S.C.O.R.E

Milestone 1

FLORIDA TECH

FLORIDA'S STEM UNIVERSITY

Team

- Charlie Collins
- Tommy Gingerelli
- Logan Klaproth
- Michael Komar

Faculty Advisor/Client

- Dr. Mohan

Milestone 1

- Select tools for server implementation, web development, file transfer, and user authentication
  - Provide small demos of the tools
- Resolve technical challenges
- Create Requirement Document
- Create Design Document
- Create Test Plan

Milestone 1 - Completion Matrix

Task	Completion	Charlie	Logan	Michael	Tommy	To Do
Select Technical Tools	100%	25%	25%	25%	25%	N/A
Select Collaboration Tools	100%	25%	25%	25%	25%	N/A
Demos	100%	25%	25%	25%	25%	N/A
Resolve Technical Challenges	80%	25%	15%	40%	20%	Waiting on response from FIT IT
Requirements	90%	30%	20%	20%	30%	Requirements for Containers
Design Document	100%	20%	20%	20%	40%	N/A
Test Document	100%	50%	20%	15%	15%	N/A

Selecting Technical Tools

FLORIDA TECH
FLORIDA'S STEM UNIVERSITY

File Transfer

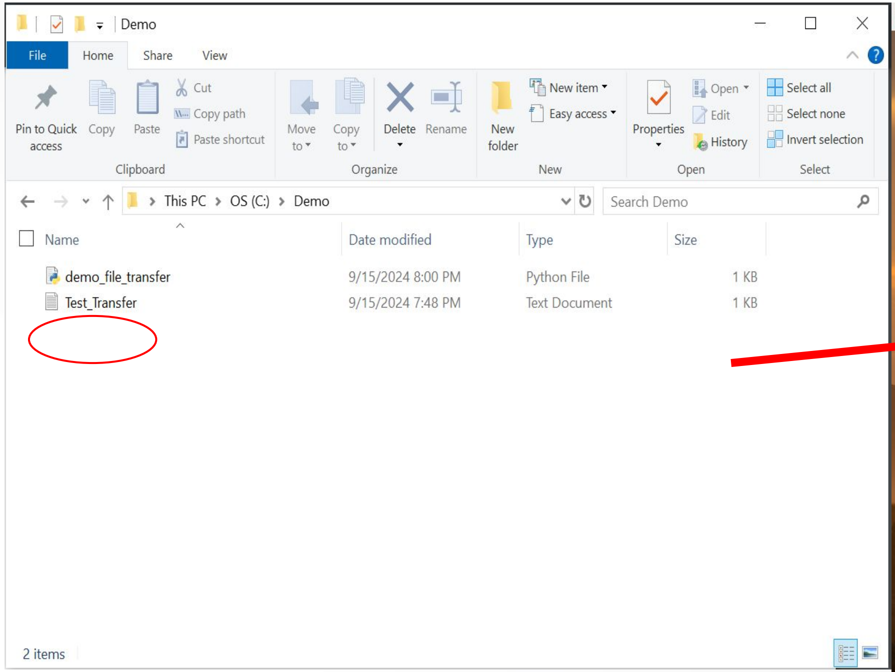
SFTP

- Application layer protocol
  - TCP Transport Layer
  - Works over SSH
- More robust
- Allows users to view and interact with files
  - View
  - Edit
  - Delete

SCP

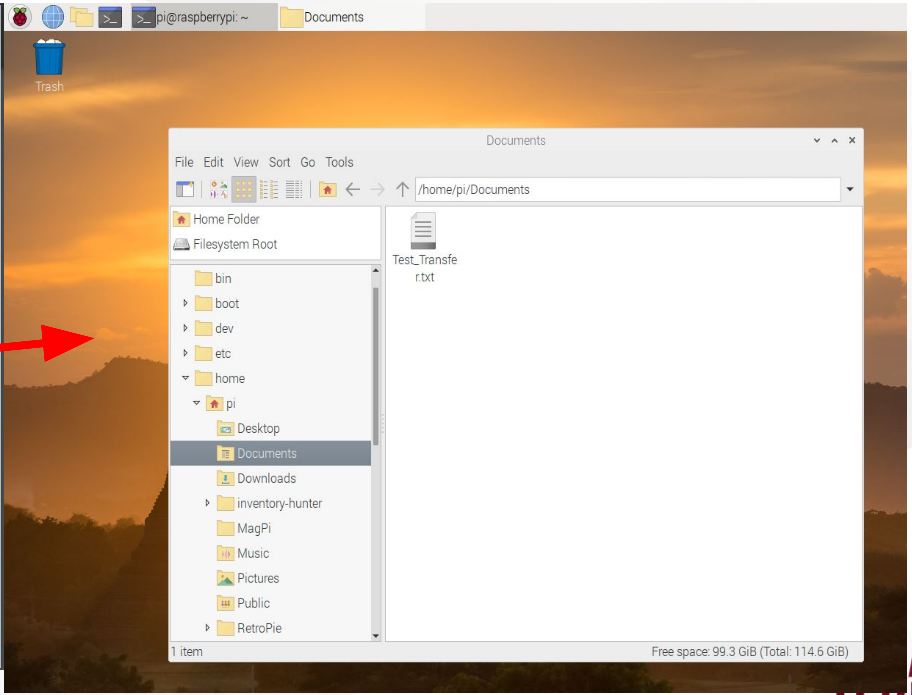
- Session layer protocol
  - TCP Transport Layer
  - Works over SSH
- Faster algorithm
- Can only copy files
- Deprecated in RHEL 9

SFTP Demo



Name	Date modified	Type	Size
demo_file_transfer	9/15/2024 8:00 PM	Python File	1 KB
Test_Transfer	9/15/2024 7:48 PM	Text Document	1 KB

2 items



User Authentication Tools

TRACKS CAS

- Implementing CAS into our system seems redundant
- Less scalable in the long term of the product

Google OAuth2

- Florida tech provides student Google accounts that are authenticated with CAS.
- More scalable, as our program would just be authenticating with any authorized Google Account.

Server Tools

Proxmox Virtual Environment

- Seamless deployment of the server virtual environment.
- Allows for snapshots of the development server to rollback in case of emergency.
- Can deploy multiple clones for load balancing purposes.

Ubuntu Virtual Machine

- Ubuntu was chosen for flexibility and wide application support among distributions.

Tailscale

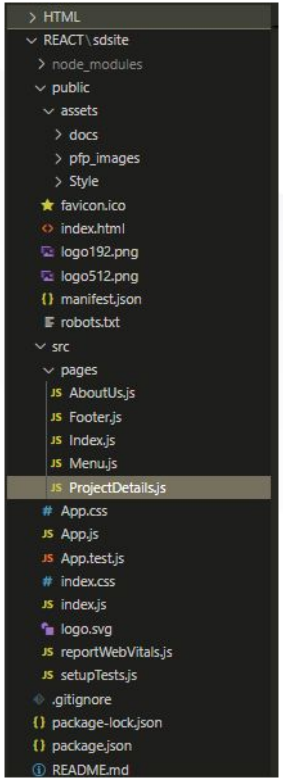
- Primary method of collaborative access to the development server.
- Simpler end user setup than WireGuard, and allows for uninterrupted free connection unlike similar apps like TeamViewer.

Web Application STACK

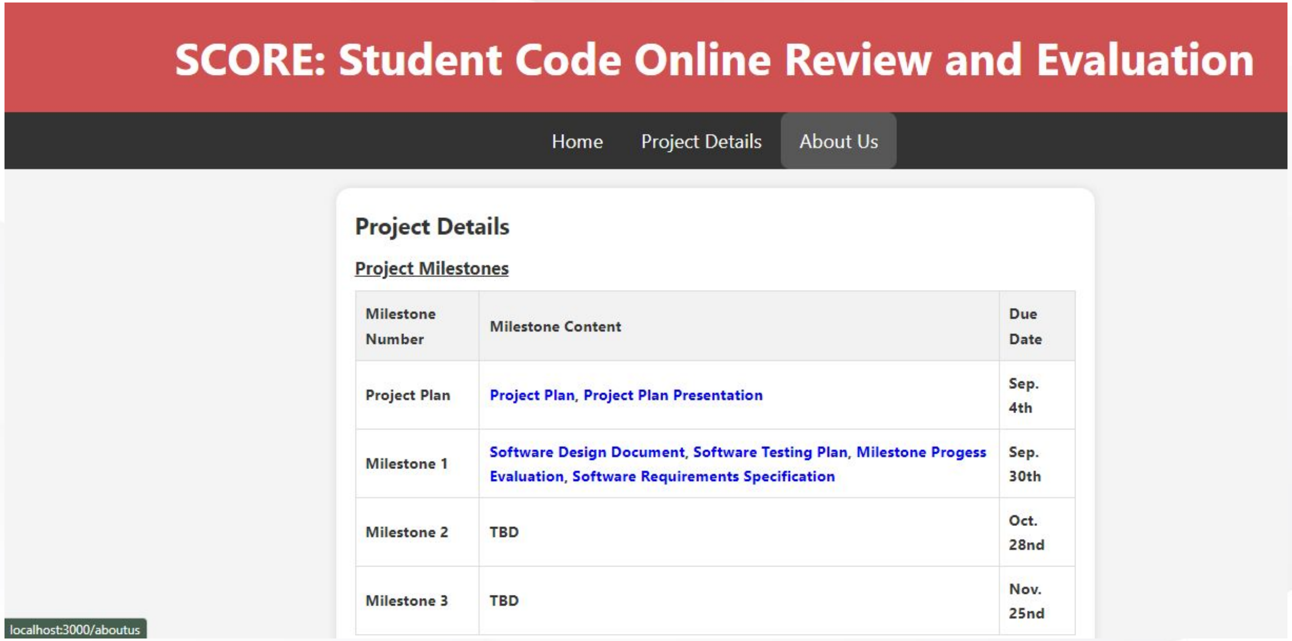
MERN - MongoDB Express.JS React Node.JS

- MERN stack offers simple and reliable application development
- Allows for storage and maintenance of large and complex data sets
- High performance with varying project scale and complexity
- Strong community and extensive documentation
- Familiar and easy to understand
- Avoids licensing issues

Web Stack Demo



```
> HTML
v REACT\sdsite
  > node_modules
  v public
    v assets
      > docs
      > pfp_images
      > Style
    favicon.ico
    index.html
    logo192.png
    logo512.png
    manifest.json
    robots.txt
  v src
    v pages
      AboutUs.js
      Footer.js
      Index.js
      Menu.js
      ProjectDetails.js
    App.css
    App.js
    App.test.js
    index.css
    index.js
    logo.svg
    reportWebVitals.js
    setupTests.js
  .gitignore
  package-lock.json
  package.json
  README.md
```



Project Details

Project Milestones

Milestone Number	Milestone Content	Due Date
Project Plan	Project Plan, Project Plan Presentation	Sep. 4th
Milestone 1	Software Design Document, Software Testing Plan, Milestone Progess Evaluation, Software Requirements Specification	Sep. 30th
Milestone 2	TBD	Oct. 28nd
Milestone 3	TBD	Nov. 25nd

REACT project running, successful routing between multiple components using react-router-dom

FLORIDA TECH
FLORIDA'S STEM UNIVERSITY

Containers

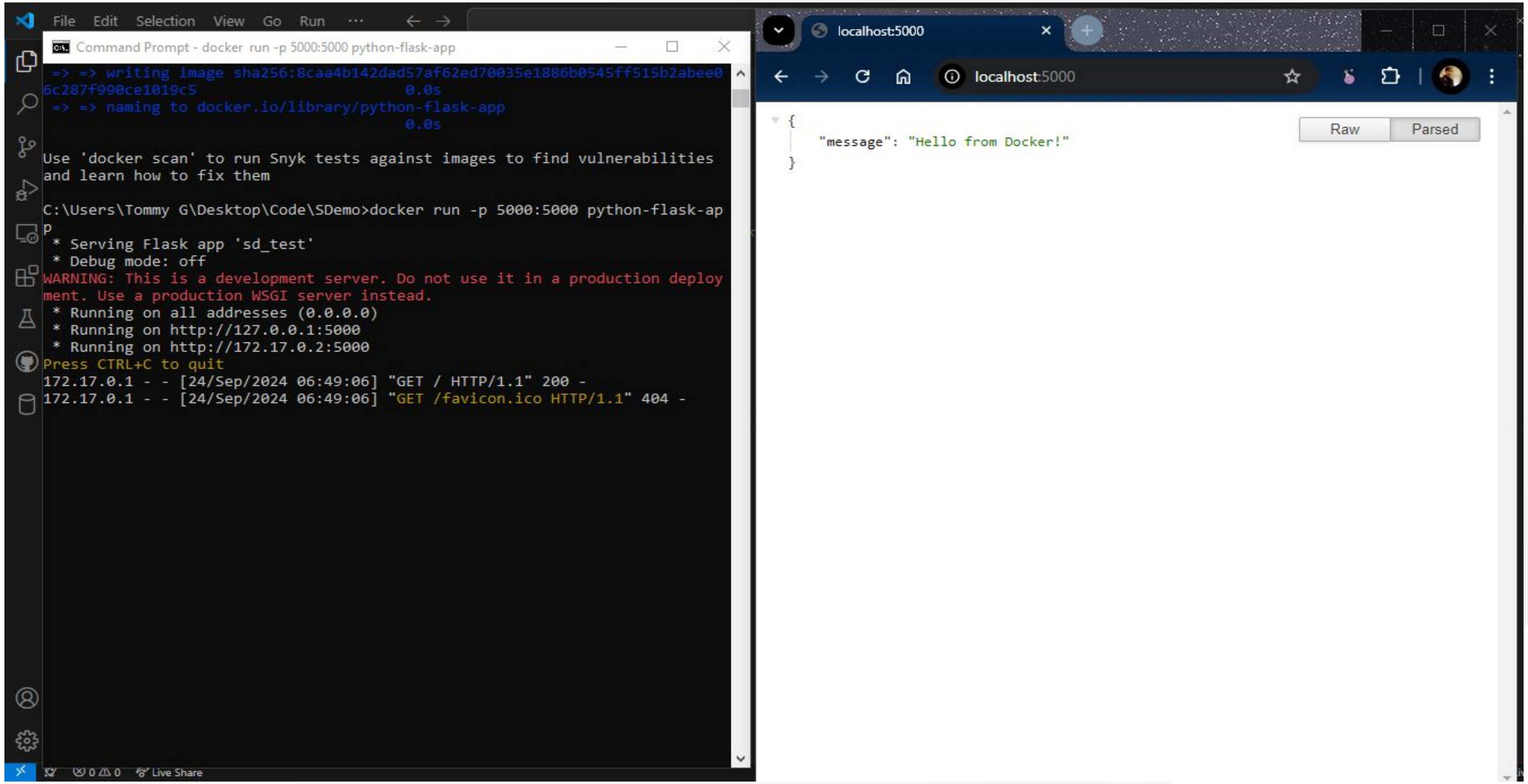
Docker

- Multi-platform, requires root permissions by default
- Innate client-server architecture, less secure
- Custom network stack

Podman

- Used less in industry
- More security focused, less overhead, quicker startup times
- Daemon-less, rootless by default, only linux
- Uses standard linux network stack

Docker Demo



File Edit Selection View Go Run ···
Command Prompt - docker run -p 5000:5000 python-flask-app
localhost:5000
Raw
Parsed
Live Share

```
 => => writing image sha256:8caa4b142dad57af62ed70035e1886b0545ff515b2abee0
6c287f990ce1019c5                                          0.0s
 => => naming to docker.io/library/python-flask-app
                                                           0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities
and learn how to fix them

C:\Users\Tommy G\Desktop\Code\SDemo>docker run -p 5000:5000 python-flask-ap
p
 * Serving Flask app 'sd_test'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deploy
ment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.17.0.2:5000
Press CTRL+C to quit
172.17.0.1 - - [24/Sep/2024 06:49:06] "GET / HTTP/1.1" 200 -
172.17.0.1 - - [24/Sep/2024 06:49:06] "GET /favicon.ico HTTP/1.1" 404 -
```

```
{
    "message": "Hello from Docker!"
}
```

Technical Challenges

Canvas API

- Found and read the documentation
  - Found the API endpoint for submitting grades

- Canvas has implemented GraphQl
  - Login to canvas and /graphiql to the end of the url
  - GUI to create JSON queries

TRACKS CAS

- Implementation of SAML2 (the protocol CAS uses) is possible, and is future-proof
- Getting access to the CAS for development purposes requires the documentation that we completed for this milestone
  - This is the main roadblock we encountered for this technical challenge

We are still in the process of deciding between CAS SAML2 and OAuth2, but the flexibility of OAuth2 is promising

Containerization

- Containerization is required to ensure the security and reliability of the server.
    - Student code needs to be executed in an isolated environment to prevent potential malicious attacks on the system.

- We researched 2 primary tools for comparison
    - Podman
    - Docker

- Tested these environments with a demo, to make sure their implementation could work for S.C.O.R.E.

Documentation

FLORIDA TECH
FLORIDA'S STEM UNIVERSITY

Software Requirements Specification

Functional Requirements

- Immediate Feedback
  - Exactly what a student will see upon auto test completion
- Auto Testing
  - How the professor will be able to configure the auto test environment and what results the professor and student will receive
- Grading Portal
  - Teachers will be able to adjust grades and “sync” with Canvas
- MOSS Integration
  - Submissions will be sent to Stanford’s MOSS server, and the html report will be parsed to be displayed by the application

Functional Requirements (Cont.)

- Assignment Creation
  - Fields: Name, description, number of allowed attempts, due date, and test cases
- Assignment Submission
  - Acceptable file types: python, java, C++, C
  - Unacceptable file types: Byte code, folders, compressed archives
- Assignment Deletion
  - Permanent action from the professor

Interface Requirements

Shell

- Access through code01
- View list of classes
- View list of assignments
  - Filter by class
- Submit assignments and view feedback
- Add, remove, or edit classes
- Add, remove, or edit assignments

Web App

- Online dashboard showing classes and assignment cards
- Students can select an assignment card to access the detail page
  - Students can submit from this page
- Separate dashboard for professors
  - Add, remove, or edit classes
- Grading portal where the professor can see submissions, auto test scores, MOSS report, and assign grades

Software Testing Plan

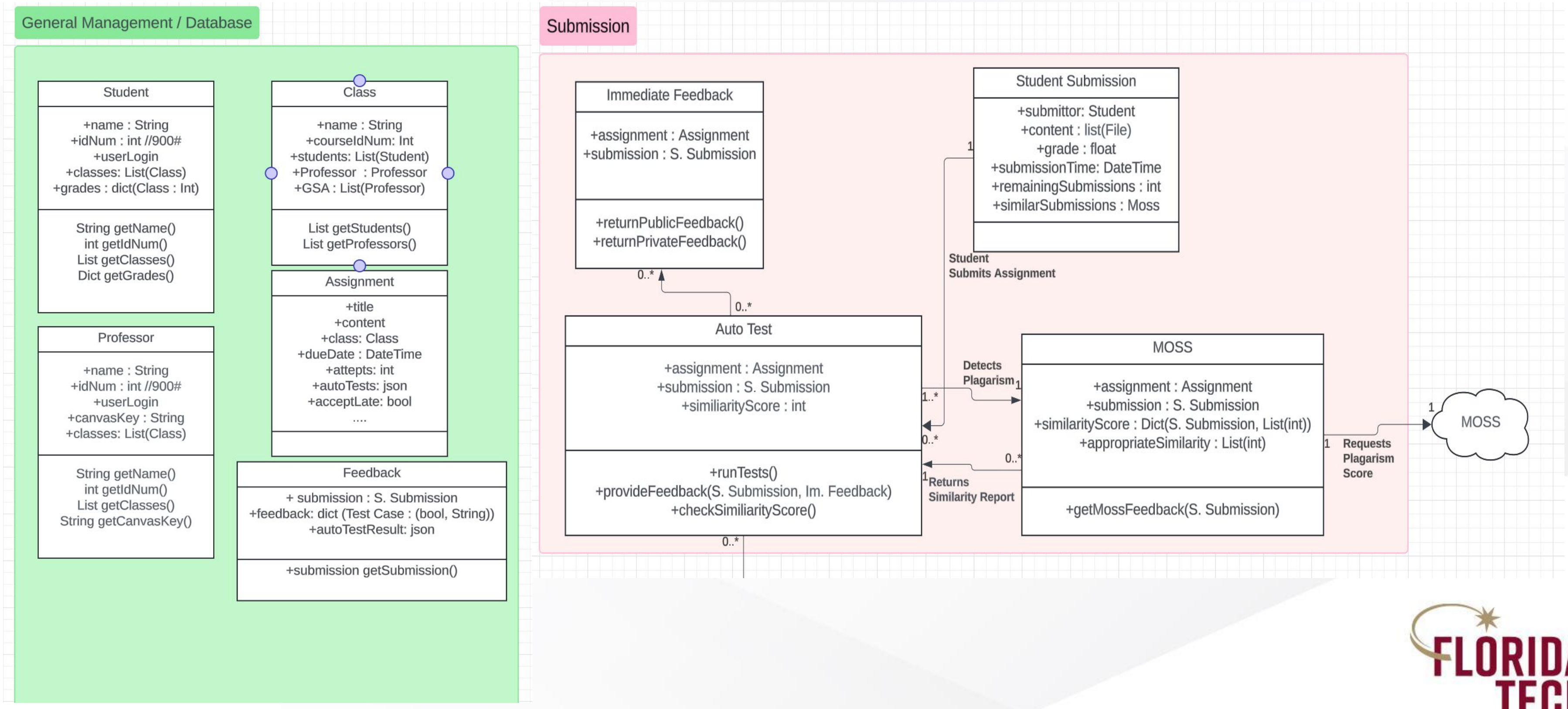
- Our test plan covers all functional requirements
- Each test case detailed a procedure of inputs and the expected outputs
- Covered error cases as well
  - Ensure that unacceptable file types are rejected

- Ensured that we accounted for both users and both interfaces
  - Some features have different interactions depending on whether the user is a student or a professor, or if they are using the shell client or web app

Software Design Document

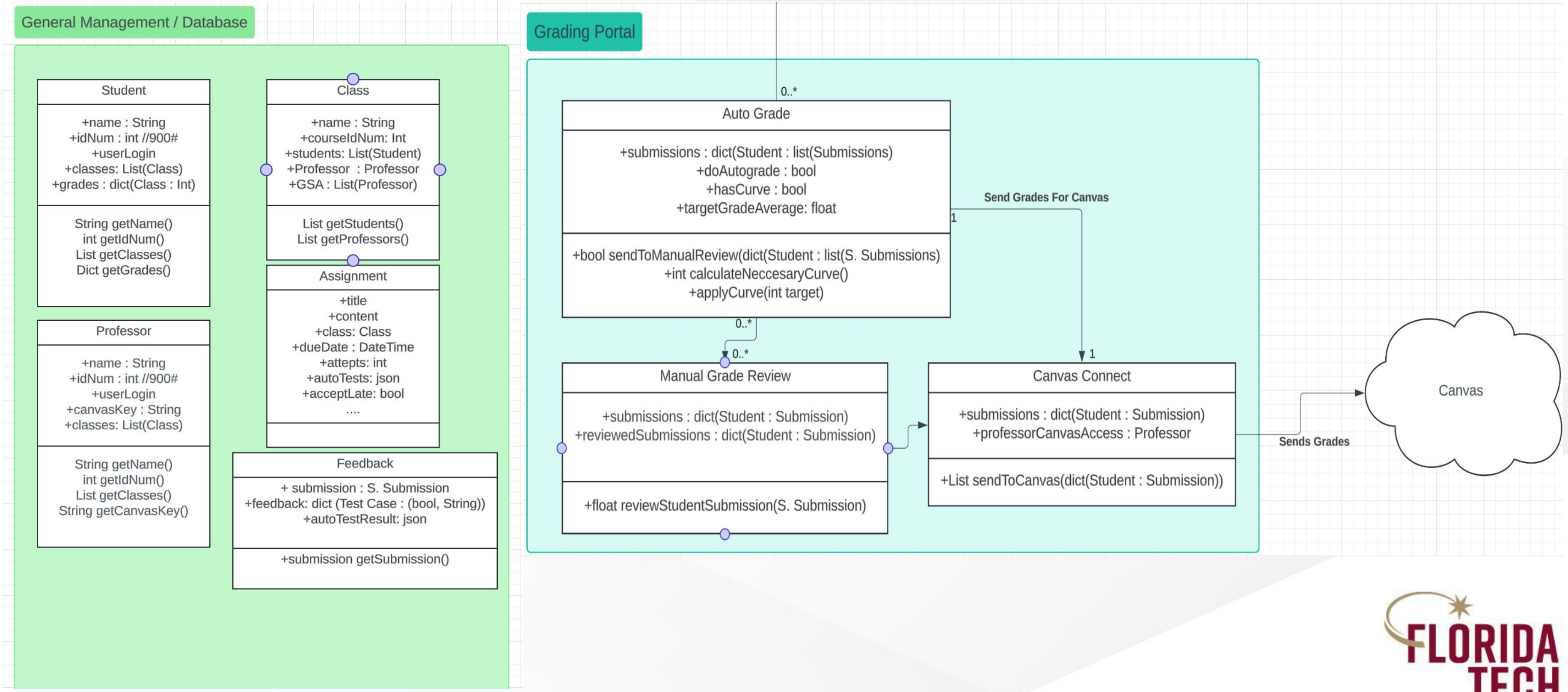
FLORIDA TECH
FLORIDA'S STEM UNIVERSITY

UML Diagram - Submission

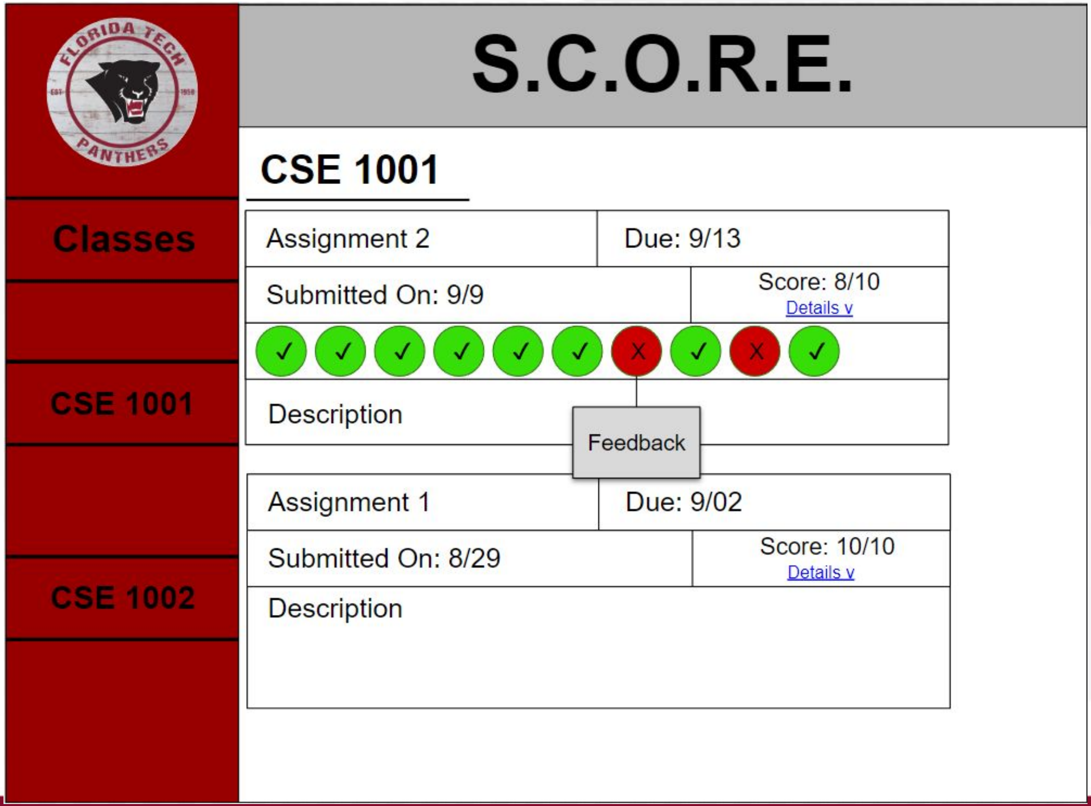


FLORIDA TECH
FLORIDA'S STEM UNIVERSITY

UML Diagram - Grading



Mockup - Student Dashboard



S.C.O.R.E.

Classes

CSE 1001

CSE 1002

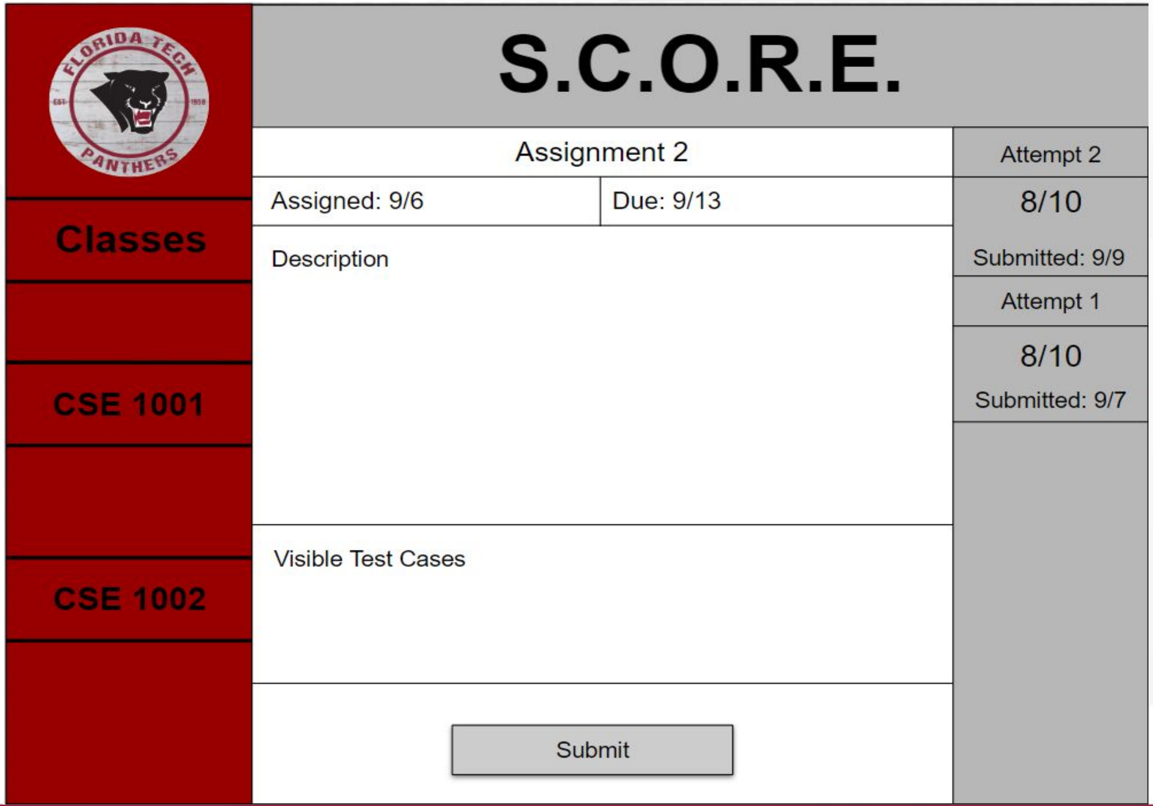
CSE 1001

Assignment 2	Due: 9/13
Submitted On: 9/9	Score: 8/10 Details v
✓ ✓ ✓ ✓ ✓ ✓ X ✓ X ✓	
Description	

Feedback

Assignment 1	Due: 9/02
Submitted On: 8/29	Score: 10/10 Details v
Description	

Mockup - Assignment Detail Page



S.C.O.R.E.

Classes

CSE 1001

CSE 1002

Assignment 2

Assigned: 9/6

Due: 9/13

Description

Visible Test Cases

Submit

Attempt 2

8/10

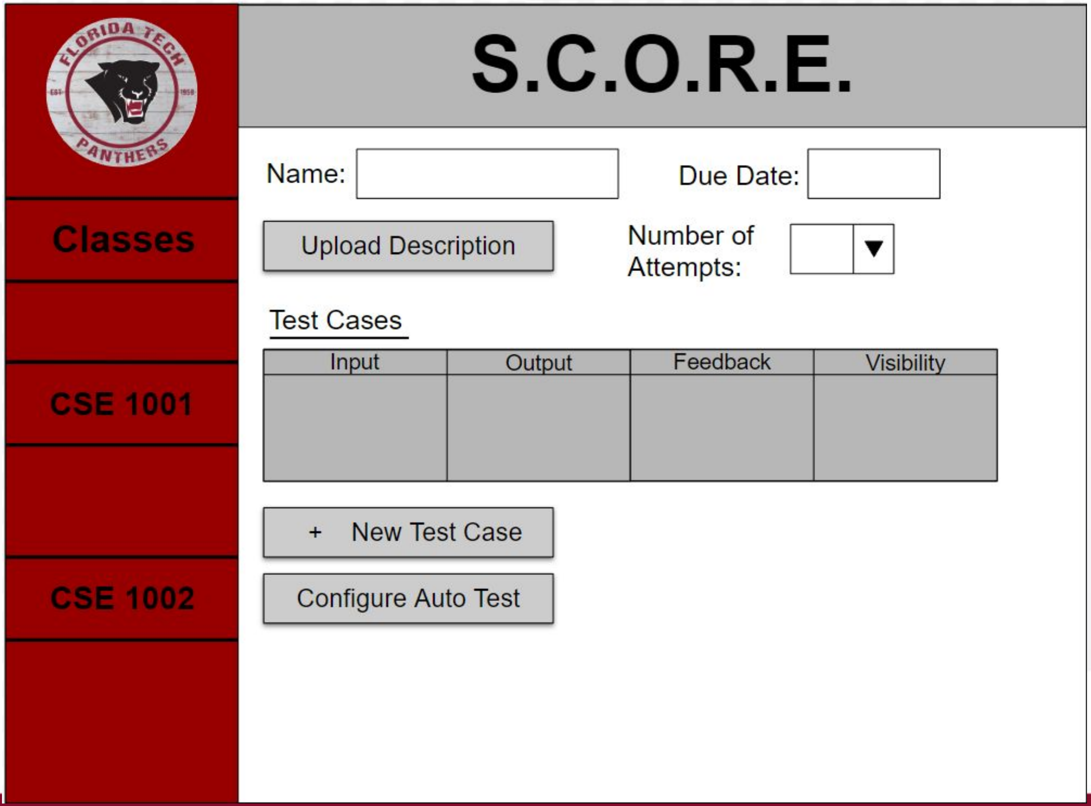
Submitted: 9/9

Attempt 1

8/10

Submitted: 9/7

Mockup - Assignment Creation Page



S.C.O.R.E.

Name:

Due Date:

Upload Description

Number of Attempts: ▼

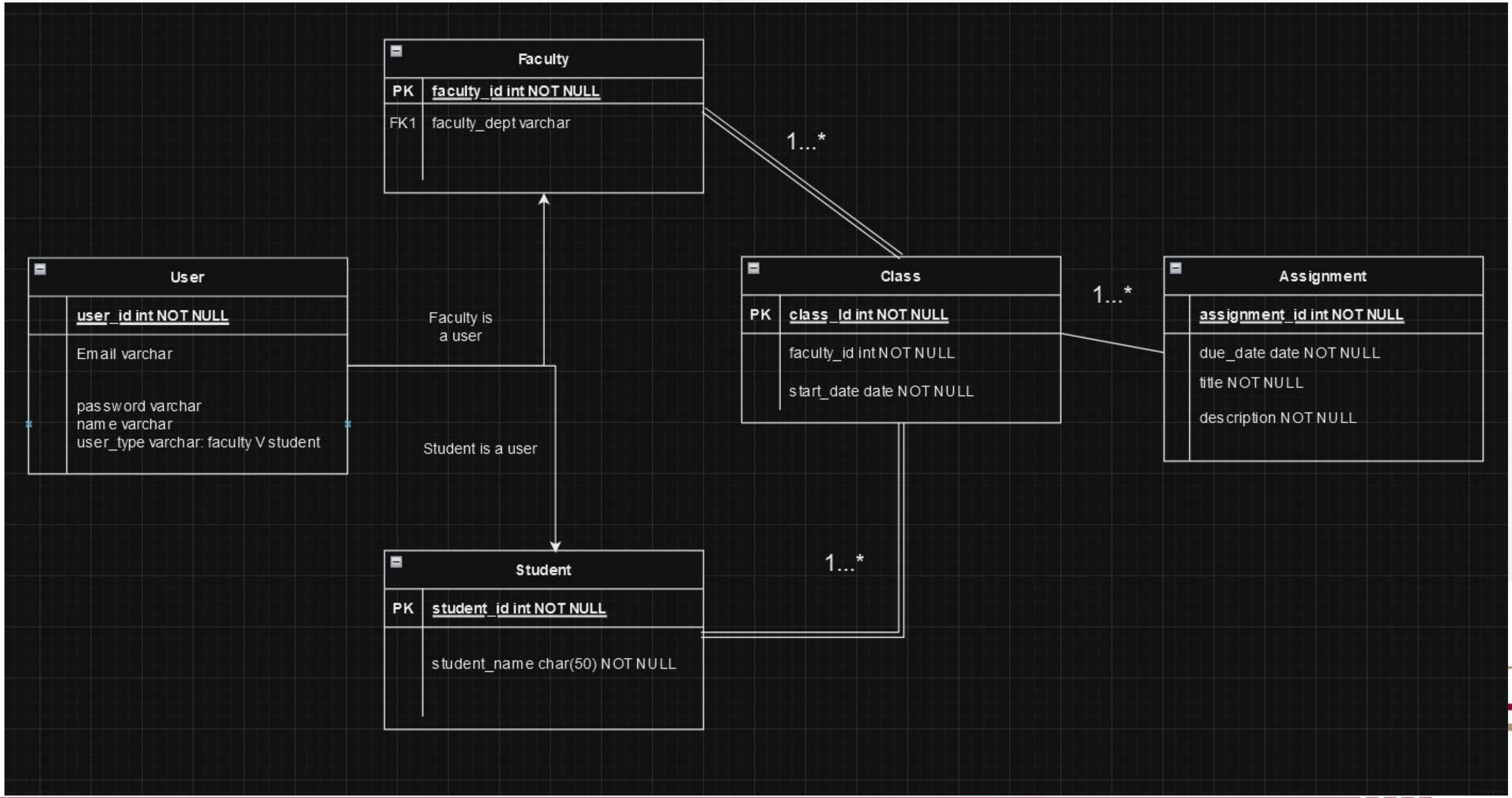
Test Cases

Input	Output	Feedback	Visibility

+ New Test Case

Configure Auto Test

Entity Relationship Diagram



FLORIDA TECH
FLORIDA'S STEM UNIVERSITY

Milestone 2 - Task Matrix

Task	Charlie	Logan	Michael	Tommy
Implement the Shell Application	20%	15%	50%	15%
Implement Assignment Creation	15%	35%	15%	35%
Implement Assignment Submission	40%	20%	20%	20%