious reports or of possible repeat offenders to reach out to students via email.
Preconditions	There are previously filed reports or a new serious report was just filed. Title IX member is logged in to their account.
Trigger	A serious report has been entered or someone previously accused has a new report by a different user.
Scenario	1. The Title IX administrator enters their email and password 2. After successfully logging in the administrator will select the tab "view alerts" 3. All reports filed from that day will be displayed on the screen. (default) 4. The administrator can view most recent reports based on a menu selection (that day, that week, that month)
Exceptions	1. No reports have been filed yet: administrator will view a message that no reports have been filed yet 2. No alerts have been generated yet: administrator will view a message that no alerts have been reported yet
Primary Actor	Title IX administrator
Priority	high/medium
When Available	3rd/4th prototype
Frequency of Use	dependant on frequency of report
Channel to Actor	via web application or email alerts
Secondary Actors	-
Channels to	-

Secondary Actors	
Open Issues	1. Should students also receive alerts automatically? Or should these reports be generated by Title IX administrators?

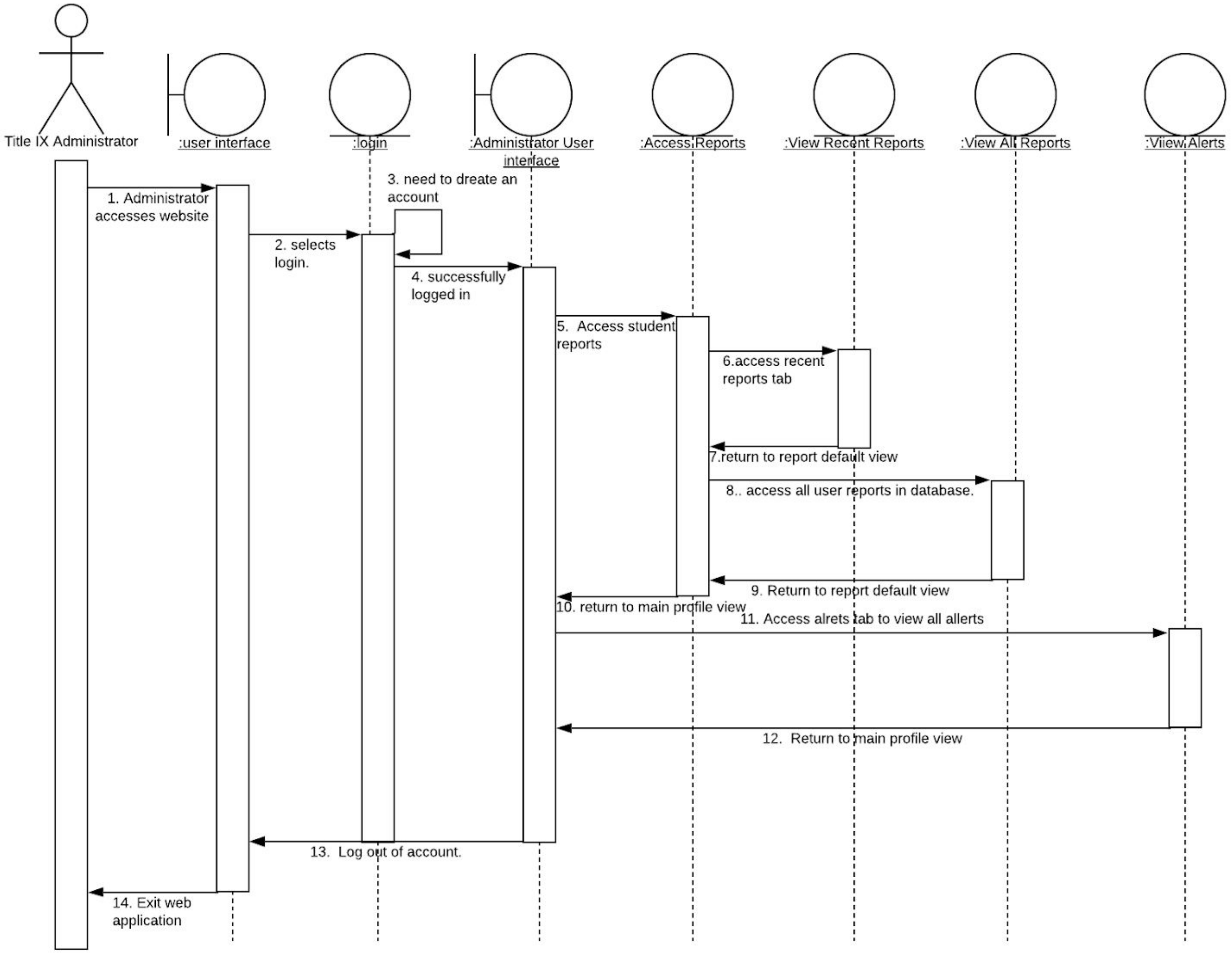
Use Case	Authenticate Email
Primary Actor	User, Title IX administrator
Goal in Context	To reduce ensure that emails are valid in order to authenticate Title IX administrators as well as users emails as a means for future contact information. Also aims to reduce potential spam reports.
Preconditions	User has entered an email when registering for an account.
Trigger	User is setting up an account.
Scenario	1. User/title IX administrator goes to the website 2. They select “join now” to register an account 3. Enter an email and password 4. An email will be sent to the user to confirm their account. 5. Email confirmation will unlock the user main page and allow users to access and navigate personal reports(if basic users) or access report data (if Title IX member)
Exceptions	1. The user already has an account: these steps will have already been completed 2. Email is not deemed valid/proper format (xxx@xxxx.xxx): user will be prompted to enter a valid email 3. Confirmation email is not activated: user will not be able to file a report or reach user login page
Priority	high, this will help with the submission of spam reports
When Available	3rd prototype
Frequency of Use	frequent, whenever a new account is created
Channel to Actor	via web application and email
Secondary Actors	-
Channels to Secondary Actors	-

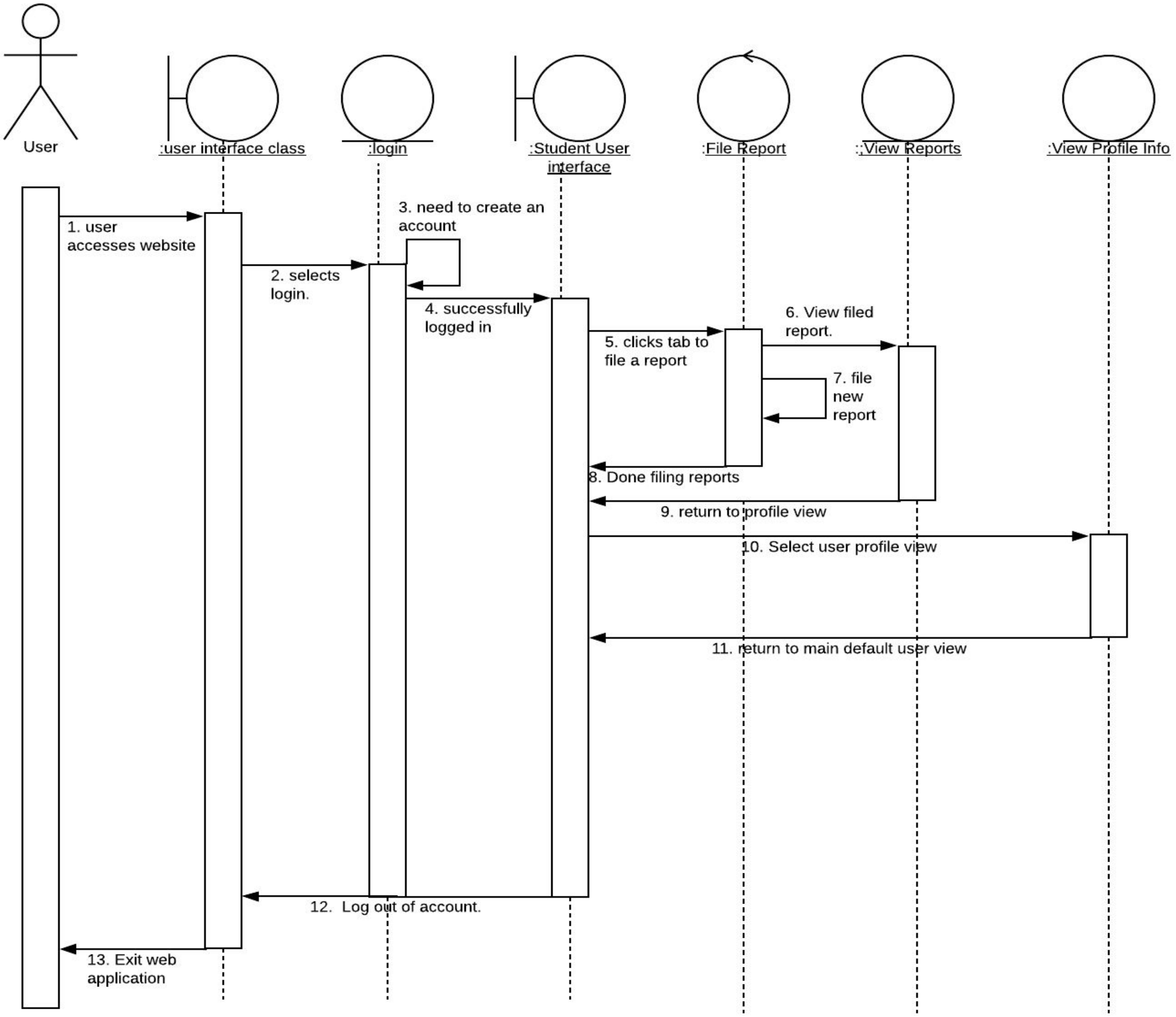
Open Issues	1. How will you differentiate Title IX emails from student emails? 2. What if users cannot view/ find the confirmation email link?

- **Detailed Design Class Diagram**

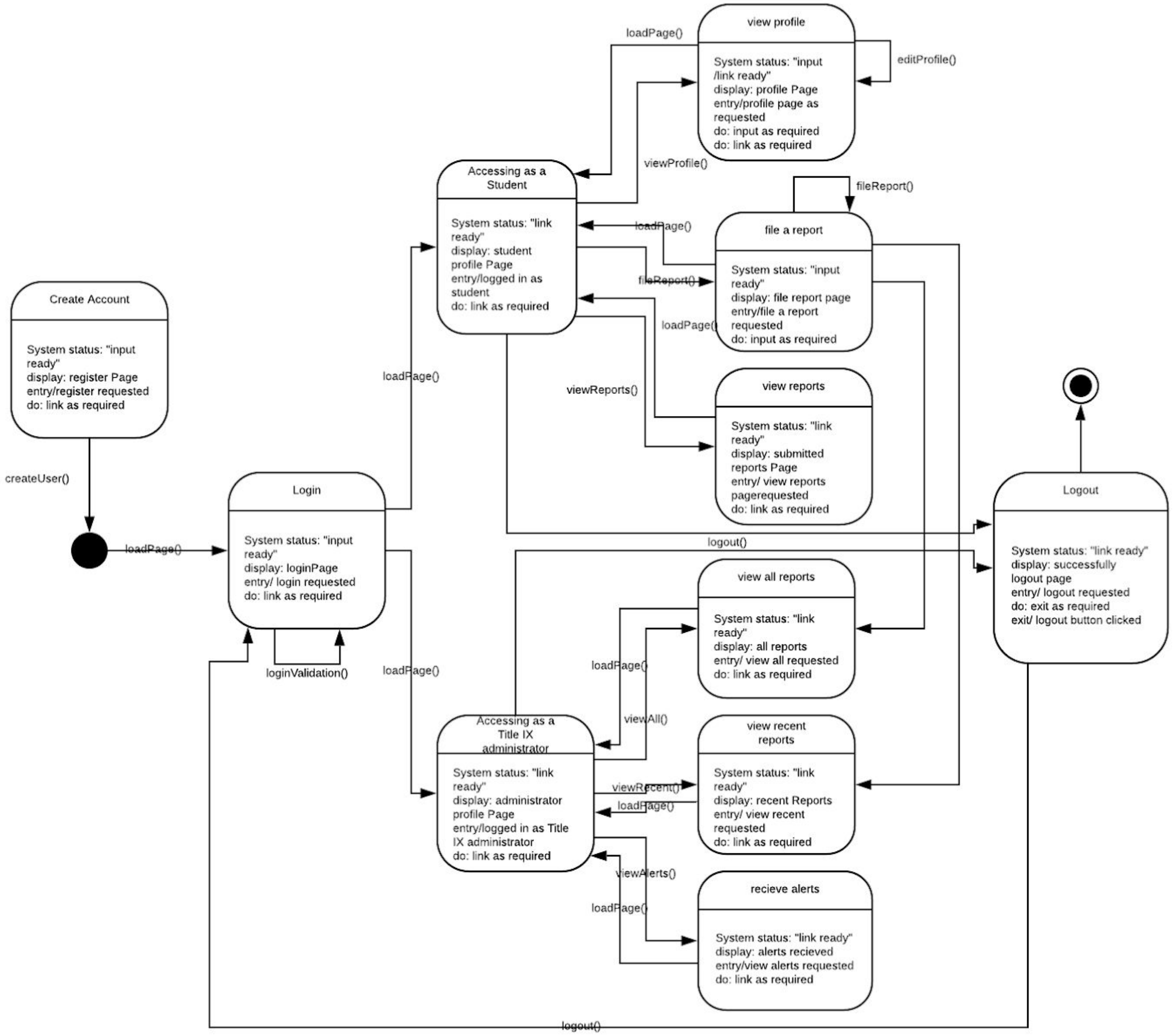


- Detailed **System Sequence Diagram(s)**

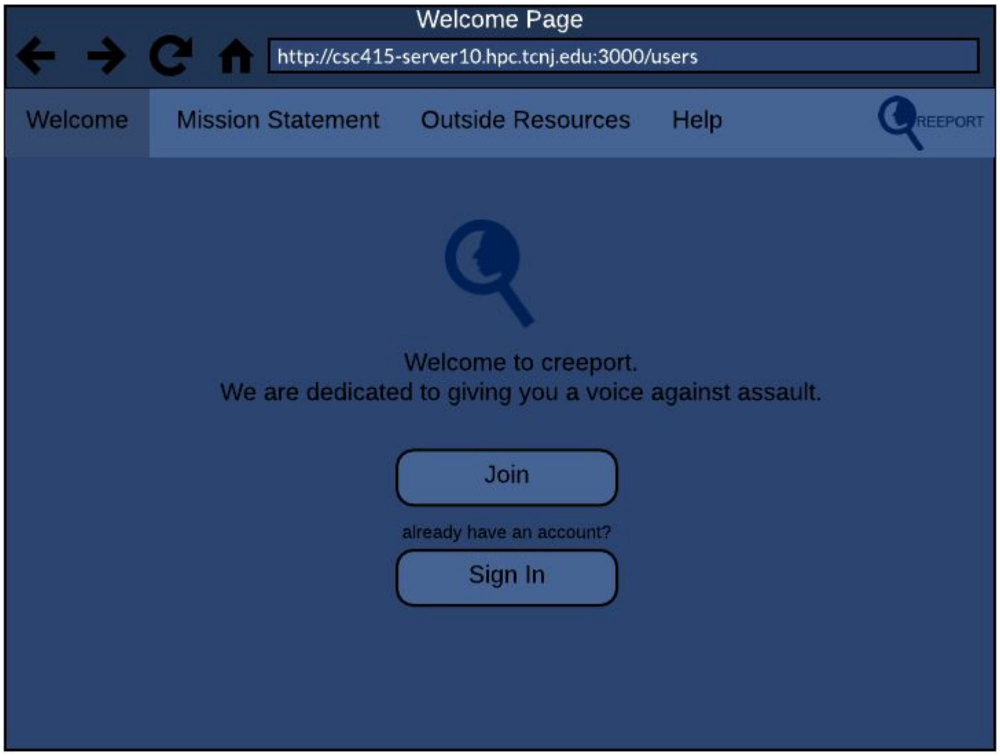

Title IX Administrator
:user interface
:login
:Administrator User interface
:Access Reports
:View Recent Reports
:View All Reports
:Viiew Alerts
1. Administrator accesses website
2. selects login.
3. need to dreate an account
4. successfully logged in
5. Access student reports
6.access recent reports tab
7.return to report default view
8.. access all user reports in database.
9. Return to report default view
10. return to main profile view
11. Access alrets tab to view all allerts
12. Return to main profile view
13. Log out of account.
14. Exit web application


User
:user interface class
:login
:Student User interface
:File Report
:;View Reports
:View Profile Info
1. user accesses website
2. selects login.
3. need to create an account
4. successfully logged in
5. clicks tab to file a report
6. View filed report.
7. file new report
8. Done filing reports
9. return to profile view
10. Select user profile view
11. return to main default user view
12. Log out of account.
13. Exit web application

- **Statechart** to model the overall behavior of the system. *See slides 88, 89 and 90 in the "SS3 Requirements" slide set for the correct notation.*

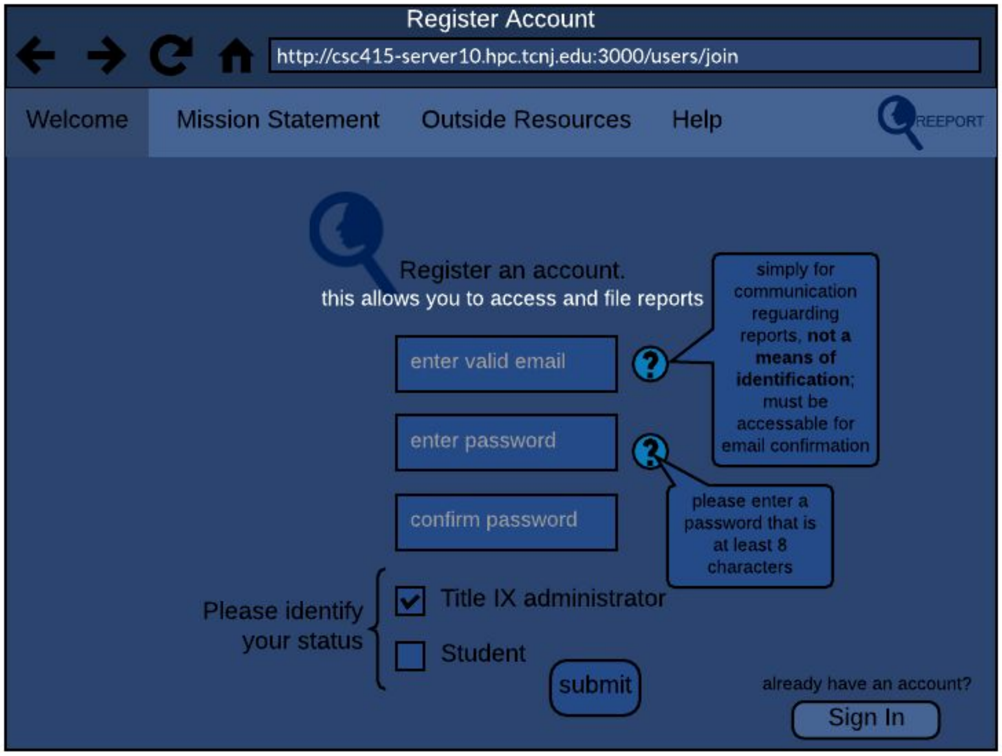


2. **User Interfaces**



Welcome to creeport.
We are dedicated to giving you a voice against assault.

already have an account?



Register an account.
this allows you to access and file reports

simply for communication reguarding reports, **not a means of identification**; must be accessable for email confirmation

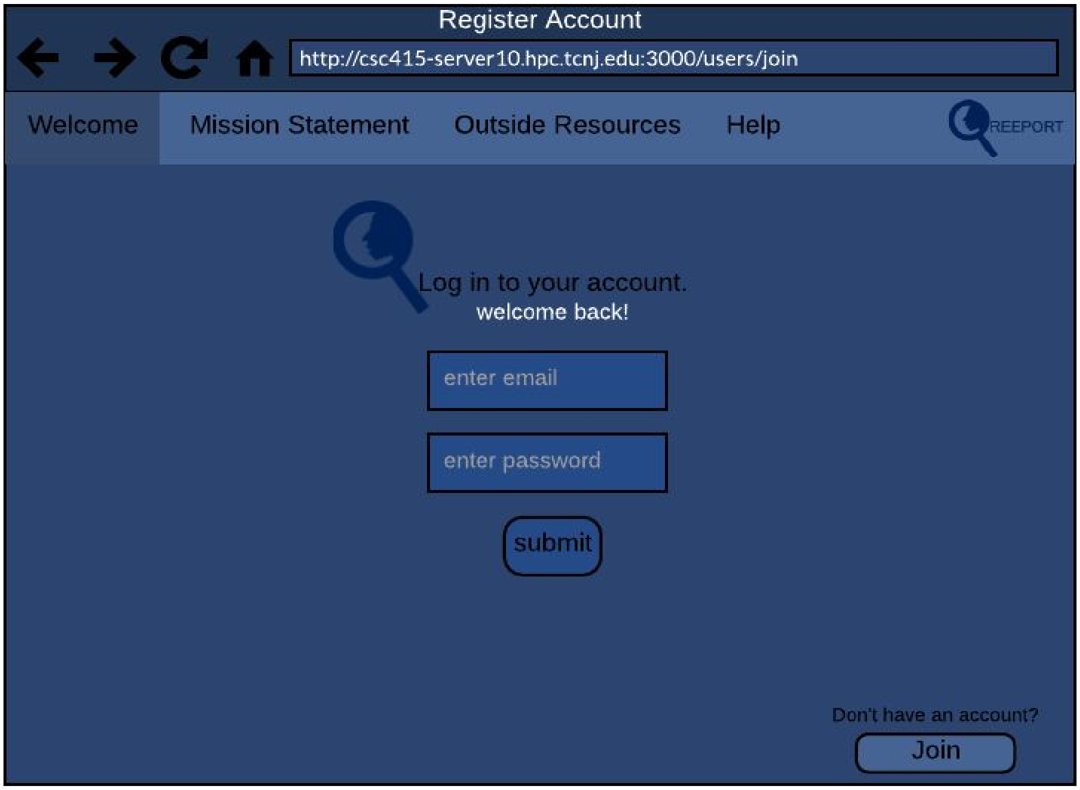
please enter a password that is at least 8 characters

Please identify your status

☑ Title IX administrator

□ Student

already have an account?



Register Account

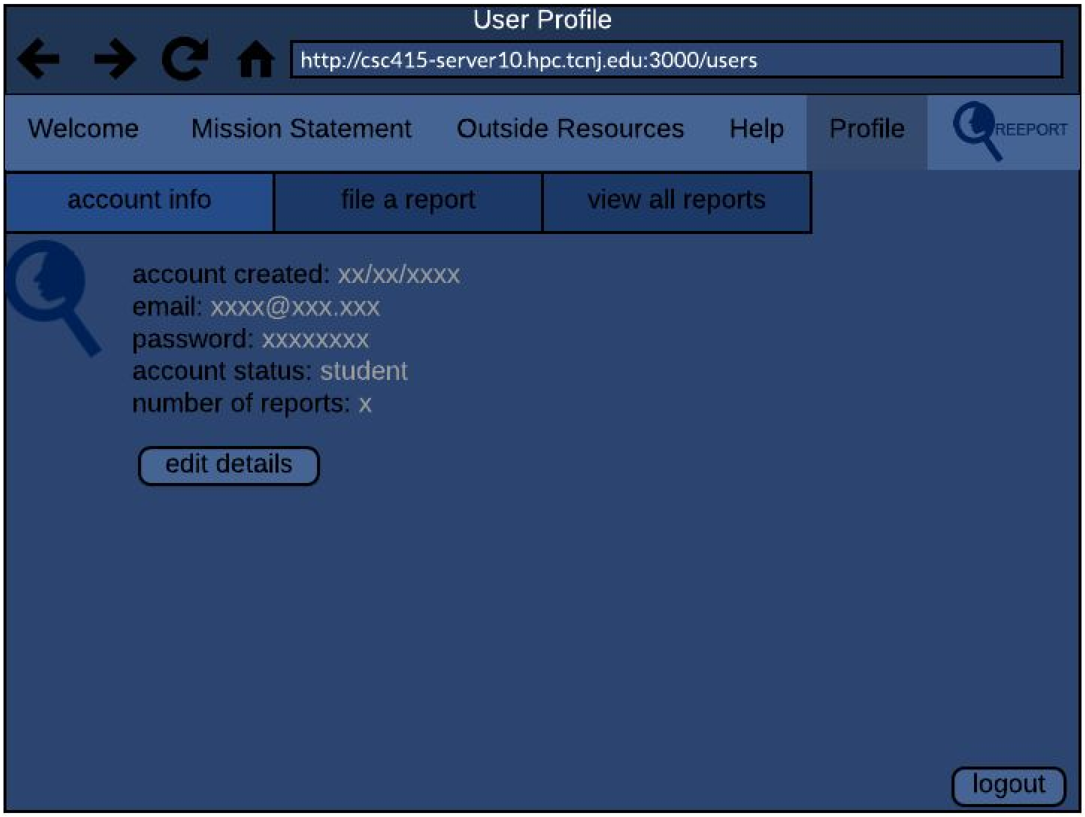
http://csc415-server10.hpc.tcnj.edu:3000/users/join

Log in to your account.
welcome back!

enter email

enter password

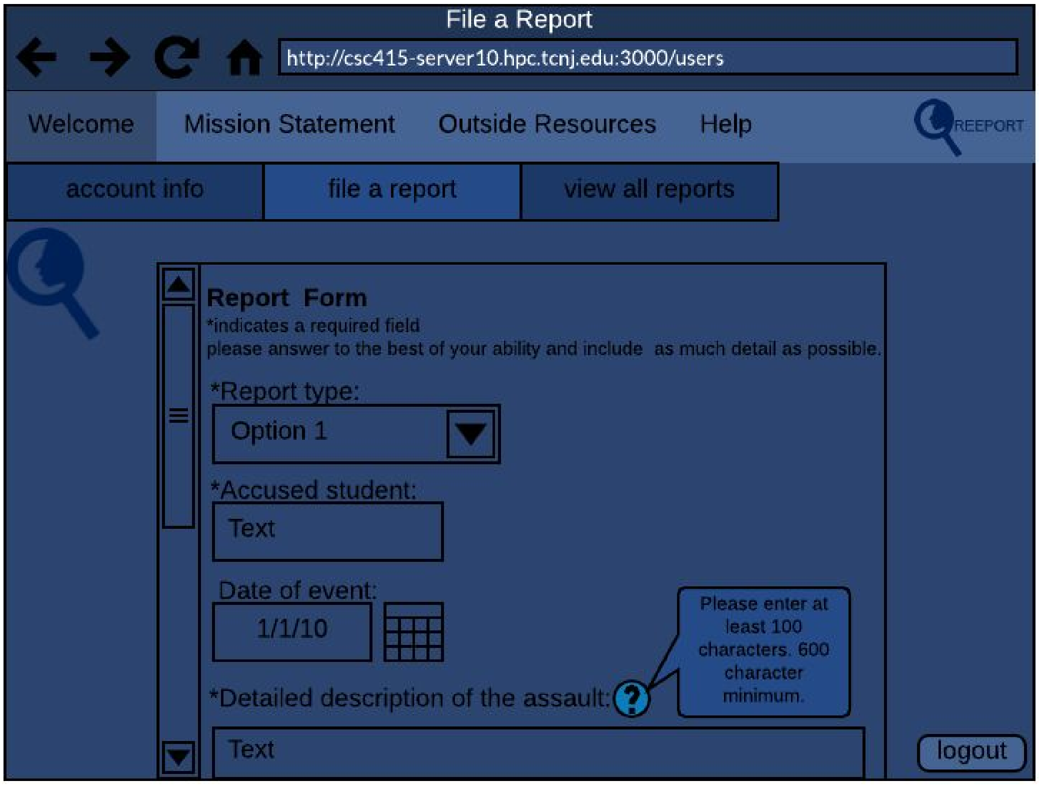
Don't have an account?



User Profile

http://csc415-server10.hpc.tcnj.edu:3000/users

account created: xx/xx/xxxx
email: xxxx@xxx.xxx
password: xxxxxxxx
account status: student
number of reports: x



Report Form

*indicates a required field

please answer to the best of your ability and include as much detail as possible.

*Report type: Option 1

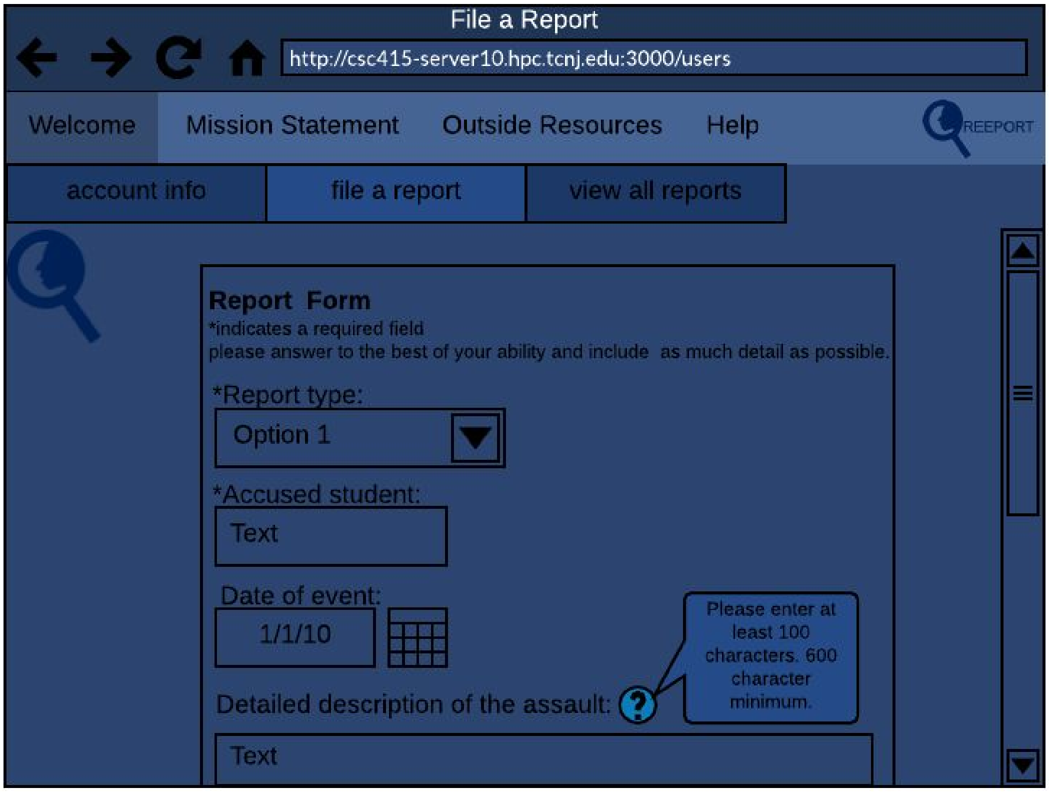
*Accused student: Text

Date of event: 1/1/10

*Detailed description of the assault: ?

Please enter at least 100 characters. 600 character minimum.

Text



Report Form

*indicates a required field

please answer to the best of your ability and include as much detail as possible.

*Report type: Option 1

*Accused student: Text

Date of event: 1/1/10

Detailed description of the assault: ?

Please enter at least 100 characters. 600 character minimum.

Text

- affordance
    i. Each control has icons and text to signify how to access/interact with each function. Uses typical web application format that users are familiar with so they will most likely have prior knowledge as to how basic aspects will function. Not overly complicated design and users do not need to be aware of additional information in order to use functionality unless specifics are specified that can be accessed through the help icons.
- visibility
    i. The screen is not over crowded so that users can easily view and locate functions within the program via navigable tabs. The design will provide visible changes when a user is interacting with an element so they obtain immediate feedback.
- 8 golden rules:
    i. strive for consistency
        1. There is a consistent color scheme, font style, and menu bar to ensure consistency within the UI. Each page appears similar so you can tell that they each go together.
    ii. enable frequent users to use shortcuts
        1. I have the login information configured so that it can be loaded in without needing to retype it for frequent users.
    iii. offer informative feedback
        1. There are help icons that users can access to additional information like password length requirements and that required fields are filled out.
    iv. design dialogs to yield closure
        1. Displays such as successfully logged in will appear before redirecting users to their personal profile.
    v. offer simple error handling
        1. Pop-ups will be displayed if a user has entered invalid information and prompt them to re-enter information correctly.
    vi. permit easy reversal of actions
        1. Users can view all reports and edit information. If you clicked on register instead of login there is a link at the bottom of the page that allows users to access the login page and vice versa.
    vii. support internal locus of control
        1. Options to edit information will allow for users to seem like they are in control and able to interact with the system.
    viii. reduce short-term memory load
        1. There is not much information that users need to remember when entering/accessing the website.

3. **How my application design meets each of the above requirements.**
    - My design meets modularity as it compartmentalizes data and functions through the use of separated functions based on variable use to avoid reusing the same variables multiple places throughout the program. Encapsulation would be utilized to group together related classes, attributes, and methods as noted within my case diagram. Through modularity and encapsulation information hiding would also be included as I will be using private and public classes to provide a controlled interface to only include what is necessary, thus making it more efficient and better for reuse. Elegance and efficiency would be utilized when implementing algorithms as I would be using algorithms better suited to my specific application, so it will not need to process specific amounts of data if it is not being used constantly, as well as running the most efficiently for my program. The data structures that I will be using will be efficient for the data types I am storing and using to reference.

4. Test Case Design

    - **Approaches I will use for each of unit, integration, and system testing.**
        i. For unit testing I will use the VS code debugger to set breakpoints to test certain functionality. This will be tested during code implementation and will account for analysis of the code and
        ii. For integration testing I will access the website and enter test cases to test for errors that may occur.
        iii. For system testing I will have family members/friends navigate through the website to see if they are able to understand the UI and obtain functionality, and uncover any errors that may occur.

    - **Debugging and testing tools I will use.**
        i. I have been coding on my VM through VS code so I will primarily be using the debuggers available through there as well as the ruby-debugger-ide gem and debase gem the VS extension vscode-ruby. This should allow me to set breakpoints for testing as well as being able to view the rails server logs.

Functionality Tested	**Inputs**	**Expected Output**	**Actual Output**
Valid email entered	medinah2@tcnj.edu	valid email format	
invalid email entered	medinah2 medinah2@	invalid email format, not supported	

	medinah2@tc medinah2@tcnj.		
sufficient password entered	password	valid, at least 8 characters entered	
insufficient password entered	pass	invalid, password is too short	
passwords match when registering	password & password	valid, passwords match	
passwords do not match when registering	password & pass password & drowssap	invalid, passwords do not match	
email confirmation	valid email	please confirm your account on creeport www.xxxx.xxx	
report fields filled out	full report	“report filed successfully”	
not all report fields are filled out	report missing fields that	“report incomplete” will not let users access	
report meets length qualifications	report with > 100 and < 600 characters	valid report length	
not all length qualifications are met	report with < 100 characters report with > 600 characters	invalid report length, too short. invalid report length, too many characters used	
view all reports	enter 3 valid reports no reports entered	all should be displayed message signifying that no reports are submitted should be displayed	
alerts are received	file a serious report file 2 reports with the same accused individual on 2 separate accounts	should display an alert on user page	
login successful	enter correct login	“login successful”	

	information		
login unsuccessful	enter non-existing account information	“login unsuccessful”	
severe report identified	file severe report	“severe report” should send alert	
repeat offender identified	file 2 reports with the same accused individual on 2 separate accounts	“repeat offender” popup should send alert	