

FINAL PROJECT HANDbook

Introduction to OOP

Survival Guide For DJJJJ



March 23, 2025

DJJJ

NSCC Ecampus Programing Year 1

Table of Contents

[Introduction to OOP 3](#_Toc193653644)

[Project Overview 3](#_Toc193653645)

[The Presentation 3](#_Toc193653646)

[What do we need for this project 4](#_Toc193653647)

[Operations for Each User Type 4](#_Toc193653648)

[File Handling for Data Storage 4](#_Toc193653649)

[Exception Handling & Validation 4](#_Toc193653650)

[Use of Java Collections & OOP Principles 4](#_Toc193653651)

[(Bonus) Graphical User Interface (GUI) Implementation 4](#_Toc193653652)

[How the heck do we pass this in? 5](#_Toc193653653)

[The Code 5](#_Toc193653654)

[The UML Diagram and project documentation 5](#_Toc193653655)

[The Proof of Concept 5](#_Toc193653656)

[The Group Policy (Ethos) 6](#_Toc193653657)

[Commitment to participation 6](#_Toc193653658)

[Respect and Communication 6](#_Toc193653659)

[Accountability 6](#_Toc193653660)

[Quality 6](#_Toc193653661)

[Support 6](#_Toc193653662)

[Conflict Resolution 6](#_Toc193653663)

[Transparency 6](#_Toc193653664)

[Work-Life Balance 6](#_Toc193653665)

[Decision Making Process 6](#_Toc193653666)

[Who is taking care of what? 7](#_Toc193653667)

[Dawson 7](#_Toc193653668)

[Jeremy 7](#_Toc193653669)

[Josh 7](#_Toc193653670)

[Judah 7](#_Toc193653671)

[Speculation of Project Development 8](#_Toc193653672)

[Week 1: 8](#_Toc193653673)

[Week 2: 8](#_Toc193653674)

[Week 3: 8](#_Toc193653675)

[Week 4 April 13-16) 8](#_Toc193653676)

# Introduction to OOP

## Project Overview

This is a basic management system to handle students, instructors (faculty) and employees, we are developing a structured application to store and manage RECORDS and using OOP to do it

**We MUST submit a “Proof of Concept 2 DAYS before presentation, aka a working prototype.**

## **The Presentation**

Will be going over the demonstration of the program itself, showing the UML design and technical discussion

# What do we need for this project

Basic User Management

* Store and manage students, instructors, and employees.
* Attributes: name, ID, email, and role (student, instructor, or employee).

## Operations for Each User Type

* Students: Register students and display student list.
* Instructors (Faculty): Store faculty details and display them.
* Employees: Store non-teaching staff records and display them.

## File Handling for Data Storage

* Save student, instructor, and employee data to a file.
* Load saved data when the program starts.

## Exception Handling & Validation

* Prevent duplicate records.
* Validate inputs (e.g., IDs must be unique).

## Use of Java Collections & OOP Principles

* Implement inheritance, encapsulation, and polymorphism.
* Use ArrayLists or HashMaps to manage records

## (Bonus) Graphical User Interface (GUI) Implementation

* Use Java Swing or JavaFX for a basic UI to add and display records.

# How the heck do we pass this in?

## The Code

A fully functional Java program with a menu-driven system for managing students, instructors, and employees.

✔ Organized into appropriate packages.

✔ Uses OOP principles (inheritance, polymorphism, encapsulation).

✔ Implements file handling for storing data.

✔ Includes exception handling for user input errors.

✔ (Bonus) GUI implementation using Java Swing or JavaFX.

We send the source code files

We also send screenshots the program is working

## The UML Diagram and project documentation

A UML class diagram showing system structure.

A short document explaining:

* The system working
* How each OOP concept was used
* Challenges we had to deal with and how did we solve it

We need the UML diagram in either png jpg or pdf

Documentation is either a docx or pdf

## The Proof of Concept

To reiterate this has to be passed in TWO DAYS before the final presentation, the prototype has to demonstrate:

* Adding and displaying student records
* Adding and displaying instructor and employee records
* Basic file handling (saving and loading data)

# The Group Policy (Ethos)

## Commitment to Participation

We will make an effort to contribute equally to our projects by attending meetings and fulfilling our responsibilities in tasks.

## Respect and Communication

We aim to promote open and respectful communication so that every team member feels comfortable sharing their ideas, concerns, and feedback.

## Accountability

Each of us is accountable for completing our assigned tasks in a timely manner. If an unexpected situation arises, it’s essential to inform the team early and collaborate on finding a solution.

## Quality

Our goal is to deliver high-quality work instead of hurrying, as rushing could lead to errors.

## Support

We will actively support each other by sharing our knowledge and offering help when someone is facing difficulties.

# The Group Policy Continued (Ethos)

## Conflict Resolution

In the event of disagreements, team members should tackle conflicts with a focus on solutions and aim for a compromise; if this fails, faculty intervention may be necessary.

## Transparency

We will keep each other informed and updated on our progress regarding the projects.

## Work-Life Balance

We will strive to avoid overworking ourselves and acknowledge the significance of taking breaks.

## Decision-Making Process

# We will conduct group votes before making decisions. The majority will decide; in the case of a tie, the designated tiebreaker will make the final decision on the course of action.

# Who is taking care of what?

## Dawson

* Programming
* Lead Speaking

## Jeremy

* Documentation, power point building/presenter
* Main Speaker, for presentation
* Requesting Bropheus programming

## Josh

* Programming,
* Building Powerpoint,
* Diagram (?)
* documentation(?), t
* roubleshooting, prettying it, try and catch

## Judah

* Wants to program, codesplaining

# Speculation of Project Development

## Week 1:

Day 1: Kickoff, assign tasks, create shared task board/tracker

Day 2-3 Work on OOP

Day 4-5: continue on OOP for two hours, switch to starting client side programming for two hours

Day 6-7: OOP build continues, assign light tasks for Linux project and C project

## Week 2:

Day 8-10: Finish Project 1 Proto Type (April 2nd)

Day 11-14 Full Sprint Client Side Development (Target prototype April 05)

Parllel Coding/set up for C Project

## Week 3:

Days 15-16 Push C project to April 09 deadline

Days 17-18 Work on Linux Project (april 12 deadline)

Day 19-20 (Review ALL PROTOTYPES, Start presentation prep)

## Week 4 April 13-16)

Refine All Projects

Fix Bugs

Finish Documentation

Rehearse Presentations

|  |  |  |
| --- | --- | --- |
| Project | Prototype Deadlines | Buffer/Polish |
| OOP | April 02 | April 07-13 |
| Client Side | April 05 | April 07-13 |
| C Project | April 09 | April 10-13 |
| Linux | April 12 | April 13-16 |

# Project Documentation Notes