CIS4328 – Information Systems Senior Project II

Final Project Report

Honors in Computing Hours Tracking

Team: CAPA Tech

Chance Abenes

Sophia Abuzeni

Christopher Clark

Alexander Perez

Instructor: Karthikeyan Umapathy

University of North Florida

School of Computing

May 1st, 2023

Executive Summary

Team members Chance Abenes, Sophia Abuzeni, Christopher Clark, and Alexander Perez, along with mentor Jonathan Ohlrich compose CAPA Tech, a group tasked with developing the Honors in Computing Hours Tracking application for the University of North Florida's (UNF) director of the School of Computing (SoC), Sherif Elfayoumy. The application's purpose would be to provide students, faculty, and administrative staff with a tool to manage student submission, faculty approval, and administrative review of events students attend and submit which pass some academic criteria set by faculty.

As many people would be utilizing the same system, users of the system would need accounts to differentiate their roles and data entered; therefore, we developed an account creation system. Support for the system would be minimal, so account management self-service was implemented. Faculty and administrator roles would need to be subject to additional approval for account creation, and an account verification process was created to facilitate that. Students would need to be able to submit details of events they attended to their instructors, so students are given a table of previously submitted forms as well as a form to submit new ones. Faculty would need to be able to approve or deny forms submitted by students, so they are given a dashboard containing all forms assigned to them by their students with buttons to approve or deny with a comment back to the student. Administrators would need to be able to approve or deny incoming account requests for new faculty or administrator accounts, and additionally view reports of approved student forms with the ability to export them. All functionalities have been provided, quality assurance testing has proven successful, the product has been delivered successfully.

Source code was organized around an MVC (model, view, controller) architecture to ensure efficient and ease of organization of the product. Implementation decisions were made using a combination of GitHub and Jira. Our solution was designed using Next,js, React, Express, and MySQL. Our application was initially set up on Microsoft Azure, but due to client constraints a server was set up on the UNF network to host our application using NGINX as a reverse proxy. Gaining an understanding of the above tools was critical to our success, and we also gained skills in teamwork, leadership, and coordination while using the Agile development process with a real client.

Introduction

Client

Sherif Elfayoumy, director of the School of Computing (SoC) at the University of North Florida (UNF) is responsible for enabling SoC faculty to be successful in their instruction of students. The SoC's role within the community, both locally and globally,

is to train future computing professionals for the workforce. If Sherif is successful, the world is better for it.

Problem Statement

The Honors in Computing program is in development, and one of the new requirements of the program is a system which can keep track of hours students spend on different events which should count toward their degree progress. A system was needed to allow students, faculty, and administrative staff to interact with these submitted hours and approve them, deny them, or run reports on them.

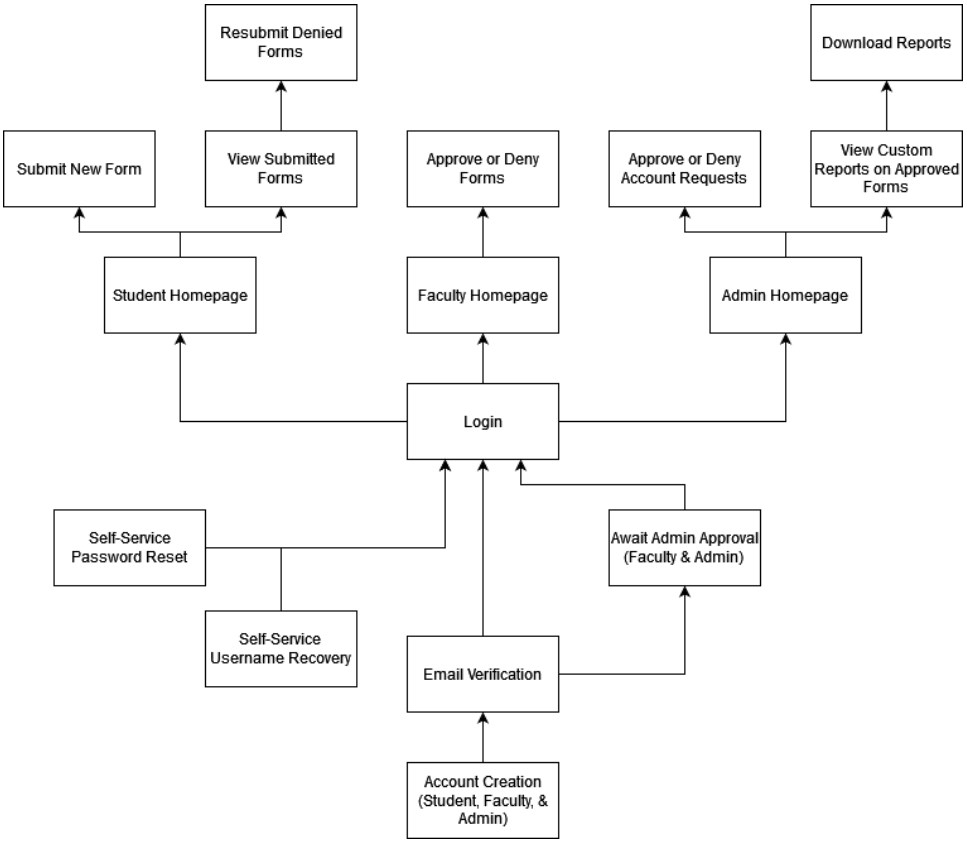
Background

No existing system was in competition with our project. We created a custom application to fill a need that had no previous answer.

Project Goals and Objectives

Our project, when complete, would address all the needs identified above. This includes but is not limited to a system which allows students to submit reports of events they've attended for which they'd like to receive academic credit, a system which allows faculty to approve or deny said reports with comment for administrators or a comment for students, and a system which allows administrators to view and download those reports.

Solution Developed



Software Development Methodology

This project was worked on iteratively. Each cycle consisted of a planning phase, a documentation phase, a programming phase, and a delivery phase. There were 6 primary cycles demarcated by deliverables necessitated by our instructor. The planning phase involved a team meeting in which we produced a plan for the cycle. The documentation phase took more time than anything else. We worked together to create documents the instructor requested based on our project during this time. Once we had completed these tasks, the last few days were used to get work done on the project itself. The delivery phase involved submission of documents to the instructor and demonstration of our progress to the client. We took feedback from these meetings and fed it back into the next cycle.

Cycle 1 documents

- Project Vision
- Initial Software Development Plan
- Product Backlog and User Stories

Cycle 2 documents

- List of identified Use Cases and relevant User Story alignment
- Outline Use Case Specification
- Complete Use Case Specification
- Use Interface Specifications

Cycle 3 documents

- Activity Diagrams
- Analysis Models
- Sequence Diagrams
- Preliminary Database Design

Cycle 4 documents

- Data Flow Diagram – Context Diagram
- Data Flow Diagram – Level 0 Diagram
- Architecture Model
- Package diagram with Class diagrams for each package

Cycle 5 documents

- Database Design
- Hardware and Software specifications
- User Interface Design

Cycle 6 documents

- Test Cases

- Deployment Plan
- Client Testing Summary Report

Academic Course Works

The answer to this question will be different depending on who you ask, but in general the required courses for Information Systems which made the greatest impact on our ability to succeed in this course were SPC2608, COP3703, COP3855, and COP4813. Others like COP3530 or CDA4010 contributed to our success more indirectly by training us to handle workloads involving planning and execution of programming techniques.

Mentor

Our mentor was instrumental in our success as a team. When we were completely lost, he was there to get us back on track. When we veered off course at times, he helped us to focus on the targets we needed to hit. When we got stuck on technical issues, he helped us break down those walls. We met with him weekly (bi-weekly on occasion), and he provided us with his thoughts on our progress and informed us of nearby pitfalls at every turn. We learned a lot about the reality of the academic processes we learned in this course as we were doing them, which helped to shape what we focused on along the way. Our interactions with our mentor were nothing but positive, and he has our sincerest gratitude for the help he provided to us along the way.

System Requirements

Overview:

CAPA Tech is working on Project 9 for UNF School of Computing's Honors in Computing Hours Tracking. The School of Computing is a part of the College of Computing, Engineering, and Construction.

Our Community Partner: Dr. Sherif Elfayoumy.

The project's aim is to automate the process of tracking honors students' leadership hours. This involves three key steps: First, enabling students to fill out and submit their activity reports; second, allowing faculty to approve or reject student-submitted requests; and finally, giving the school administration the ability to review individual student records and generate end-of-semester reports.

User Stories and Non-Functional Requirements:

- *User Stories:*

- Login

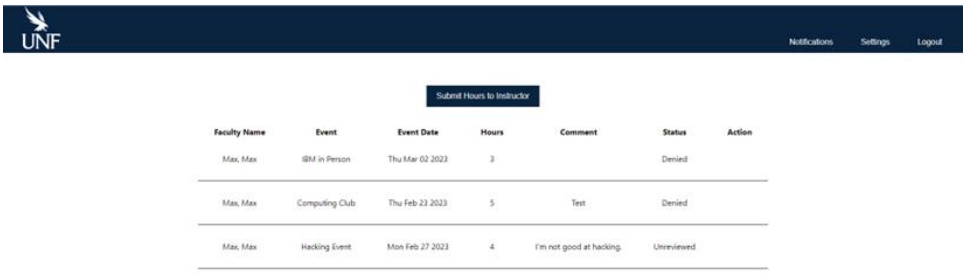
- Account creation
- Account verification
- Navigation
- Student submitting form
- Faculty reviewing student forms
- Reporting
- Logout
- Student, Faculty, Admin Homepage
    - *Non-Functional requirements:*
        - Password Security
        - End of semester reports
        - Different account roles

Functional Requirements

The primary objective of the project is to streamline the procedure for monitoring the leadership hours of honors students by automating the process. This includes three significant steps: Firstly, providing the facility to students to fill out and submit their activity reports; secondly, permitting faculty members to approve or decline student-submitted requests; and lastly, enabling the school administration to scrutinize individual student records and generate reports at the end of each semester.

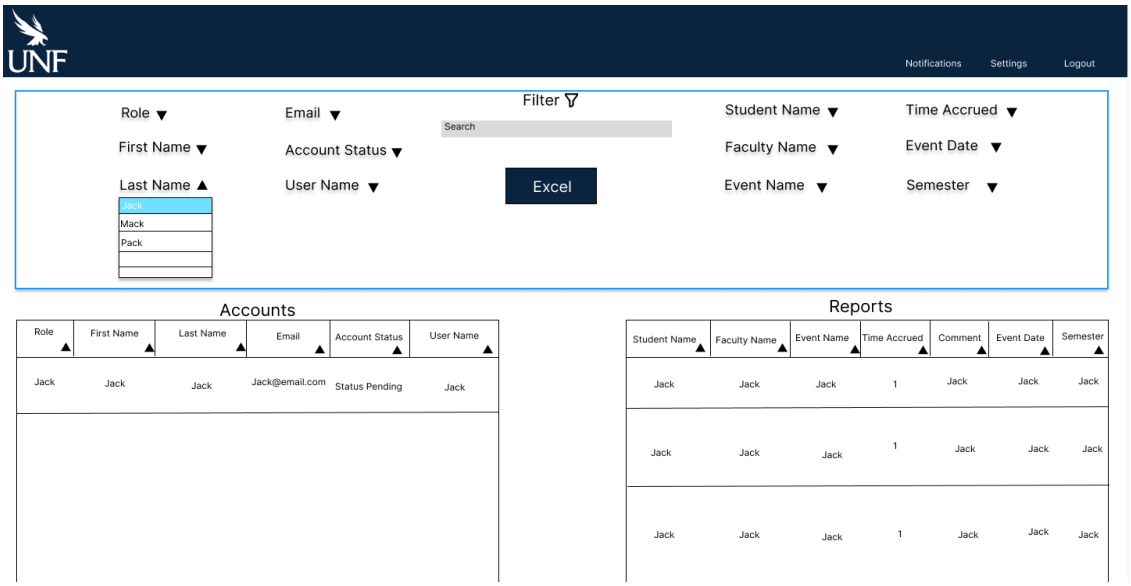
Use Case Diagram:

- *Student Homepage*



- *Faculty Homepage*

- *Admin Homepage*



Mandatory and Special Requirements

- *Mandatory and Special Requirements: Describe any other special requirements and mandatory needs provided by your community partner that are relevant to the successful functioning of the software system (e.g., hardware requirements, connections to external systems, etc.).*

  Hardware Requirements:

Hardware	Minimum	Recommended
CPU	2 vCPUs	4 vCPUs
Memory	4GB RAM	16GB RAM
Storage	50GB HDD	50GB SSD

Software Requirements:

- *Server*
  - Ubuntu 22.04 LTS
  - MySQL server
  - NodeJS
  - HTTPS certificate
- *Tech Stack*
  - MySQL server
  - ExpressJS
  - ReactJS
  - NodeJS
  - NextJS
  - Iron-Session

Client:

All modern browsers/operating systems are supported. (Google Chrome, Microsoft Edge, Firefox, and Safari) mobile devices supported. (Google Pixel, Android, & Apple iPhone).

User Requirements:

- Must be a UNF Student otherwise be associated with a UNF N-Number.
- Faculty must get account approved by an Admin.
- Admin account overviews the whole application and other user roles like student & faulty.

Use of Application:

- Students submit forms to Faculty.
- Faculty approves or denies forms.
- Admin views report and approves or denies accounts.

I previously used Azure to deploy to the cloud but ran out of credits and am currently in the process of switching to Ubuntu 22.04 LTZ.

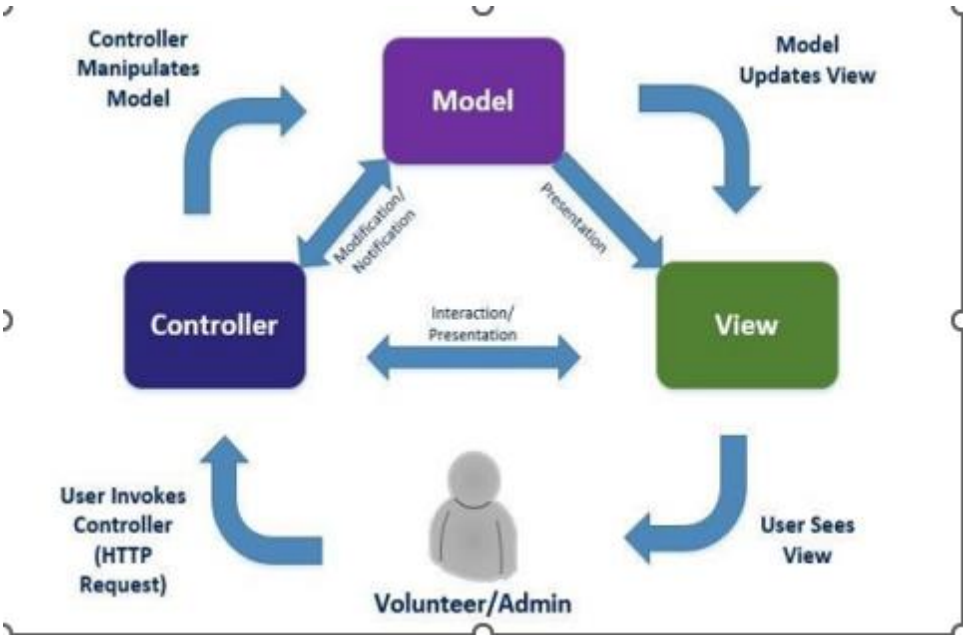
Detailed Design

Architectural Design

Summary

Model View Controller will be used as the main architectural design for UNF's School of Computing Hours System. The system will also feature architectural components that encapsulate parts of the Client-Server pattern.

Model



Purpose

The project's main architectural design is based on the Model, View, and Controller pattern. Due to the purposes behind the project, the system has a need for all CRUD elements. Users accessing the system will need to create reports upon which faculty members will need to be able to read these reports. A system administrator will also need to update and delete reports as needed. Users/clients utilizing the system will have to make these requests to the server through the web API which in turn will act upon the database, inserting/deleting based on the respective user. Any user that uses the system will be expected to access it via most web browsers on any mobile or static device given that it has access to the internet. Users will be constantly updating their profiles with every report they submit, and hours and reports will be logged into the system. At the admin level, they will be able to filter through these submitted reports and manipulate this recorded data within the database. MVC Architecture is key in this process and allows for this complete process to be cohesive and efficient for the purposes

behind our system. Users will utilize the views to use buttons and click links to call controller methods which in turn manipulate the model to access and display specific reports/data back to the view for users to utilize. All these processes interact with each other in specific patterns to ensure all data flows correctly and efficiently.

Architectural Rationale

Divide and Conquer

- The Model, View, and Controller architecture embodies the idea of dividing and conquering. Each competent of this architecture is separated by three main logical components: model, view, and controller. The user interacts with views which render the controller logic as interactions within the system which in turn manipulate data within the model.

Increased Cohesion

- The Model, View and Controller architecture is separated into three partitions of logic. However, each section relies on one another to create a responsive and overall, more cohesive structure. Each part of the structure flows perfectly to create our web application and ensure everything runs smoothly; this is only accomplished because of the MVC architecture.

Reduce Coupling

- Fundamentally the idea of MVC reduces the threat of coupling in code bases. Coupling is prevented by isolating components and preventing entities from propagating from one to another.

Increase Reuse

- Model, View, and Controller architecture allows for better reuse of code. While maintaining high cohesion of code, MVC allows for the reusability of code in the future. Because of the separation of these logical structures, it makes them easier to reuse.

Design for Flexibility

- The way MVC is structured may seem rigid due to the separation of these key components but, it creates a more flexible environment. If a new view is to be created based on an older controller it should be created with minimal problems. Controllers do not have to be overcomplicated and the redesigning of Views and Models alike will not be required with changes to older controllers.

Design for Testability

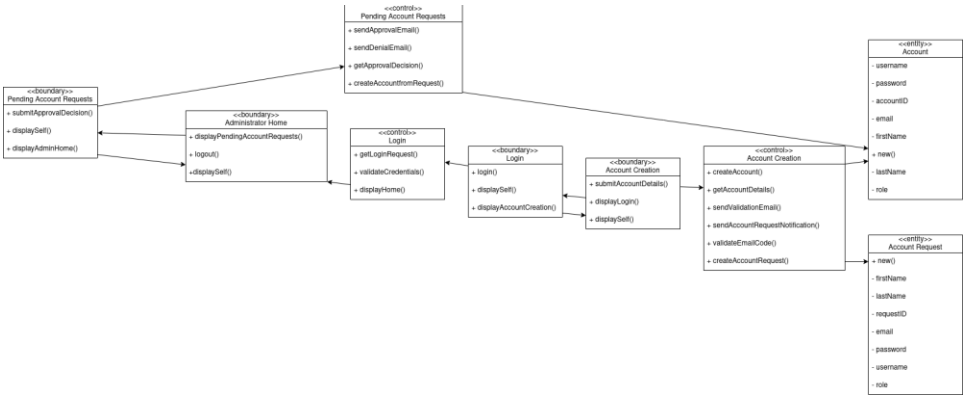
- The MVC architecture allows for excellent testability and debugging. Unit testing allows for singular views to be tested alongside the coinciding models and controllers

associated with them. In many cases this is used alongside the entire project and makes debugging a straightforward process to hash out small bugs in subdivisions of the project.

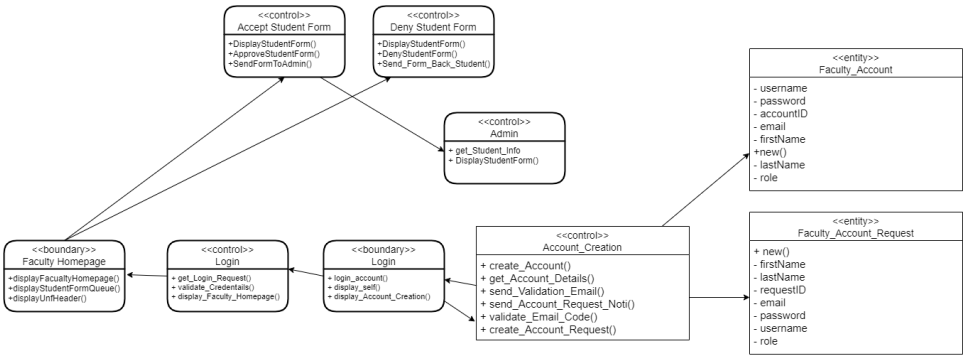
Design Specifications

This section will describe the key decisions made regarding the architecture of the system, along with a UML Package Diagram to provide a visual representation of the system's structure.

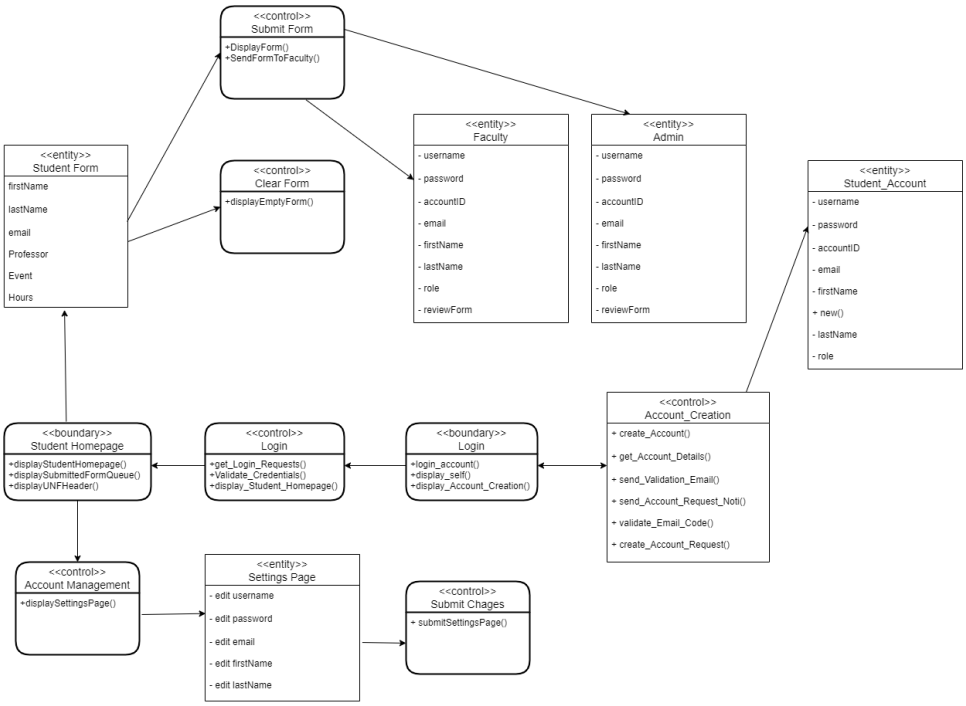
Account Creation:



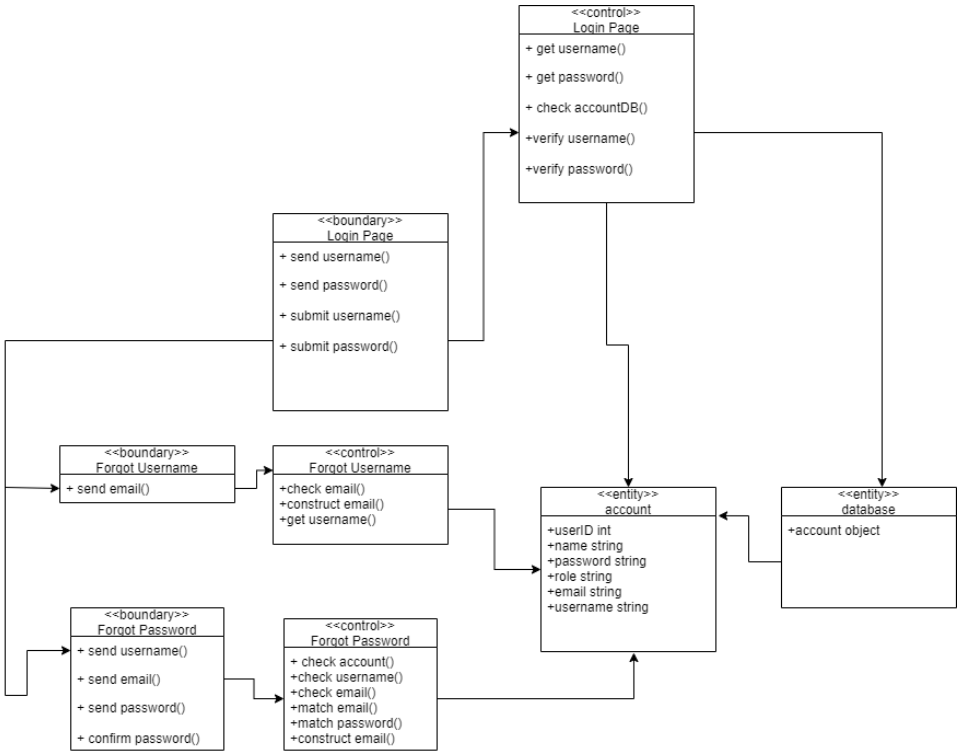
Student Submit Hours:



Faculty Homepage:

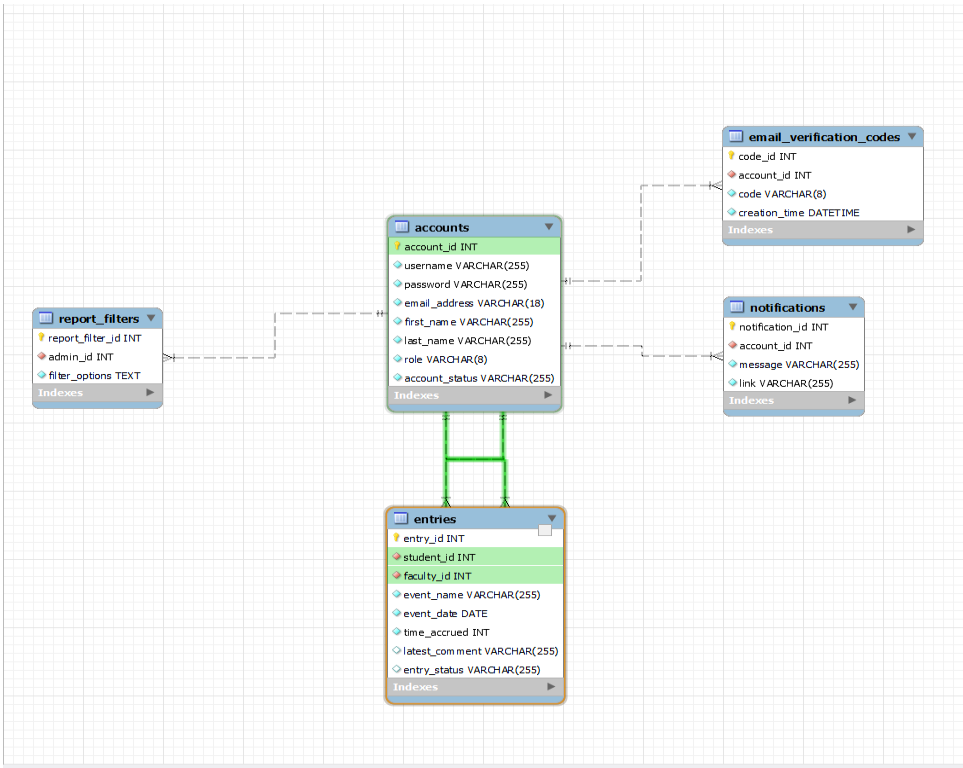


Admin Homepage:



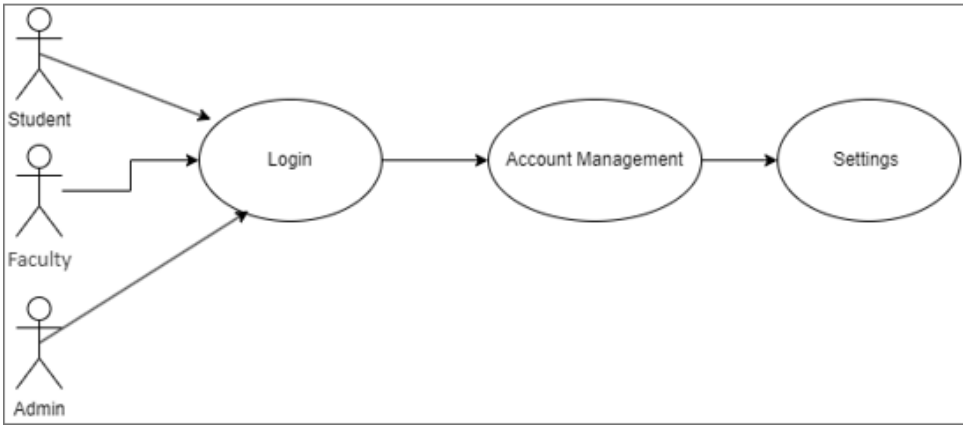
Database Design

- *Database Design: Description of database design along with Entity Relationship Diagram.*



Component Diagram

- *Component Diagram: Description of components and interactions between them.*



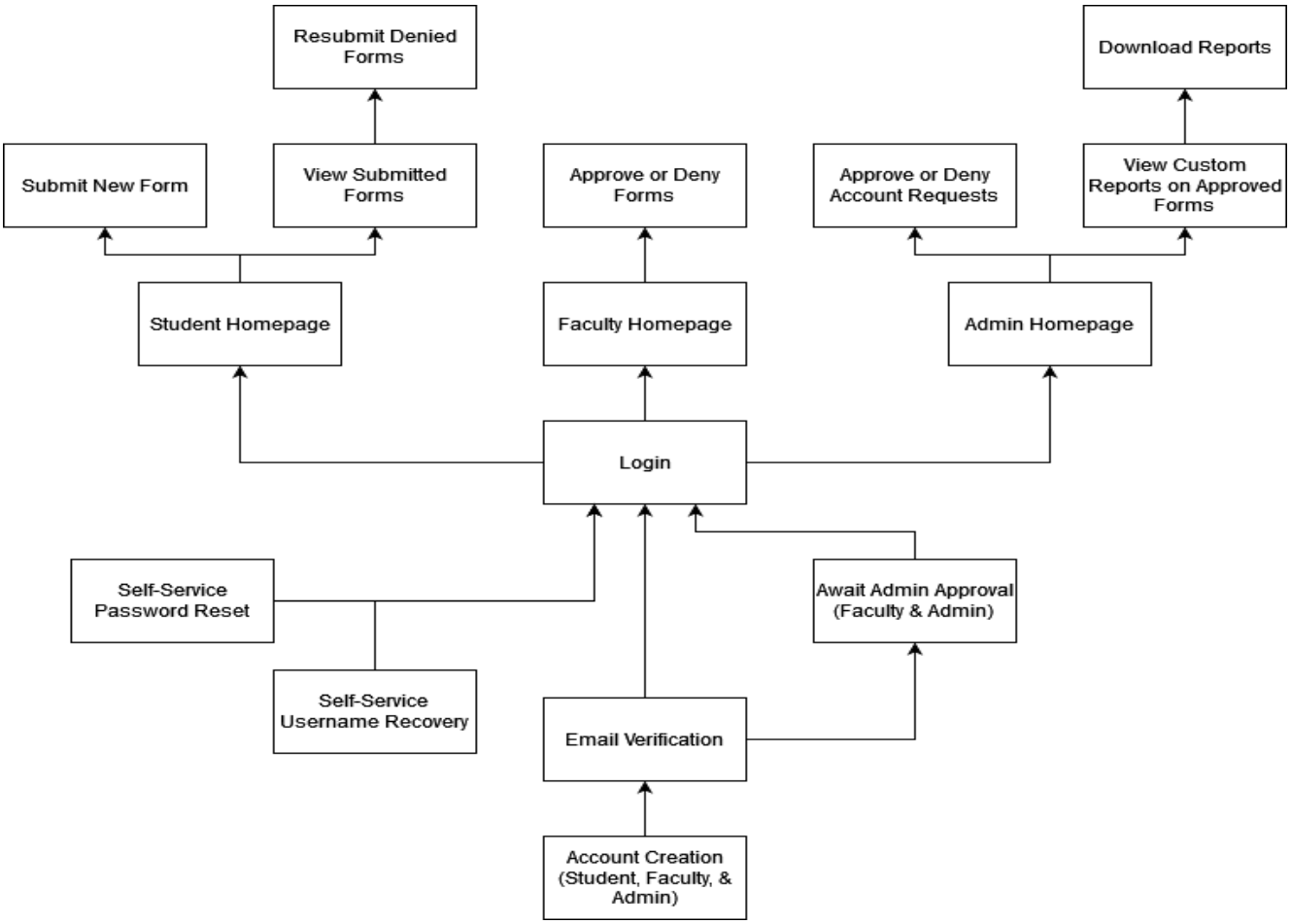
1. The faculty successfully signs into their account.
2. From the Faculty homepage the user selects the settings page.
3. On the settings page, the user can now edit their account information.

Deployment Diagram

- *Deployment Diagram: Description of deployment infrastructure, components, and where they are hosted.*

Deployment Plan:

1. The software will be provided in source code via GitHub.com.
2. The URL for the web application is - https://github.com/Ronatos/UNF-SoCHonors-in-Computing-Hours-Tracking. The code will be provided via the repository and can be cloned by the administrator of the deployment server.
3. The software is installed via cloning the GitHub repository. The packages needed are NPM, MySQL, and all package dependencies.
4. For the System requirements, please refer to the Hardware and Software Specification document.
5. By means of installation, the host will require a type of server that is needed to host a URL that is accessible to the web and runs a reverse proxy on port 3000.
6. The client will receive this document, the Hardware and Software Specification document, the database documentation for MySQL, and the GitHub repository.
7. We have a database creation script to create any table (Create new, Update, or delete), also make sure to use 'NPM Run Start' to start & run the application.
8. (1) - The server will be requested by the instructor with our specifications, (2) - The server will be built according to UNF (University of North Florida) Standards by ITS (Information Technology Services). (3) - Senior Project team members will ensure stable deployment of the application. (4) - Senior Project team members will provide documentation to UNF administrators.



Implementation

From August of 2022 to April of 2023 the CAPA Tech team has worked on administering a project for Dr. Sherif Elfayoumy. With the help of our project mentor, Jonathan Ohlrich, we successfully implemented our Hours Tracking Program for the UNF School of Computing. The technology stack used included Next.js, Express, React, and MySQL and while previously deployed on Microsoft Azure our project has been moved onto the UNF network with NGINX as a reverse proxy. Design solutions were implemented using Jira and GitHub along with Visual Studio Code to develop and organize any implementations.

Development requirements were received from Dr.Sherif Elfayoumy. The anticipated features included: Access Control for students, faculty, and administrators; proper activity workflow management that allows students to submit reports to faculty members, and the ability for administrators to download this information from the system. All of which were completed and delivered according to specifications provided. Over the development cycle, some features were scrapped to ensure a complete product is delivered, including a full notification system. This has been resolved as faculty members/administrators can send emails instead. A proper filter bar was scrapped due to time constraints although a thorough sorting and search bar function was implemented. A more complex account management system was also in place however it was

more fitting to only include password/account reset functionality. These are only some of the development constraints realized over the timeline. Listed below are all user stories and use cases developed over the project.

Work Completed (User Stories):

Be able to determine what types of notifications I receive
Be able to change my first and last name while logged in to the system
Be able to change my password while logged in to the system
Log in with my username and password
Recover my username without being logged in to the system
Reset my password without being logged in to the system
Register by creating a username and password, and selecting my role in the system
Verify my school email address
Receive an email notification when one of my submissions has been updated
Receive a notification in the system when one of my submissions has been updated
Be able to click a notification in the system to take me to the submission that has been updated
Resubmit my denied leadership hour submissions
View links to "Submit a New Form" and "Review Previously Submitted Forms" on my homepage
Submit a new leadership hour form
Have my status as a faculty member verified by an administrator
Receive an email notification when a student submits a form to me
Receive a notification in the system when a student submits a form to me
Be able to click a notification in the system to take me to a submitted form
View a list of student forms submitted to me on my homepage
View more details on a particular form by clicking it
Approve a form from my homepage by clicking an "Approve" button next to the entry
Deny a form from my homepage with or without a comment by clicking a "Deny" button next to the entry
Approve a form from the detailed form view by clicking an "Approve" button
Deny a form from the detailed form view with or without a comment by clicking a "Deny" button next to the entry
Have my status as a administrator verified by an existing administrator
Receive an email notification when a new faculty or administrator account is in need of verification
Receive a notification in the system when a new faculty or administrator account is in need of verification
Be able to click a notification in the system to take me to the faculty or administrator account verification
View links to "Account Verification Requests" and "Reports" on my homepage
View a list of account verification requests submitted to me on my Account Verification Requests page
Approve an account verification request from my Account Verification Requests page by clicking an "Approve" button next to the entry

Deny an account verification request from my Account Verification Requests page by clicking a "Deny" button next to the entry
Filter all entries with configurable values and ranges to generate a list of approved student form submissions
View a list of all approved student forms that match the conditions specified in the filter
Download an excel file with the data pulled in the currently viewed report
Log out of the system

Work Completed (Use Cases):

Update how/which notifications are received
Update first and last name while logged in
Update password while logged in
Create browser login session
Read username(s) associated with email without being logged in
Read password associated with username without being logged in
Create account with email, username, password, and role
Verify school email and Update account status
Send an email notification to students when one of their submissions is updated
Send a system notification to students when one of their submissions is updated
Link student system notifications to appropriate editable submissions
Update existing submissions as a student
Link student functions to the student homepage ("Submit a New Form" and "Review Previously Submitted Forms")
Submit new leadership hour form
Update Faculty account status to note verification by administrator
Send an email notification to faculty when a student submits a form to them
Send a system notification to faculty when a student submits a form to them
Link faculty system notifications to appropriate editable submissions
Display all student forms submitted to faculty member in need of approval on faculty homepage
Link forms listed on faculty homepage to appropriate editable submissions
Provide form approval functionality on faculty homepage form previews without needing to visit detailed editable form first
Provide form denial functionality on faculty homepage form previews without needing to visit detailed editable form first
Provide form approval functionality on detailed editable form submitted to faculty
Provide form denial functionality on detailed editable form submitted to faculty
Update Administrator account status to note verification by administrator
Send an email notification to administrators when a new faculty or administrator account request is submitted
Send a system notification to administrators when a new faculty or administrator account request is submitted
Link listed account request forms to appropriate detailed account request forms

Link administrator functions to the administrator homepage ("Account Verification Requests" and "Reports")
Display all account creation requests submitted to administrators in need of approval on Account Verification Requests webpage
Provide account request approval functionality on Account Verification Requests webpage request previews without needing to visit detailed account requests first
Provide account request denial functionality on Account Verification Requests webpage request previews without needing to visit detailed account requests first
Allow administrator to filter all approved reports by specific values as desired or search for specific reports (by semester, student, etc.)
Provide administrator with a way to view all filtered reports/data
Provide administrator with a way to export report data in an excel file
Delete browser login session

Test Case Results

The test case results section should provide details on results on tests ran for each scenario identified in the Use Case Specifications. This section should contain the completed test case table used in deliverable 6 along with test results and relevant descriptions for each table. If there are some peculiar or unexpected results, provide your explanations for those.

Close to the end of the development cycle the CAPA Tech team had created a list of test cases to stress test the system and ensure working quality of the project. Following the test case results the product was fully debugged and corrected reflecting these test cases. Use cases tested include:

Test Case

Use Case: Student Form Submission.

By: Christopher Clark

UC Step	**Step Description/ Condition**	**Data values**	**Expected Result**	**Actual Result** (If different from expected)	**Successful/ Failed**	**Environment Nbr (if failed)**	**Log Nbr (if failed)**
Basic flow step 1	User Clicks Submit Leaders hours button on the student homepage.	N/A	User gets redirected to the student form page.	(Same as expected)	Successful	N/A	N/A

Basic flow step 2	User fills out all input fields on the form.	Event name, date, hours amount, professor name and any comments.	The student is able to fill in all input fields correctly, with no errors.	(Same as expected)	Successful	N/A	N/A
Basic flow step 3	User Clicks the submit button.	N/A	The form will be submitted, and the user will get redirected to the homepage.	(Same as expected)	Successful	N/A	N/A
Basic flow step 4	After clicking the submit button.	N/A	The user is redirected back to the student homepage; Now a table is populated with the student forms, with more details for the student.	(Same as expected)	Successful	N/A	N/A
				Test Case Status	Successful/ Failed		

Test Case Use Case: Account Creation

By: Alexander Perez

Test Cases

UC Step	Step Description/ Condition	Data values	Expected Result	Actual Result (if different from expected)	Successful/ Failed	Environment Nbr (if failed)	Log Nbr (if failed)
Basic flow step 1	Student, Faculty, or Administrator reaches the Login webpage	https://honorshourtracking.ccec.unf. edu/	200 ok		Successful		
Basic flow step 2	Student, Faculty, or Administrator navigates to the Account Creation webpage through a link on the Login webpage	https://honorshourtracking.ccec.unf. edu/account_creat ion/	200 ok		Successful		
Basic flow step 3	Student, Faculty, or Administrator		Form is editable		Successful		
	clicks the UNF Email Address text input box						
Basic flow step 4	Student, Faculty, or Administrator fills in their UNF email address.	Must fit regex: /^n\d{8}@unf\.ed u$/i	Form submission is accepted		Success dependent on user entry		
Basic flow step 5	Student, Faculty, or Administrator clicks the Username text input box.		Form is editable		Successful		

Basic flow step 6	Student, Faculty, or Administrator fills in their preferred username.	Any value <255 characters	Form is editable		Successful		
Basic flow step 7	Student, Faculty, or Administrator clicks the Password text input box.		Form is editable		Successful		
Basic flow step 8	Student, Faculty, or Administrator fills in their preferred password.	Any value < 255 characters	Form is editable		Successful		
Basic flow step 9	Student, Faculty, or Administrator selects their role in the system from one of: Student, Faculty, or Administrator.		Form radio buttons are selectable		Successful		
Basic flow step 10	Student, Faculty, or Administrator clicks the submit button		Form is submittable		Successful		
	to create their account.						
Basic flow step 11	System verifies that the inputted email address matches regex /^n\d{8}@unf\ .edu$/i	Email address	Regex is matched		Success dependent on user entry		

Basic flow step 12	System verifies that the inputted username is not already taken.	Username	Username is unique		Success dependent on user entry		
Basic flow step 13	System creates a new account entry in the database for the new Student, Faculty, or Administrator with an account status noting that the Student, Faculty, or Administrator has not yet verified their email address.	`{ username: string, email: n01234567@unf .edu, password: string, role: 'student', 'faculty', or 'admin', first_name: string, last_name: string, }`	User values submitted to database		Successful		
				Test Case Status	Successful		

Use case: Faculty Homepage – Approval and denial

By: Sophia Abuzeni

UC Step	Step Description/ Condition	Data values	Expected Result	Actual Result (if different from expected)	Successful/ Failed	Environment Nbr (if failed)	Log Nbr (if failed)
Basic flow step 1	Faculty Logs into Account	Username and Password	Log in	Log in	Successful -- dependent on user entry		
Basic flow step 2	Faculty directed to Homepage	Student submitted data and approved or denied buttons	View of homepage	View of homepage	Successful		
Basic flow step 3	View of Student submitted forms	Student submitted data and approved or denied buttons	Homepage –view of student submitted forms	Homepage –view of student submitted forms	Successful		
Basic flow step 4	Faculty can approve or deny	Student submitted data and approved or denied buttons	A pop up that confirms if they want to approve or deny	A pop up that confirms if they want to approve or deny	Successful		
Basic flow step 4	Faculty can filter through submitted forms	Filter of data	A search bar where user can search for form	A search bar where user can search for form	Successful -- depending on if there is data		
Basic flow step 4	Nothing to work on page	No more data to approve or deny	A message appears that lets the user know there are no more forms to review	A message appears that lets the user know there are no more forms to review	Successful -- when no more forms to review		
				Test Case Status	Successful/ Failed		

Test Case

Use case: Reporting / Account Verification

By: Chance Abenes

Test Case							
UC Step	**Step Description / Condition**	**Data values**	**Expected Result**	**Actual Result** (if different from expected)	**Successful/ Failed**	**Environment Nbr (if failed)**	**Log Nbr (if failed)**
Basic flow step 1	Admin logs into system	Username Password	Log In	Log In	Successful		
Basic flow step 2	Admin is directed to Homepage	Buttons to navigate to Database or Account verification	Homepage direction	Homepage	Successful		
Basic flow step 3	View Account Approval Page	Faculty/Admin Account Information	Succesful direction of account approval page	Succesful direction of account approval page	Successful		
Basic flow step 4	Admin can approve or deny account creation	Faculty/Admin Account Information	Pop up confirming Approval/ Denial of account	A pop up that confirms if they want to approve or deny	Successful		

Basic flow step 4	Admin successfully approves / denies after confirmation	Faculty/Admin Account Information	Page reloads and removes the approved/ denied account	Page reloads and removes the approved /denied account	Successful		
				Test Case Status	Successful/ Failed		

The noted test cases above were tested on local machines and were mostly successful. However, on the deployed application the team encountered many bugs. Since our test case meeting with Dr. Elfayoumy all test case results listed below have been debugged and corrected as our product stands it is fully complete and functioning. Below is a list of all test use cases and their results all of which are also listed in the test case results document.

TEST SUMMARY

The program was walked through with the client from the creation of fresh accounts to the exploration of every feature in the program. Many features were tested, and the results are described below.

Project Name: UNF SoC Honors in Computing Hours Tracking

2.1 ACCOUNT CREATION

Student, Faculty, and Administrator accounts were created.

Test Owner: Alexander Perez

Test Date: 03/30/2023

Test Results: Accounts are being created successfully.

2.2 STUDENT HOMEPAGE (EMPTY)

A student account was created, and the user was redirected to the Student Homepage.

Test Owner: Christopher Clark

Test Date: 03/30/2023

Test Results: After creating an account and verifying account, the user was then redirected to the Student Homepage with an empty field.

2.3 STUDENT HOMEPAGE (1 OR MORE)
A student account was created.

The user was redirected to the Student Homepage.

1 or more forms submitted. Then, the homepage has a table with multiple forms populated.

Test Owner: Christopher Clark.

Test Date: 03/30/2023.

Test Results: After creating a single form, the user is redirected to their homepage and a table is populated with information.

2.4 STUDENT FORM SUBMISSION
Student hits the submit button on the Honors Tracking form.

Test Owner: Christopher Clark

Test Date: 03/30/2023

Test Results: Faculty enumerating denied accounts, and faculty enumeration lists first names twice.

2.5 STUDENT FORM EDIT BUTTON
On the Student Homepage, there is an action column on the table. In the action column there is a button to edit form(s).

Test Owner: Christopher Clark

Test Date: 03/30/2023

Test Results: Form did not work as the client intended. Therefore, we will be using a comment solution system. The faculty will now deny student forms and send them a comment about what was wrong etc.

Additional Messages: After the client meeting, we came to a conclusion to remove the student form edit button. As the client's suggestion, we will be moving to a comment system, when faculty are denying forms.

FACULTY HOMEPAGE
A faculty account was created.

Admin approval required.

The faculty was redirected to the Faculty Homepage.

The queue of student submitted forms are displayed.

Faculty is given an option to approve or deny forms and add a comment. **Test Owner**: Sophia Abuzeni **Test Date**: 03/30/2023.

Test Results: A popup should appear where user will confirm decision. Comment is still being implemented.

FACULTY APPROVAL OR DENIAL POP UP

Once faculty approves or denies, a pop up appears which confirms their decision. **Test Owner**: Sophia Abuzeni **Test Date**: 03/30/2023.

Test Results: After faculty approves or denies the pop up the table is reloaded, and the form is removed from the queue.

Cancel button will remove popup window and keep data in table.

2.6 EMAIL VERIFICATION

Student, Faculty, and Administrator accounts were created, and emails were supposed to be sent automatically to verify the email addresses used.

Test Owner: Alexander Perez

Test Date: 03/30/2023

Test Results: Emails are not being sent due to an issue with the sending service, but all other functionalities of the system appear functional.

2.7 USERNAME RECOVERY

An existing email was selected to send a recovery email to.

Test Owner: Alexander Perez

Test Date: 03/30/2023

Test Results: Emails are not being sent due to an issue with the sending service, but all logs indicate that the system is functional otherwise.

2.8 PASSWORD RESET

An existing account was selected to send a password recovery email to.

Test Owner: Alexander Perez

Test Date: 03/30/2023

Test Results: Emails are not being sent due to an issue with the sending service, but logs indicate that the password reset is functional otherwise.

2.9 STUDENT LOGIN
Student account logs in to the system and gets to the appropriate homepage for them.

Test Owner: Alexander Perez

Test Date: 03/30/2023

Test Results: System functional

2.10 FACULTY LOGIN
Faculty account logs in to the system and gets to the appropriate homepage for them.

Test Owner: Alexander Perez

Test Date: 03/30/2023

Test Results: System functional

2.11 ADMIN LOGIN
Administrator account logs in to the system and gets to the appropriate homepage for them.

Test Owner: Alexander Perez

Test Date: 03/30/2023

Test Results: System functional

TEST ASSESSMENT

Everything was tested adequately, and all issues with the product were identified. Additionally, the client had an opportunity to contribute to how they'd like a solution implemented.

TEST RESULTS

- Email Verification o Emails not being sent
- Student Homepage (1 or more forms) o Left justify the comments
- Student Form Submission o Faculty enumerating denied accounts o Faculty enumeration lists first name twice
- Student Form Resubmission o Handling resubmission
- Faculty Homepage (1 or more forms) o Left justify the comments

o We need a third option: send back for corrections

- Faculty Form Denial o ERCONNRESET
- Faculty Form Filtering o Use state to reload the list so the filter remains
- Administrator Homepage o Not functional
- Administrator Account Requests (1 or more requests) o Filtering on “Status Pending”, not the previously agreed upon “pending admin approval” o Role and Status are flipped o Deny is not functional
- Administrator Reports o Incorrect fields displayed o Logout not functional
- Administrator Report Filtering o Filtering not functional

Project Installation and Delivery

Installation Guide

1. The software will be provided in source code via GitHub.com.
2. The URL for the web application is - https://github.com/Ronatos/UNF-SoCHonors-in-Computing-Hours-Tracking. The code will be provided via the repository and can be cloned by the administrator of the deployment server.
3. The software is installed via cloning the GitHub repository. The packages needed are NPM, MySQL, and all package dependencies.
4. For the System requirements, please refer to the Hardware and Software Specification document.
5. By means of installation, the host will require a type of server that is needed to host a URL that is accessible to the web and runs a reverse proxy on port 3000.
6. The client will receive this document, the Hardware and Software Specification document, the database documentation for MySQL, and the GitHub repository.
7. We have a database creation script to create any table (Create new, Update, or delete), also make sure to use ‘NPM Run Start’ to start & run the application.
8. (1) - The server will be requested by the instructor with our specifications, (2) - The server will be built according to UNF (University of North Florida) Standards by ITS (Information Technology Services). (3) - Senior Project team members will ensure stable deployment of the application. (4) - Senior Project team members will provide documentation to UNF administrators.

User Manual

-

Application Prerequisites for Production Use

- Server hosting the application must be running
- MySQL service must be running on the server hosting the application
- Application database must contain the tables described in the build_tables.sql file in the application db directory
- NGINX must be acting as a reverse proxy for port 3000
- Application must be running on port 3000
- If no other administrator accounts exist which can be used to approve new accounts, the very first administrator account created must be manually set to 'active' in the accounts table in the production database instead of the normal administrator account approval process
- Once the application is running, it can be reached at https://honors-hour-tracking.ccec.unf.edu/
- The site certificate is self-signed, and most modern browsers will warn about this the first time accessing the site

-

Account Creation

- Once the application is running and the root login page is reached, a link to the account_creation page will be found at the bottom of the login page.
- Username must be unique.
- Email address must be an accessible UNF n-number@unf.edu. Email aliases like a.employee@unf.edu will not work. This is important to limit the publicly accessible website to logins by active UNF faculty, staff, and students. Email addresses do not need to be unique. You may have more than one account per email address.
- Passwords have no requirements enforced.
- First and last names are required, but are not validated against any UNF database. Students should be expected to treat this website seriously, as they will be interacting with their instructors and school administrators with it.

-

-

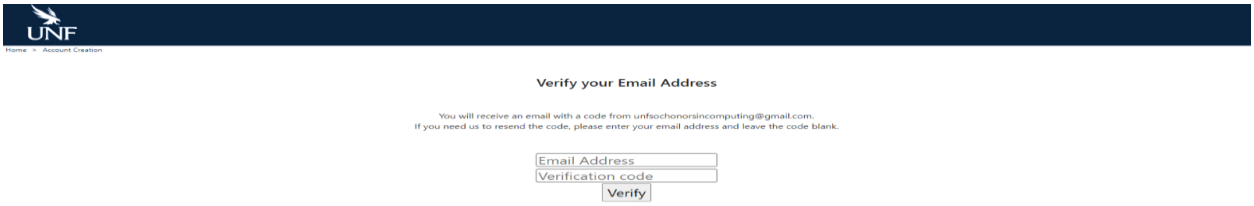
-

- **Email Verification**
    - Once accounts have been created in the system, the associated email address will need to be verified. You should be automatically redirected to the email_verification page, but if you are not redirected automatically or you find yourself needing to verify your email at another time, a link exists at the bottom of the account_creation page to get there manually.
    - An email is automatically sent to the email address used to create your account. Due to a combination of UNF email policies and the email service chosen for the application, this email will always go to your junk folder. If you don't receive it, verify your email rules are not preventing you from receiving an email from unfsonhonorsincomputing@gmail.com, and resend it using the instructions on the email_verification page.
    - Once submitting your email address and the code you received in the email_verification form, your account should proceed through the email verification process. Student accounts only require email verification. Faculty and Administrator accounts are subject to additional approval.

    -

-

- Faculty and Administrator accounts will now show up in the account_requests Administrator page. An administrator with existing access will need to approve these accounts for them to move to the active status. The accounts are deleted on administrator denial with no notification to the requestor of the account.



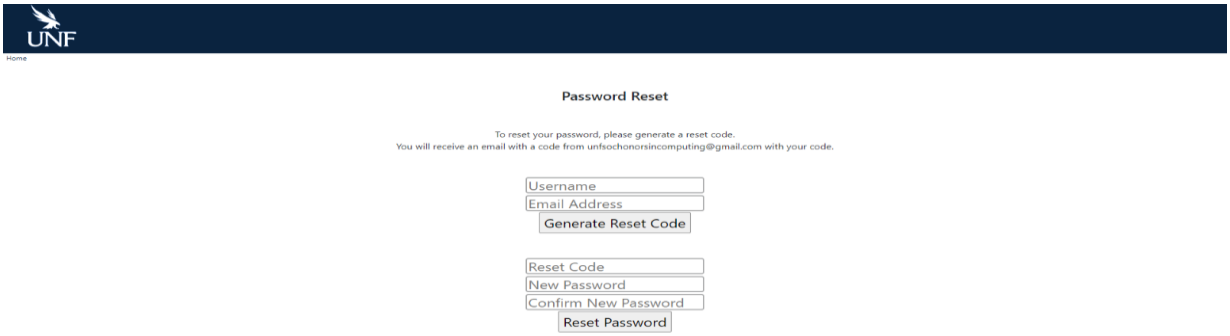
-

- **Username Recovery**

- If you cannot remember your login username, you may recover it by using the appropriate link on the root login page and entering your email address associated with the account in the form provided.
- All account usernames associated with that email address will be emailed to you.

- **Password Reset**

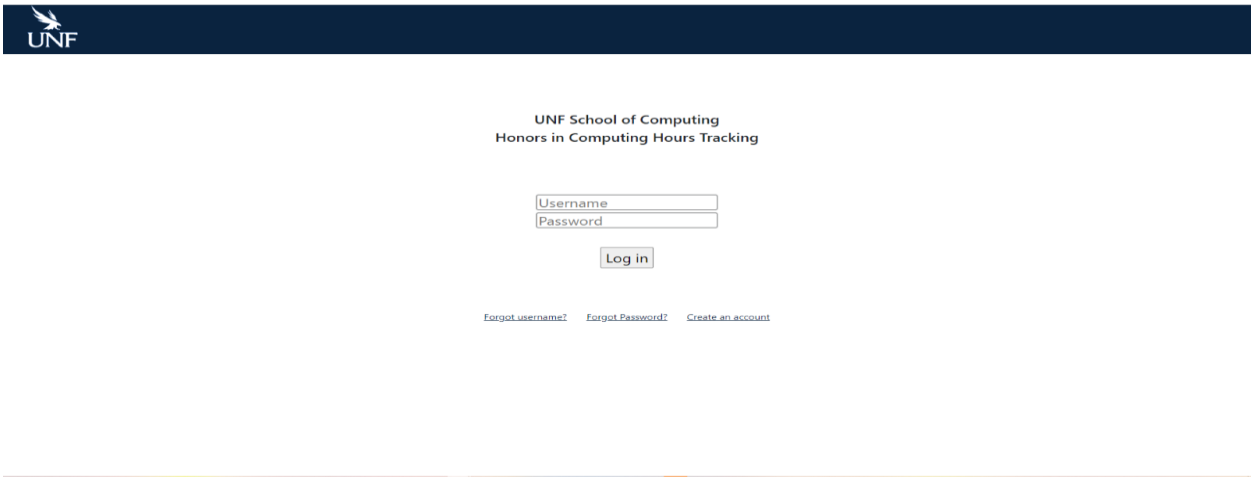
  - If you cannot remember your login password, you may reset it by using the appropriate link on the root login page and entering your email address and username in the form at the top of the page.
  - If the credentials match, a code to reset the password for that account will be sent to your email address. Submit the code you received along with the new password you'd like to use from now on. If done successfully, your password will be reset. You do not need to hold on to that code you were sent.



-

  - **Login**

  - Simply enter your username and password once your account has been created and is active. If you do not know how to activate your account, please contact an administrator.
  - Once login is successful, a cookie will be stored in your browser readable only to the application over an https connection. It stores your username and your role. This is used to keep track of who you are as you use the website. Login will automatically redirect you to your homepage. The logout button will delete the cookie, and you will be taken back to the login page.


UNF
UNF School of Computing
Honors in Computing Hours Tracking
Username
Password
Log in
Forgot username?
Forgot Password?
Create an account

- **School of Computing: Honors Tracking System**

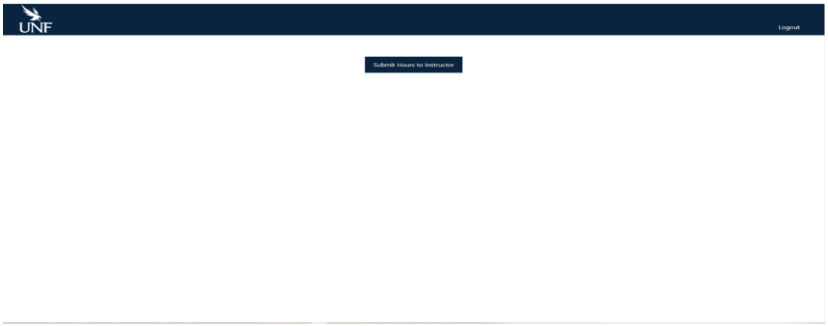
- **Student Homepage**

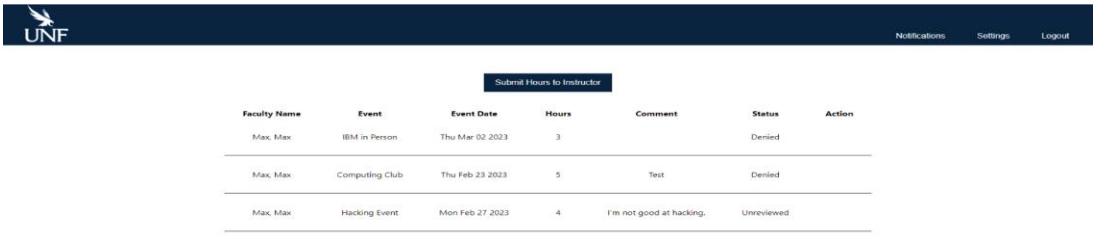
- **<u>Prerequisites for Production use of the system:</u> The user must make an account and verify their account as a student. This way, the user will be redirected to the Student Homepage, once signing into their account.**

- ***<u>Student Homepage:</u>***

  - **• Once being redirected to the Student Homepage, the user can click the "Submit Honors" button.**

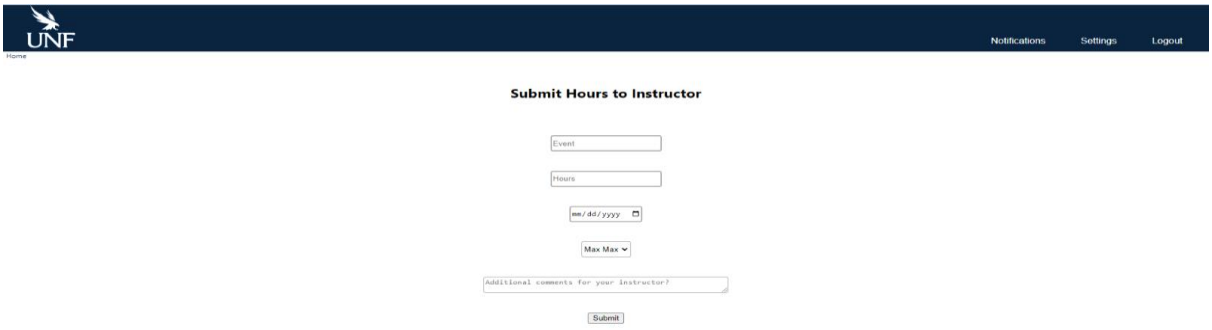
  - Now the user is redirected to the student submitting form page.
    - When 1+ form has been submitted, the user is redirected to the student homepage and a table will be created with 1+ forms, showing the approved or denied information, about each form.

  - 

- 

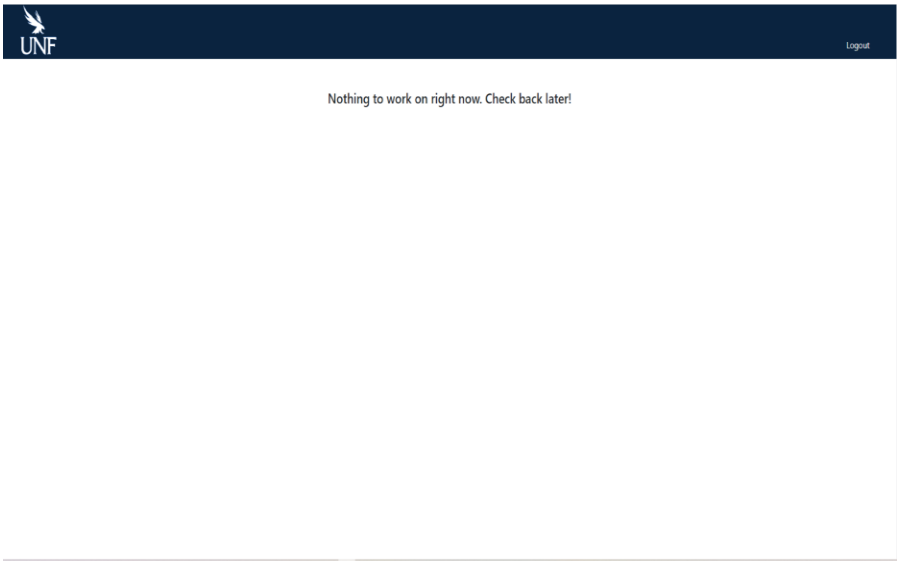
-
-

- ***<u>Student Submit Form Page:</u>***
- **• Once being redirected to the Student Homepage, the user can click the "Submit Honors" button.**
- Now the user is redirected to the student submitting form page.
    - On this page, the user can enter information and submit their form to a faculty member. The information needed on this page is the Events name, number of hours, the date of the event, the faculty member's names, and any comments the student has.



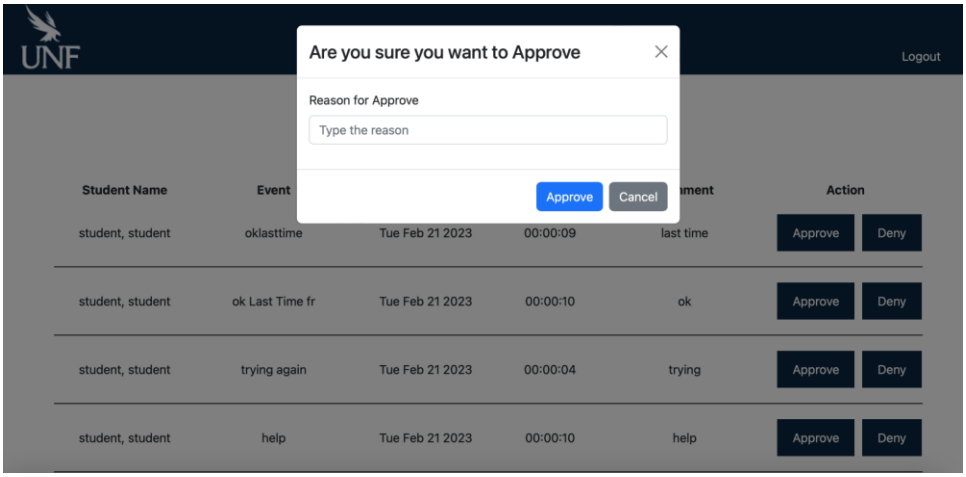
-
-

-
    - **School of Computing: Honors Tracking System**
        - **Faculty Homepage**

- **<u>Prerequisites for Production use of the system:</u> The faculty member must make an account and verify their account as a faculty. This way, the user will be redirected to the Faculty Homepage, once signing into their account.**
- ***<u>Faculty Homepage:</u>***
    - **• Once being redirected to the Faculty Homepage, the faculty member can approve or deny submitted forms.**
        - Once the faculty member clicks approve or deny they are prompted to add a comment and confirm their decision.
        - The decision of the form and the comment will appear on the students account where they can view the reasoning.
        - 

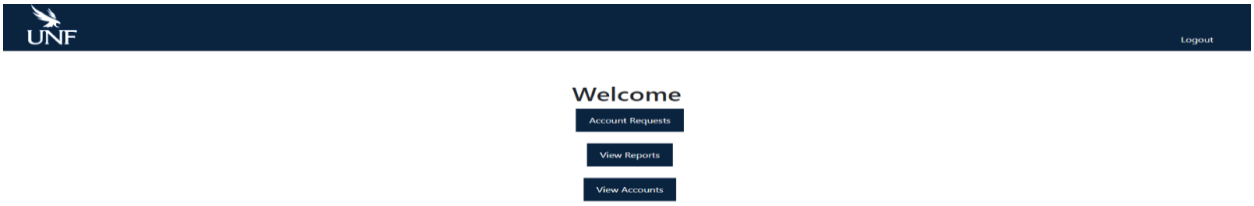
        - **• Faculty members can also search for submitted forms by searching student name or event name.**
        - Faculty member can search for student by first or last name.
-


UNF
Logout
Search Submitted Forms
Student Name
Event
Event Date
Hours
Comment
Action
clark, chris
UNF Computing
Tue Apr 18 2023
4
Also did the extra paper work, I can send it to your work email.
Approve
Deny


UNF
Logout
Are you sure you want to Approve
Reason for Approve
Type the reason
Approve
Cancel
Student Name
Event
Action
student, student
oklasttime
Tue Feb 21 2023
00:00:09
last time
Approve
Deny
student, student
ok Last Time fr
Tue Feb 21 2023
00:00:10
ok
Approve
Deny
student, student
trying again
Tue Feb 21 2023
00:00:04
trying
Approve
Deny
student, student
help
Tue Feb 21 2023
00:00:10
help
Approve
Deny

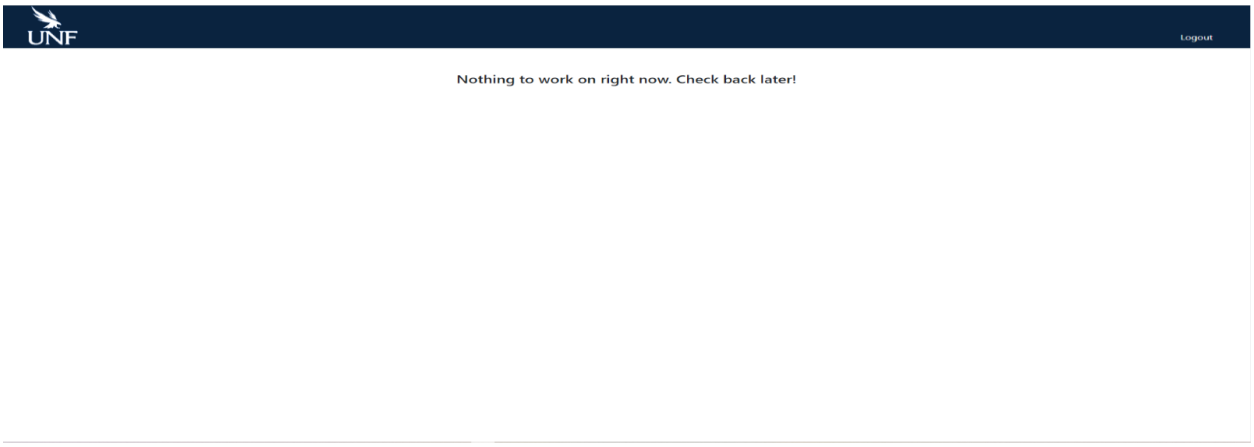
- **Admin Homepage**

-
- **<u>Prerequisites for Production use of the system: The user must make an account and verify their account as a student. This way, the user will be redirected to the Admin Homepage, once signing into their account.</u>**
-
- ***<u>Admin Homepage:</u>***
- **• Once being redirected to the Admin Homepage, the user can click 3 different button links: “Account Requests,” “View Reports,” and “View Accounts.”**



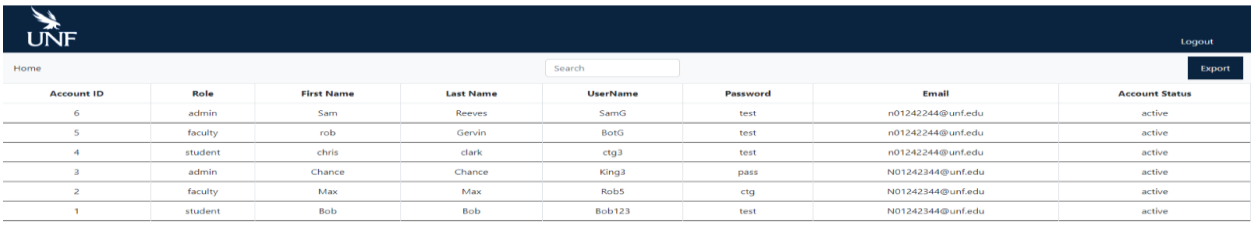
-
- ***<u>Account Requests</u>***

- After clicking on the Account Requests button, the user will be greeted with a table showing all accounts pending admin approval. If there are no available accounts, the table will be empty.
    - If there is an available account, the user will be able to either approve or deny said account.
    - Clicking approve will change the account status to approved, allowing access to the system.
    - Clicking deny will remove the account from the system.



- ***View Reports***

- After clicking on the View Reports button, the user will be greeted with a table showing all approved reports within the system. If there are no available reports, the table will be empty.
    - Here, the user may click on the headers to sort the table by the desired attribute. Ex. Clicking “Event Name” will sort the table by Event Name alphabetically.
    - The user may also manually search for reports by typing into the search bar.
    - The user may also click the Export button. This button will export all entries shown in the table. If there are no filters applied every single report will be exported.



Account ID	Role	First Name	Last Name	UserName	Password	Email	Account Status
6	admin	Sam	Reeves	SamG	test	n01242244@unf.edu	active
5	faculty	rob	Gervin	BotG	test	n01242244@unf.edu	active
4	student	chris	clark	ctg3	test	n01242244@unf.edu	active
3	admin	Chance	Chance	King3	pass	N01242344@unf.edu	active
2	faculty	Max	Max	Rob5	ctg	N01242344@unf.edu	active
1	student	Bob	Bob	Bob123	test	N01242344@unf.edu	active

- ***View Accounts***

- After clicking on the View Accounts button, the user will be greeted with a table showing all accounts within the system. If there are no available accounts, the table will be empty.
    - Here, the user may click on the headers to sort the table by the desired attribute. Ex. Clicking “First Name” will sort the table by First Name alphabetically.
    - The user may also manually search for accounts by typing into the search bar.
    - The user may also click the Export button. This button will export all entries shown in the table. If there are no filters applied every single entry will be exported.

Delivery Process

- A meeting with client occurred on May 1st where they were trained on how to use the product.
- The software will be provided in source code via GitHub.com.
- The URL for the web application is - https://github.com/Ronatos/UNF-SoCHonors-in-Computing-Hours-Tracking. The code will be provided via the repository and can be cloned by the administrator of the deployment server.
- The software is installed via cloning the GitHub repository. The packages needed are NPM, MySQL, and all package dependencies.
- Server hosting the application must be running
- MySQL service must be running on the server hosting the application
- Application database must contain the tables described in the build_tables.sql file in the application db directory
- NGINX must be acting as a reverse proxy for port 3000
- Application must be running on port 3000
- If no other administrator accounts exist which can be used to approve new accounts, the very first administrator account created must be manually set to 'active' in the accounts table in the production database instead of the normal administrator account approval process
- Once the application is running, it can be reached at https://honors-hour-tracking.ccec.unf.edu/
- The site certificate is self-signed, and most modern browsers will warn about this the first time accessing the site

Conclusion

Contributions and Limitations

- *Contributions and Limitations: Provide a discussion on the advantages and disadvantages of the software developed through this project. What aspects of your project (requirements, solutions, and implementation) are satisfactory? What aspects of your project (requirements, solutions, and implementation) are deficient?*

Advantages

- *All features of software application works perfectly*
  - *Approval and denial works*
  - *Search bars works for filtering*
  - *Login, password reset, and email verification works*
  - *Student submission works*
  - *View/Export reports works*
  - *Account approval/denial works*

Disadvantages

- The password reset is not a traditional password reset however, it works.
- Emails go to junk because it is not from UNF email.
- We needed to use send grid instead of UNF email tenant so we needed to create our own email.
- Reports are not displayed as first name, last name.

Benefits to Community Partner

- The Honors in Computing program is in development, and one of the new requirements of the program is a system which can keep track of hours students spend on different events which should count toward their degree progress. A system was needed to allow students, faculty, and administrative staff to interact with these submitted hours and approve them, deny them, or run reports on them. No existing system was in competition with our project. We created a custom application to fill a need that had no previous answer. Our Project will address all the needs identified above. This includes but is not limited to a system which allows students to submit reports of events they've attended for which they'd like to receive academic credit, a system which allows faculty to approve or deny said reports with comment for administrators or a comment for students, and a system which allows administrators to view and download those reports.

- **Future Work**
- We plan to stay in touch with the School of Computing in case issues arise where they may need help with problems that may surface.
- We will be sending our user manual.
- ITS will have the ability to make changes or fix issues.

Summary

- UNF School of Computing's Honors in Computing Hours Tracking. The School of Computing is a part of the College of Computing, Engineering, and Construction and our community partner is Dr. Sherif Elfayoumy.
- The project's aim is to automate the process of tracking honors students' leadership hours. This involves three key steps: First, enabling students to fill out and submit their activity reports; second, allowing faculty to approve or reject student-submitted requests; and finally, giving the school administration the ability to review individual student records and generate end-of-semester reports.
- **User Stories and Non-Functional Requirements:**
- *User Stories:*
- - Login

- - Account creation
- - Account verification
- - Navigation
- - Student submitting form
- - Faculty reviewing student forms
- - Reporting
- - Logout
- - Student, Faculty, Admin Homepage
- *Non-Functional requirements:*
    - Password Security
    - End of semester reports
    - Different account roles
- **Functional Requirements**
- The primary objective of the project is to streamline the procedure for monitoring the leadership hours of honors students by automating the process. This includes three significant steps: Firstly, providing the facility to students to fill out and submit their activity reports; secondly, permitting faculty members to approve or decline student-submitted requests; and lastly, enabling the school administration to scrutinize individual student records and generate reports at the end of each semester.
-