SC2002 OBJECT ORIENTED DESIGN & PROGRAMMING

Internship Placement Management System

Declaration of Original Work for SC/CE/CZ2002 Assignment

We hereby declare that the attached group assignment has been researched, undertaken, completed, and submitted as a collective effort by the group members listed below.

We have honoured the principles of academic integrity and have upheld the Student Code of Academic Conduct in the completion of this work.

We understand that if plagiarism is found in the assignment, then lower marks or no marks will be awarded for the assessed work. In addition, disciplinary actions may be taken.

NAME	COURSE	LAB GROUP	SIGNATURE
Chan Tzen Loong, Max (U2422224B)	SC2002	SCE4	
ALHASHIM AHMED SALEH T (U2520942G)	SC2002	SCE4	أحمد الهاشم
Chay Yi Xuan (U2421752G)	SC2002	SCE4	
Roy Koulik (U2421208L)	SC2002	SCE4	Roy Koulik
KAPUR KRISHIV (U2411560C)	SC2002	SCE4	

1. Understanding Requirements & Summary

This console-based Internship Placement Management System supports three roles—**Student**, **Company Representative**, and **Career Centre Staff**—and manages the end-to-end internship lifecycle (posting → application → offer/acceptance → withdrawal). Students browse visible, in-window postings matched to their profile, apply (with caps/eligibility), accept offers, and request withdrawals. Representatives create and manage their own postings but cannot self-approve. Staff govern the process by authorising representatives, approving/rejecting postings, and adjudicating withdrawals. State is persisted to CSV files and exposed via a role-specific, menu-driven CLI. Non-functional priorities include separation of concerns, a repeatable Maven build (runnable JAR), and an architecture that remains easy to extend (e.g., new eligibility checks, reports) without disrupting core flows.

This report explains the design decisions and OO techniques used, including a UML class diagram, application of SOLID, and key design considerations. It also presents test cases with results and a reflection on challenges, lessons learned, and targeted improvements.

2. Design Considerations

The design follows a Boundary–Control–Entity architecture aligned with MVC principles to separate domain state, application workflows, and user interaction. Domain models in com.internshipapp.models are authoritative for data and local invariants. User (abstract) is specialised by Student, CompanyRepresentative, and CareerCenterStaff. Internship models posting attributes and capacity. Application captures the bond between a student and an internship with lifecycle status. WithdrawalRequest provides a durable request envelope with a UUID and timestamp and Report encapsulates generated analytics. Application logic is centralised in manager classes within com.internshipapp.controllers. UserManager loads users, authenticates logins, and authorises representatives. InternshipManager creates postings, manages approval and visibility, cascades deletion cleanly, and generates reports. ApplicationManager governs the application lifecycle, including creation, offer acceptance, and the end-to-end withdrawal workflow. The CLI boundary (com.internshipapp.ui) remains intentionally thin. UserMenu defines the UI contract and is realised by StudentMenu, CompanyRepMenu, and StaffMenu, while App strings everything together, installs Jansi for legible output, loads CSV state, and dispatches to the correct menu post-authentication. Persistence is embedded in the managers for this submission, but the code is structured so that file I/O can be extracted into repositories and injected, enabling storage evolution without disturbing business logic or UI.

3. Assumptions Made

The system assumes a single-user, single-process CLI environment with small CSV datasets suitable for linear reads and writes. Students can hold at most three active applications, only applications in the OFFERED state may be accepted and internship capacity is authoritative on Internship, with slotsFilled incremented atomically on acceptance and decremented on an approved withdrawal that frees a seat. Students only see internships marked isVisible within their openingDate–closingDate window. Representatives manage only the postings they own and cannot self-approve. Staff will approve internships and withdrawal requests and authorise representatives. Identifiers are unique in their CSV domains which includes userID for users, INT%03d for internships, and APP%03d for applications while WithdrawalRequest uses UUID.randomUUID() and records createdAt. Presently, passwords are stored and compared as plaintext to simplify local CLI runs, with a clear path to hashing via an IHasher adapter. Data integrity is enforced by manager-level validation of cross-file references and by centralising cascade deletion in InternshipManager.removeInternship(String).

1. Use of Object-Oriented Concepts

Abstraction, encapsulation, inheritance, and polymorphism are applied in ways that reflect the problem domain and protect invariants.

1) **Abstraction** appears in two locations. First, User is an abstract base class that encapsulates shared identity and credential behaviour while allowing Student, CompanyRepresentative, and CareerCenterStaff to add role-specific state. Second, UserMenu abstracts the CLI contract with a single displayMenu() entry point; concrete menus implement their role-specific interactions while hiding input/output details behind a uniform interface.
2) **Encapsulation** is enforced by keeping entity fields private and mediating state transitions through managers that apply preconditions consistently. For example, ApplicationManager.acceptPlacement(String applicationId) verifies that the target application is in the OFFERED state before transitioning to ACCEPTED, increments Internship.slotsFilled, and marks the posting FILLED when capacity is reached. Similarly, ApplicationManager.processWithdrawalRequest(String requestId, RequestStatus decision) resolves a WithdrawalRequest, updates the underlying Application status, and, when appropriate, releases a seat by adjusting slotsFilled.
3) **Inheritance** models natural 'is-a' relationships: Student, CompanyRepresentative, and CareerCenterStaff extend User, thereby reusing common identity and credential code and enforcing a consistent contract for login and dispatch while keeping role data separate.
4) **Polymorphism** allows the application to reason about roles at the level of their abstractions. After UserManager.login(...) returns a User, App dispatches to the correct menu purely by the concrete subtype without changing the login contract itself; likewise, the CLI can treat all menus as UserMenu, calling displayMenu() uniformly while each menu responds with different behaviour—browsing and applying for students, posting management for representatives, and approvals for staff.

2. Design Principles

The system deliberately aligns with the SOLID principles.

Single Responsibility Principle(SRP)

The project is organised into coherent packages, each with a single, well-defined responsibility. These include Stores, Models, Views, Controllers, Services, Interfaces, and Enums. Within each package, individual classes focus on one clear task aligned with the package's purpose.

1) **Stores**

Purpose: hold app-wide state and handle persistence.

For example, UserManager loads/saves users and handles authentication and password updates (login(...), changePassword(...)), InternshipManager maintains postings and sequential IDs (INT%03d) and controls lifecycle actions (create/approve/toggle/remove), and

ApplicationManager maintains applications/withdrawals with sequential IDs (APP%03d) and persists them to CSV.

2) **Models**

Entity classes and the relationships with each other. For example, User (abstract) is specialised by Student, CompanyRepresentative, and CareerCenterStaff; Internship owns posting data such as owner, capacity, visibility, dates, and status; Application links a student to an internship with lifecycle status; WithdrawalRequest wraps an Application with a durable UUID id and createdAt; and Report encapsulates a timestamped printout.

3) **Views**

Classes that display information to the user on the CLI. For instance, StudentMenu covers browsing, applying, accepting offers, and requesting withdrawals; CompanyRepMenu manages a representative's own postings; and StaffMenu surfaces approvals, representative authorisation, and withdrawal processing. Each menu format outputs, collects input, and calls the appropriate manager methods.

4) **Controllers**

Coordinate use cases and navigation. For example, App performs login via UserManager.login(...), dispatches to the correct UserMenu by role, runs the main loop, and (with your new requirement) forces a re-login after a successful password change. At the subdomain level, the managers act as application controllers by executing atomic, multi-entity transactions—for example, acceptance and withdrawal in ApplicationManager and posting approval or cascade removal in InternshipManager.

i. **Services**

Classes that perform specific business logic functionality. For example, ApplicationManager.createApplication(Student, Internship) enforces caps and initial states; acceptPlacement(String) guards the OFFERED → ACCEPTED transition and updates capacity; processWithdrawalRequest(String, RequestStatus) resolves a request and recovers capacity when appropriate; and InternshipManager.generateReport(InternshipStatus, String, InternshipLevel) composes filters and returns a timestamped Report.

ii. **Interfaces**

Classes that define a set of related methods that must be implemented by a concrete class. Some examples include UserMenu, which defines the CLI boundary contract (displayMenu()) and is implemented by StudentMenu, CompanyRepMenu, and StaffMenu.

iii. Enums

Classes that define a set of constant values that are used throughout the application. For example ApplicationStatus, InternshipStatus, RequestStatus, and InternshipLevel.

Open-Closed Principle

For our project, core manager logic is kept closed but open to extensions via small composable units. InternshipManager.generateReport(InternshipStatus, String, InternshipLevel) already composes filters in a predictable sequence and can be generalized to accept ReportFilter and ReportSorter strategies so new criteria and orders can be added without editing existing logic. Similarly, adding an EligibilityStrategy interface that is invoked by ApplicationManager.createApplication(Student, Internship) would enable new admission policies as separate classes rather than branches inside managers.

Liskov Substitution Principle

For our project, this principle can be seen wherever User is consumed, such as in App and UserManager.login(...), where a User reference can point to a Student, CompanyRepresentative, or CareerCenterStaff without violating preconditions or breaking dispatch.

Interface Segregation Principle

We apply ISP by exposing role-specific interfaces so clients never depend on methods they don't use. For example, the student boundary consumes only student business capabilities (e.g., listing visible internships, applying, accepting, withdrawing), the representative boundary consumes only representative capabilities (e.g., creating/managing own postings), and the staff boundary consumes only staff capabilities (e.g., approvals, authorizations, withdrawal decisions).

Dependency Inversion Principle

This is achieved by structuring managers to depend on interfaces rather than concrete CSV logic. This can be seen when extracting file I/O into StudentRepository, InternshipRepository, and ApplicationRepository (with CSV implementations injected into managers) would complete DIP and allow a storage swap without any changes to application workflows or UI, while also unlocking straightforward unit testing through in-memory fakes.

3. Extensibility & Maintainability

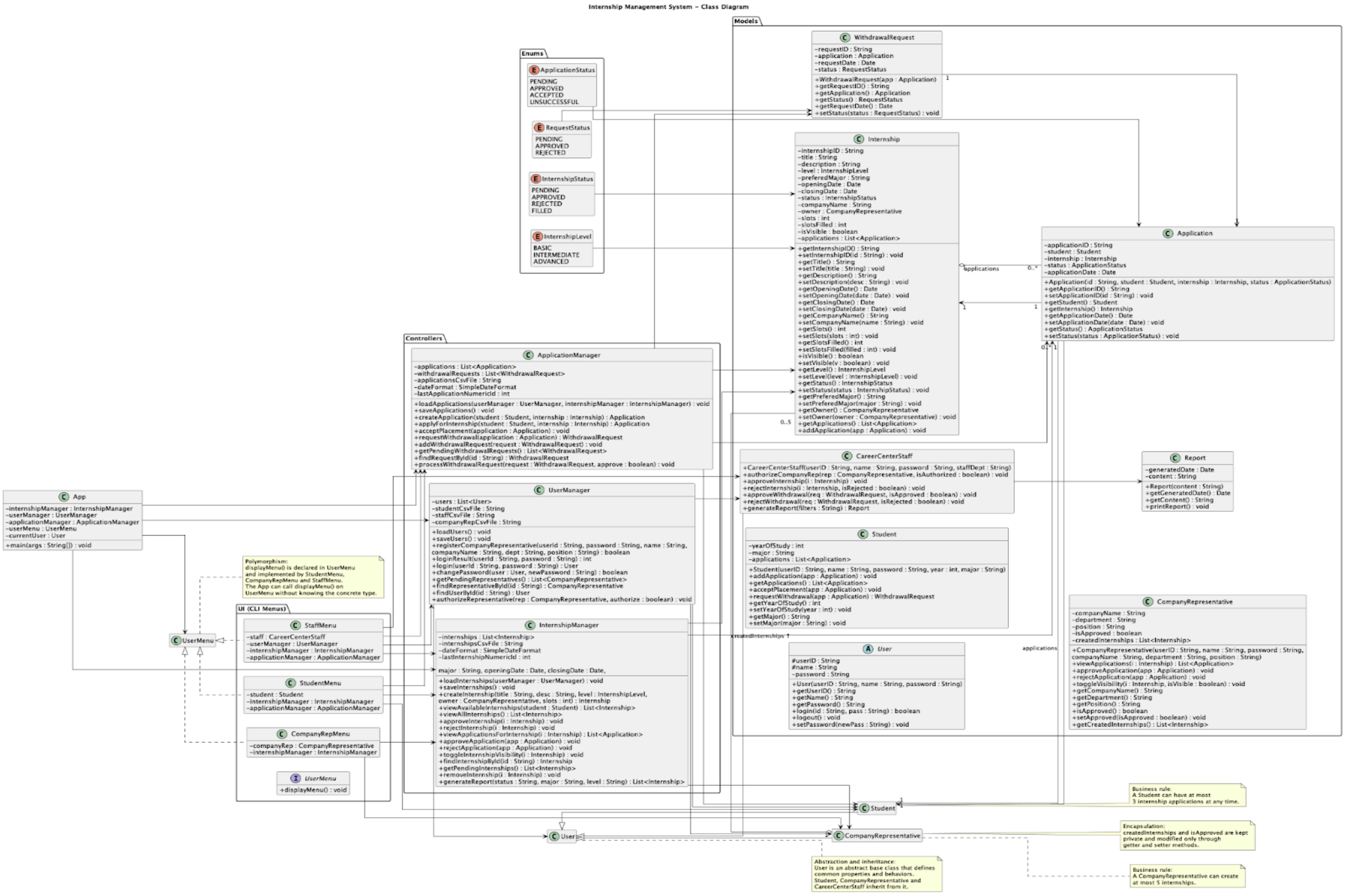
By keeping all business rules in the manager layer and putting each piece of state on the model that owns it, the code gains clean "seams" to extend. For example, acceptance is handled only by ApplicationManager.acceptPlacement(...), withdrawal only by ApplicationManager.processWithdrawalRequest(...), and cascade delete only by InternshipManager.removeInternship(...). Because each flow is funneled through a single orchestration method, you can add features without hunting through the UI or entities. If extra eligibility checks (e.g., GPA or skills) or new reporting dimensions (e.g., sort by remaining capacity) are needed later on, they can be added in as small strategies/helpers that those manager methods call so no invasive rewrites.

Maintainability comes from the Boundary Control Entity split: entities (models) just hold data and invariants, menus only handle input/output, and managers coordinate multi-object work. This keeps classes focused and small. Finally, moving CSV read/write into repository interfaces (and injecting them into managers) shrinks managers further, reduces coupling to storage details, and makes unit tests trivial (mock the repositories, test the rules).

4. **Design Trade-offs**

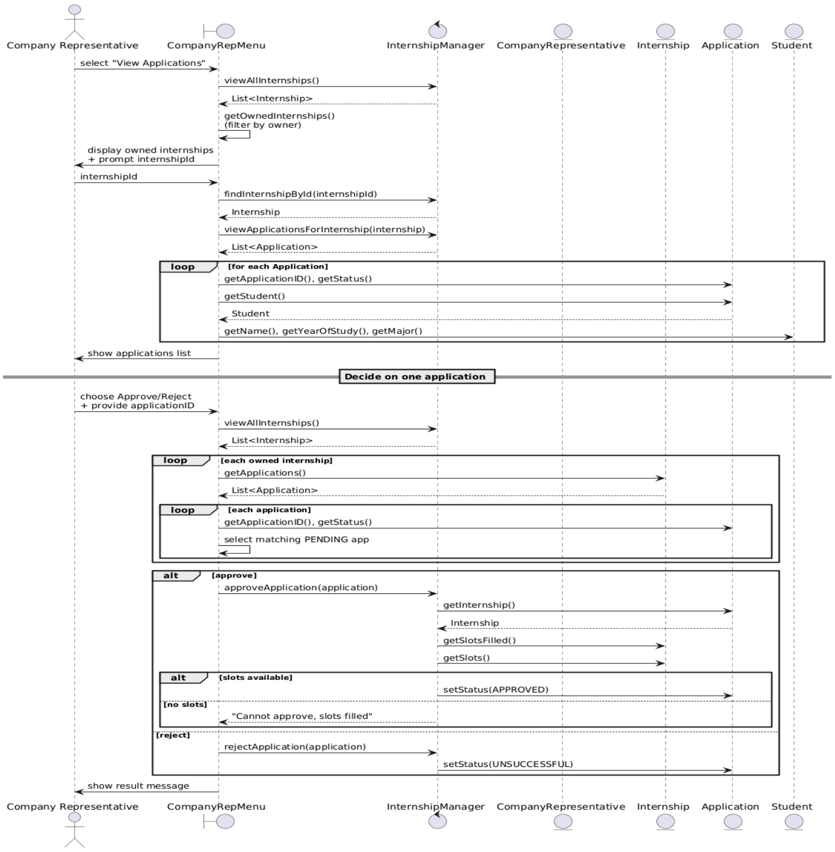
There were several instances of having trade-offs in this project. Firstly, keeping CSV I/O inside the managers made the initial codebase small, but it made individual classes heavier and tightly coupled business logic to storage. Secondly, simple CLI traversal and reporting made it easy to navigate between students, applications, and internships but resulted in greater risk of cyclical or inconsistent updates if writes occur in multiple places.

5. UML Class Diagram

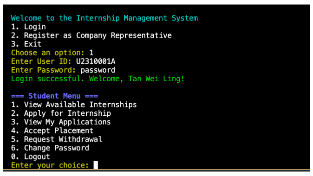
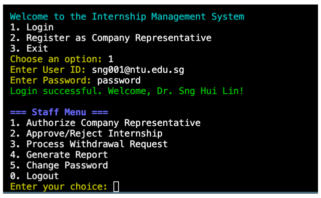
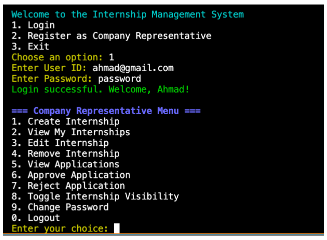
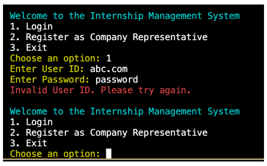
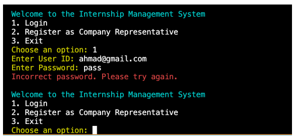


Internship Management System – Class Diagram
Enums
E ApplicationStatus
PENDING
APPROVED
ACCEPTED
UNSUCCESSFUL
E RequestStatus
PENDING
APPROVED
REJECTED
E InternshipStatus
PENDING
APPROVED
REJECTED
FILLED
E InternshipLevel
BASIC
INTERMEDIATE
ADVANCED
Models
C WithdrawalRequest
-requestID : String
-application : Application
-requestDate : Date
-status : RequestStatus
+WithdrawalRequest(app : Application)
+getRequestID() : String
+getApplication() : Application
+getStatus() : RequestStatus
+getRequestDate() : Date
+setStatus(status : RequestStatus) : void
1
C Internship
-internshipID : String
-title : String
-description : String
-level : InternshipLevel
-preferedMajor : String
-openingDate : Date
-closingDate : Date
-status : InternshipStatus
-companyName : String
-owner : CompanyRepresentative
-slots : int
-slotsFilled : int
-isVisible : boolean
-applications : List<Application>
+getInternshipID() : String
+setInternshipID(id : String) : void
+getTitle() : String
+setTitle(title : String) : void
+getDescription() : String
+setDescription(desc : String) : void
+getOpeningDate() : Date
+setOpeningDate(date : Date) : void
+getClosingDate() : Date
+setClosingDate(date : Date) : void
+getCompanyName() : String
+setCompanyName(name : String) : void
+getSlots() : int
+setSlots(slots : int) : void
+getSlotsFilled() : int
+setSlotsFilled(filled : int) : void
+isVisible() : boolean
+setVisible(v : boolean) : void
+getLevel() : InternshipLevel
+setLevel(level : InternshipLevel) : void
+getStatus() : InternshipStatus
+setStatus(status : InternshipStatus) : void
+getPreferedMajor() : String
+setPreferedMajor(major : String) : void
+getOwner() : CompanyRepresentative
+setOwner(owner : CompanyRepresentative) : void
+getApplications() : List<Application>
+addApplication(app : Application) : void
applications
0..*
1
1
0..5
C Application
-applicationID : String
-student : Student
-internship : Internship
-status : ApplicationStatus
-applicationDate : Date
+Application(id : String, student : Student, internship : Internship, status : ApplicationStatus)
+getApplicationID() : String
+setApplicationID(id : String) : void
+getStudent() : Student
+getInternship() : Internship
+getApplicationDate() : Date
+setApplicationDate(date : Date) : void
+getStatus() : ApplicationStatus
+setStatus(status : ApplicationStatus) : void
0..*
1
C CareerCenterStaff
+CareerCenterStaff(userID : String, name : String, password : String, staffDept : String)
+authorizeCompanyRep(rep : CompanyRepresentative, isAuthorized : boolean) : void
+approveInternship(i : Internship) : void
+rejectInternship(i : Internship, isRejected : boolean) : void
+approveWithdrawal(req : WithdrawalRequest, isApproved : boolean) : void
+rejectWithdrawal(req : WithdrawalRequest, isRejected : boolean) : void
+generateReport(filters : String) : Report
C Report
-generatedDate : Date
-content : String
+Report(content : String)
+getGeneratedDate() : Date
+getContent() : String
+printReport() : void
C Student
-yearOfStudy : int
-major : String
-applications : List<Application>
+Student(userID : String, name : String, password : String, year : int, major : String)
+addApplication(app : Application) : void
+getApplications() : List<Application>
+acceptPlacement(app : Application) : void
+requestWithdrawal(app : Application) : WithdrawalRequest
+getYearOfStudy() : int
+setYearOfStudy(year : int) : void
+getMajor() : String
+setMajor(major : String) : void
createdInternships
applications
A User
#userID : String
#name : String
-password : String
+User(userID : String, name : String, password : String)
+getUserID() : String
+getName() : String
+getPassword() : String
+login(id : String, pass : String) : boolean
+logout() : void
+setPassword(newPass : String) : void
C CompanyRepresentative
-companyName : String
-department : String
-position : String
-isApproved : boolean
-createdInternships : List<Internship>
+CompanyRepresentative(userID : String, name : String, password : String, companyName : String, department : String, position : String)
+viewApplications(i : Internship) : List<Application>
+approveApplication(app : Application) : void
+rejectApplication(app : Application) : void
+toggleVisibility(i : Internship, isVisible : boolean) : void
+getCompanyName() : String
+getDepartment() : String
+getPosition() : String
+isApproved() : boolean
+setApproved(isApproved : boolean) : void
+getCreatedInternships() : List<Internship>
Controllers
C ApplicationManager
-applications : List<Application>
-withdrawalRequests : List<WithdrawalRequest>
-applicationsCsvFile : String
-dateFormat : SimpleDateFormat
-lastApplicationNumericId : int
+loadApplications(userManager : UserManager, internshipManager : InternshipManager) : void
+saveApplications() : void
+createApplication(student : Student, internship : Internship) : Application
+applyForInternship(student : Student, internship : Internship) : Application
+acceptPlacement(application : Application) : void
+requestWithdrawal(application : Application) : WithdrawalRequest
+addWithdrawalRequest(request : WithdrawalRequest) : void
+getPendingWithdrawalRequests() : List<WithdrawalRequest>
+findRequestById(id : String) : WithdrawalRequest
+processWithdrawalRequest(request : WithdrawalRequest, approve : boolean) : void
C UserManager
-users : List<User>
-studentCsvFile : String
-staffCsvFile : String
-companyRepCsvFile : String
+loadUsers() : void
+saveUsers() : void
+registerCompanyRepresentative(userId : String, password : String, name : String, companyName : String, dept : String, position : String) : boolean
+loginResult(userId : String, password : String) : int
+login(userId : String, password : String) : User
+changePassword(user : User, newPassword : String) : boolean
+getPendingRepresentatives() : List<CompanyRepresentative>
+findRepresentativeById(id : String) : CompanyRepresentative
+findUserById(id : String) : User
+authorizeRepresentative(rep : CompanyRepresentative, authorize : boolean) : void
C InternshipManager
-internships : List<Internship>
-internshipsCsvFile : String
-dateFormat : SimpleDateFormat
-lastInternshipNumericId : int
major : String, openingDate : Date, closingDate : Date,
+loadInternships(userManager : UserManager) : void
+saveInternships() : void
+createInternship(title : String, desc : String, level : InternshipLevel, owner : CompanyRepresentative, slots : int) : Internship
+viewAvailableInternships(student : Student) : List<Internship>
+viewAllInternships() : List<Internship>
+approveInternship(i : Internship) : void
+rejectInternship(i : Internship) : void
+viewApplicationsForInternship(i : Internship) : List<Application>
+approveApplication(app : Application) : void
+rejectApplication(app : Application) : void
+toggleInternshipVisibility(i : Internship) : void
+findInternshipById(id : String) : Internship
+getPendingInternships() : List<Internship>
+removeInternship(i : Internship) : void
+generateReport(status : String, major : String, level : String) : List<Internship>
C App
-internshipManager : InternshipManager
-userManager : UserManager
-applicationManager : ApplicationManager
-userMenu : UserMenu
-currentUser : User
+main(args : String[]) : void
Polymorphism:
displayMenu() is declared in UserMenu
and implemented by StudentMenu,
CompanyRepMenu and StaffMenu.
The App can call displayMenu() on
UserMenu without knowing the concrete type.
C UserMenu
UI (CLI Menus)
C StaffMenu
-staff : CareerCenterStaff
-userManager : UserManager
-internshipManager : InternshipManager
-applicationManager : ApplicationManager
C StudentMenu
-student : Student
-internshipManager : InternshipManager
-applicationManager : ApplicationManager
C CompanyRepMenu
-companyRep : CompanyRepresentative
-internshipManager : InternshipManager
I UserMenu
+displayMenu() : void
C Student
1
C User
C CompanyRepresentative
Business rule:
A Student can have at most
3 internship applications at any time.
Encapsulation:
createdInternships and isApproved are kept
private and modified only through
getter and setter methods.
Abstraction and inheritance:
User is an abstract base class that defines
common properties and behaviors.
Student, CompanyRepresentative and
CareerCenterStaff inherit from it.
Business rule:
A CompanyRepresentative can create
at most 5 internships.

6. UML Sequence Diagram



Testing & Results

Test Cases	**Input**	**Output**
1 Valid User Login	**Student:** Userid: u2310001a Pass:password **Staff:** userid:sng@001@ntu.edu.sg pass:password **Company representative:** Userid:ahmad@gmail.com pass:password	 Welcome to the Internship Management System 1. Login 2. Register as Company Representative 3. Exit Choose an option: 1 Enter User ID: U2310001A Enter Password: password Login successful. Welcome, Tan Wei Ling! === Student Menu === 1. View Available Internships 2. Apply for Internship 3. View My Applications 4. Accept Placement 5. Request Withdrawal 6. Change Password 0. Logout Enter your choice:  Welcome to the Internship Management System 1. Login 2. Register as Company Representative 3. Exit Choose an option: 1 Enter User ID: sng001@ntu.edu.sg Enter Password: password Login successful. Welcome, Dr. Sng Hui Lin! === Staff Menu === 1. Authorize Company Representative 2. Approve/Reject Internship 3. Process Withdrawal Request 4. Generate Report 5. Change Password 0. Logout Enter your choice:  Welcome to the Internship Management System 1. Login 2. Register as Company Representative 3. Exit Choose an option: 1 Enter User ID: ahmad@gmail.com Enter Password: password Login successful. Welcome, Ahmad! === Company Representative Menu === 1. Create Internship 2. View My Internships 3. Edit Internship 4. Remove Internship 5. View Applications 6. Approve Application 7. Reject Application 8. Toggle Internship Visibility 9. Change Password 0. Logout Enter your choice:
2 Invalid ID	(invalid)userid: abc.com (valid)pass: password	 Welcome to the Internship Management System 1. Login 2. Register as Company Representative 3. Exit Choose an option: 1 Enter User ID: abc.com Enter Password: password Invalid User ID. Please try again. Welcome to the Internship Management System 1. Login 2. Register as Company Representative 3. Exit Choose an option:
3 Incorrect Password	(valid)userid:ahmad@gmail.com (invalid)pass:pass	 Welcome to the Internship Management System 1. Login 2. Register as Company Representative 3. Exit Choose an option: 1 Enter User ID: ahmad@gmail.com Enter Password: pass Incorrect password. Please try again. Welcome to the Internship Management System 1. Login 2. Register as Company Representative 3. Exit Choose an option:

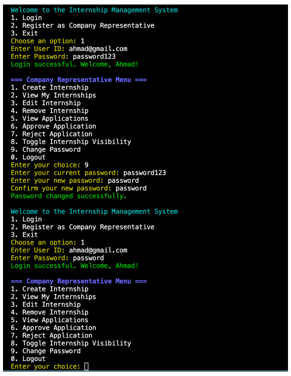
<table>
<tr>
<td>

4
Password Change Functionality

</td>
<td>

Login as company representative
userid:ahmad@gmail.com
Pass:password123
Newpassword:password
Confirm new password:password

</td>
<td>



```
Welcome to the Internship Management System
1. Login
2. Register as Company Representative
3. Exit
Choose an option: 1
Enter User ID: ahmad@gmail.com
Enter Password: password123
Login successful. Welcome, Ahmad!

=== Company Representative Menu ===
1. Create Internship
2. View My Internships
3. Edit Internship
4. Remove Internship
5. View Applications
6. Approve Application
7. Reject Application
8. Toggle Internship Visibility
9. Change Password
0. Logout
Enter your choice: 9
Enter your current password: password123
Enter your new password: password
Confirm your new password: password
Password changed successfully.

Welcome to the Internship Management System
1. Login
2. Register as Company Representative
3. Exit
Choose an option: 1
Enter User ID: ahmad@gmail.com
Enter Password: password
Login successful. Welcome, Ahmad!

=== Company Representative Menu ===
1. Create Internship
2. View My Internships
3. Edit Internship
4. Remove Internship
5. View Applications
6. Approve Application
7. Reject Application
8. Toggle Internship Visibility
9. Change Password
0. Logout
Enter your choice: 
```

</td>
</tr>
<tr>
<td>

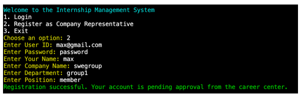
5
Company Representative Account Creation

</td>
<td>

New company representative:
userid:max@gmail.com
pass:password

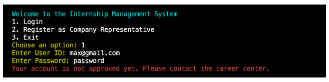
</td>
<td>

Register as company representative



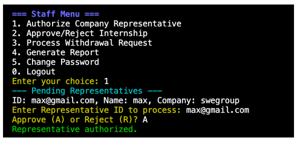
```
Welcome to the Internship Management System
1. Login
2. Register as Company Representative
3. Exit
Choose an option: 2
Enter User ID: max@gmail.com
Enter Password: password
Enter Your Name: max
Enter Company Name: swegroup
Enter Department: group1
Enter Position: member
Registration successful. Your account is pending approval from the career center.
```

Trying to login before approval:



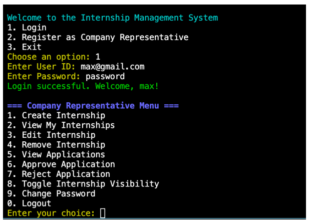
```
Welcome to the Internship Management System
1. Login
2. Register as Company Representative
3. Exit
Choose an option: 1
Enter User ID: max@gmail.com
Enter Password: password
Your account is not approved yet. Please contact the career center.
```

Representative authorized:



```
=== Staff Menu ===
1. Authorize Company Representative
2. Approve/Reject Internship
3. Process Withdrawal Request
4. Generate Report
5. Change Password
0. Logout
Enter your choice: 1
--- Pending Representatives ---
ID: max@gmail.com, Name: max, Company: swegroup
Enter Representative ID to process: max@gmail.com
Approve (A) or Reject (R)? A
Representative authorized.
```

Login as company representative successful:



```
Welcome to the Internship Management System
1. Login
2. Register as Company Representative
3. Exit
Choose an option: 1
Enter User ID: max@gmail.com
Enter Password: password
Login successful. Welcome, max!

=== Company Representative Menu ===
1. Create Internship
2. View My Internships
3. Edit Internship
4. Remove Internship
5. View Applications
6. Approve Application
7. Reject Application
8. Toggle Internship Visibility
9. Change Password
0. Logout
Enter your choice: 
```

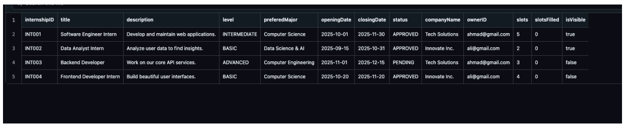
</td>
</tr>
<tr>
<td>

6
Internship Opportunity Visibility Based on User Profile and Toggle

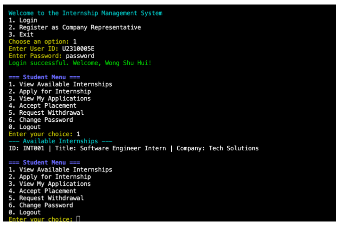
</td>
<td>

Available internships for student wong shu hui should be software engineer intern because its the only intermediate level cs internship

</td>
<td>



1	internshipID	title	description	level	preferedMajor	openingDate	closingDate	status	companyName	ownerID	slots	slotsFilled	isVisible
2	INT001	Software Engineer Intern	Develop and maintain web applications.	INTERMEDIATE	Computer Science	2025-10-01	2025-11-30	APPROVED	Tech Solutions	ahmad@gmail.com	5	0	true
3	INT002	Data Analyst Intern	Analyze user data to find insights.	BASIC	Data Science & AI	2025-09-15	2025-10-31	APPROVED	Innovate Inc.	ali@gmail.com	2	0	true
4	INT003	Backend Developer	Work on our core API services.	ADVANCED	Computer Engineering	2025-11-01	2025-12-15	PENDING	Tech Solutions	ahmad@gmail.com	3	0	false
5	INT004	Frontend Developer Intern	Build beautiful user interfaces.	BASIC	Computer Science	2025-10-20	2025-11-20	APPROVED	Innovate Inc.	ali@gmail.com	4	0	false



```
Welcome to the Internship Management System
1. Login
2. Register as Company Representative
3. Exit
Choose an option: 1
Enter User ID: U2310005E
Enter Password: password
Login successful. Welcome, Wong Shu Hui!

=== Student Menu ===
1. View Available Internships
2. Apply for Internship
3. View My Applications
4. Accept Placement
5. Request Withdrawal
6. Change Password
0. Logout
Enter your choice: 1
--- Available Internships ---
ID: INT001 | Title: Software Engineer Intern | Company: Tech Solutions

=== Student Menu ===
1. View Available Internships
2. Apply for Internship
3. View My Applications
4. Accept Placement
5. Request Withdrawal
6. Change Password
0. Logout
Enter your choice: 
```

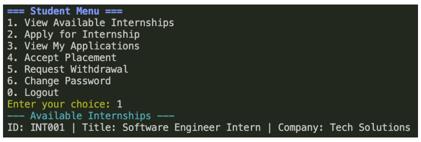
</td>
</tr>
<tr>
<td>

7
Internship Application Eligibility

</td>
<td>

Choose an option: 1
Enter User ID: U2310001A
Enter Password: password
Enter your choice: 1

</td>
<td>



```
=== Student Menu ===
1. View Available Internships
2. Apply for Internship
3. View My Applications
4. Accept Placement
5. Request Withdrawal
6. Change Password
0. Logout
Enter your choice: 1
--- Available Internships ---
ID: INT001 | Title: Software Engineer Intern | Company: Tech Solutions
```

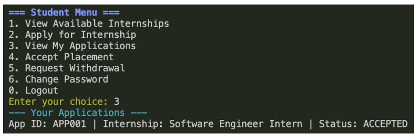
</td>
</tr>
<tr>
<td>

8
Viewing Application Status after Visibility Toggle Off

</td>
<td>

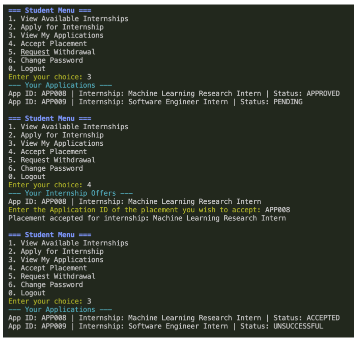
(Before Hiding)
=== Student Menu ===
Enter your choice: 3

</td>
<td>

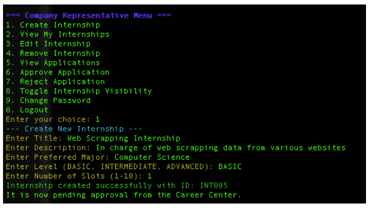
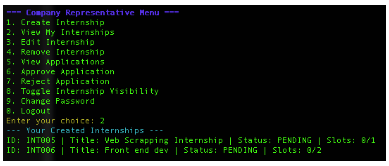
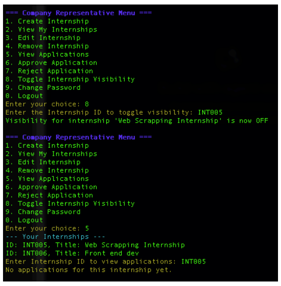


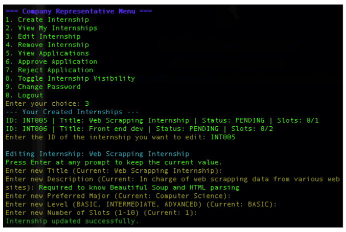
```
=== Student Menu ===
1. View Available Internships
2. Apply for Internship
3. View My Applications
4. Accept Placement
5. Request Withdrawal
6. Change Password
0. Logout
Enter your choice: 3
--- Your Applications ---
App ID: APP001 | Internship: Software Engineer Intern | Status: ACCEPTED
```

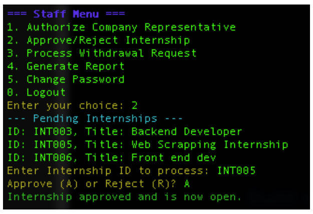
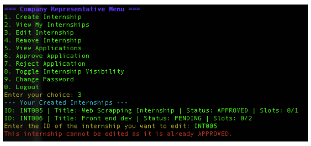
</td>
</tr>
</table>

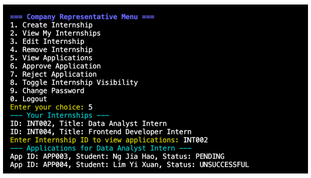
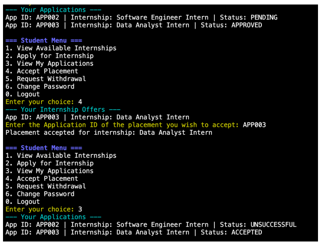
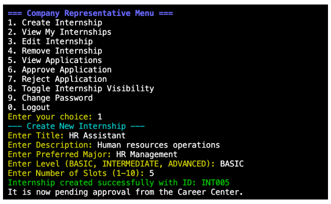
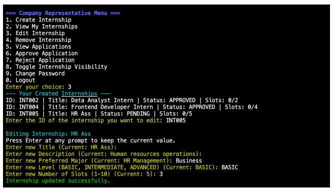
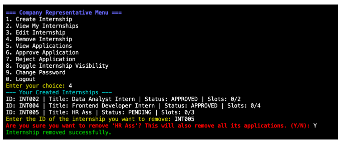
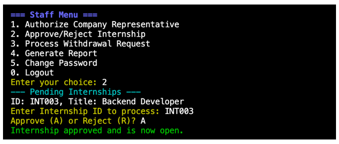
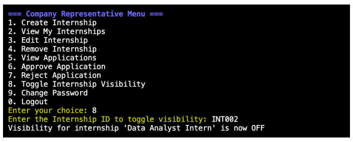
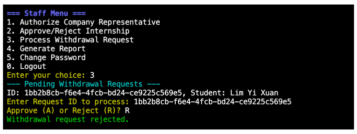
<table>
<tr>
<td><b>10</b>
Single Internship Placement Acceptance per
Student</td>
<td>Before accepting: APP008
APP008(...)Status: APPROVED
APP009(...)Status: PENDING
After accepting :APP008
APP008(...)Status: ACCEPTED
APP009(...)Status: UNSUCCESSFUL
Other applications successfully auto-withdrawn</td>
<td>
=== Student Menu ===
1. View Available Internships
2. Apply for Internship
3. View My Applications
4. Accept Placement
5. Request Withdrawal
6. Change Password
0. Logout
Enter your choice: 3
--- Your Applications ---
App ID: APP008 | Internship: Machine Learning Research Intern | Status: APPROVED
App ID: APP009 | Internship: Software Engineer Intern | Status: PENDING

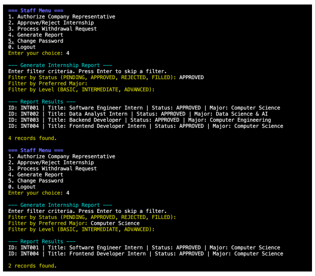
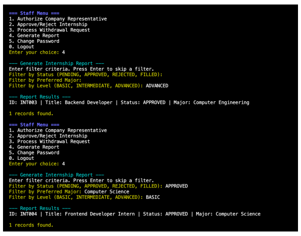
=== Student Menu ===
1. View Available Internships
2. Apply for Internship
3. View My Applications
4. Accept Placement
5. Request Withdrawal
6. Change Password
0. Logout
Enter your choice: 4
--- Your Internship Offers ---
App ID: APP008 | Internship: Machine Learning Research Intern
Enter the Application ID of the placement you wish to accept: APP008
Placement accepted for internship: Machine Learning Research Intern

=== Student Menu ===
1. View Available Internships
2. Apply for Internship
3. View My Applications
4. Accept Placement
5. Request Withdrawal
6. Change Password
0. Logout
Enter your choice: 3
--- Your Applications ---
App ID: APP008 | Internship: Machine Learning Research Intern | Status: ACCEPTED
App ID: APP009 | Internship: Software Engineer Intern | Status: UNSUCCESSFUL</td>
</tr>
<tr>
<td><b>13</b>
Company Representative Internship Opportunity Creation</td>
<td>1) 1
2) Web Scrapping Internship
3) In charge of web scrapping data from various websites
4) Computer Science
5) BASIC
6) 1</td>
<td>
=== Company Representative Menu ===
1. Create Internship
2. View My Internships
3. Edit Internship
4. Remove Internship
5. View Applications
6. Approve Application
7. Reject Application
8. Toggle Internship Visibility
9. Change Password
0. Logout
Enter your choice: 1
--- Create New Internship ---
Enter Title: Web Scrapping Internship
Enter Description: In charge of web scrapping data from various websites
Enter Preferred Major: Computer Science
Enter Level (BASIC, INTERMEDIATE, ADVANCED): BASIC
Enter Number of Slots (1-10): 1
Internship created successfully with ID: INT005
It is now pending approval from the Career Center.</td>
</tr>
<tr>
<td><b>14</b>
Internship Opportunity Approval Status</td>
<td>1) 2</td>
<td>
=== Company Representative Menu ===
1. Create Internship
2. View My Internships
3. Edit Internship
4. Remove Internship
5. View Applications
6. Approve Application
7. Reject Application
8. Toggle Internship Visibility
9. Change Password
0. Logout
Enter your choice: 2
--- Your Created Internships ---
ID: INT005 | Title: Web Scrapping Internship | Status: PENDING | Slots: 0/1
ID: INT006 | Title: Front end dev | Status: PENDING | Slots: 0/2</td>
</tr>
<tr>
<td><b>15</b>
Internship Detail Access
for Company Representative</td>
<td>1) 8
2) 5
3) INT005</td>
<td>
=== Company Representative Menu ===
1. Create Internship
2. View My Internships
3. Edit Internship
4. Remove Internship
5. View Applications
6. Approve Application
7. Reject Application
8. Toggle Internship Visibility
9. Change Password
0. Logout
Enter your choice: 8
Enter the Internship ID to toggle visibility: INT005
Visibility for internship 'Web Scrapping Internship' is now OFF

=== Company Representative Menu ===
1. Create Internship
2. View My Internships
3. Edit Internship
4. Remove Internship
5. View Applications
6. Approve Application
7. Reject Application
8. Toggle Internship Visibility
9. Change Password
0. Logout
Enter your choice: 5
--- Your Internships ---
ID: INT005, Title: Web Scrapping Internship
ID: INT006, Title: Front end dev
Enter Internship ID to view applications: INT005
No applications for this internship yet.</td>
</tr>
<tr>
<td><b>16</b>
Restriction on Editing Approved Opportunities</td>
<td>Internship can be edited when the status is pending
1) 3
2) INT005
3) Required to know Beautiful Soup and HTML parsing
Career Staff approving the internship
1) 2
2) INT005
3) A
Internship can no longer be edited by the Company Rep
1) 4
2) INT005</td>
<td>
=== Company Representative Menu ===
1. Create Internship
2. View My Internships
3. Edit Internship
4. Remove Internship
5. View Applications
6. Approve Application
7. Reject Application
8. Toggle Internship Visibility
9. Change Password
0. Logout
Enter your choice: 3
--- Your Created Internships ---
ID: INT005 | Title: Web Scrapping Internship | Status: PENDING | Slots: 0/1
ID: INT006 | Title: Front end dev | Status: PENDING | Slots: 0/2
Enter the ID of the internship you want to edit: INT005

Editing Internship: Web Scrapping Internship
Press Enter at any prompt to keep the current value.
Enter new Title (Current: Web Scrapping Internship):
Enter new Description (Current: In charge of web scrapping data from various websites): Required to know Beautiful Soup and HTML parsing
Enter new Preferred Major (Current: Computer Science):
Enter new Level (BASIC, INTERMEDIATE, ADVANCED) (Current: BASIC):
Enter new Number of Slots (1-10) (Current: 1):
Internship updated successfully.

=== Staff Menu ===
1. Authorize Company Representative
2. Approve/Reject Internship
3. Process Withdrawal Request
4. Generate Report
5. Change Password
0. Logout
Enter your choice: 2
--- Pending Internships ---
ID: INT003, Title: Backend Developer
ID: INT005, Title: Web Scrapping Internship
ID: INT006, Title: Front end dev
Enter Internship ID to process: INT005
Approve (A) or Reject (R)? A
Internship approved and is now open.

=== Company Representative Menu ===
1. Create Internship
2. View My Internships
3. Edit Internship
4. Remove Internship
5. View Applications
6. Approve Application
7. Reject Application
8. Toggle Internship Visibility
9. Change Password
0. Logout
Enter your choice: 3
--- Your Created Internships ---
ID: INT005 | Title: Web Scrapping Internship | Status: APPROVED | Slots: 0/1
ID: INT006 | Title: Front end dev | Status: PENDING | Slots: 0/2
Enter the ID of the internship you want to edit: INT005
This internship cannot be edited as it is already APPROVED.</td>
</tr>
</table>

18 Student Application Management and Placement Confirmation	5	 === Company Representative Menu === 1. Create Internship 2. View My Internships 3. Edit Internship 4. Remove Internship 5. View Applications 6. Approve Application 7. Reject Application 8. Toggle Internship Visibility 9. Change Password 0. Logout Enter your choice: 5 --- Your Internships --- ID: INT002, Title: Data Analyst Intern ID: INT004, Title: Frontend Developer Intern Enter Internship ID to view applications: INT002 --- Applications for Data Analyst Intern --- App ID: APP003, Student: Ng Jia Hao, Status: PENDING App ID: APP004, Student: Lim Yi Xuan, Status: UNSUCCESSFUL
19 Internship Placement Confirmation Status Update	1. 4 (to accept an offer) 2. 3(APP003 was accepted so APP002 became unsuccessful automatically)	 --- Your Applications --- App ID: APP002 | Internship: Software Engineer Intern | Status: PENDING App ID: APP003 | Internship: Data Analyst Intern | Status: APPROVED === Student Menu === 1. View Available Internships 2. Apply for Internship 3. View My Applications 4. Accept Placement 5. Request Withdrawal 6. Change Password 0. Logout Enter your choice: 4 --- Your Internship Offers --- App ID: APP003 | Internship: Data Analyst Intern Enter the Application ID of the placement you wish to accept: APP003 Placement accepted for internship: Data Analyst Intern === Student Menu === 1. View Available Internships 2. Apply for Internship 3. View My Applications 4. Accept Placement 5. Request Withdrawal 6. Change Password 0. Logout Enter your choice: 3 --- Your Applications --- App ID: APP002 | Internship: Software Engineer Intern | Status: UNSUCCESSFUL App ID: APP003 | Internship: Data Analyst Intern | Status: ACCEPTED
20 Create, Edit, and Delete Internship Opportunity Listings	1.) 1 2.) 3 3.) 4	 === Company Representative Menu === 1. Create Internship 2. View My Internships 3. Edit Internship 4. Remove Internship 5. View Applications 6. Approve Application 7. Reject Application 8. Toggle Internship Visibility 9. Change Password 0. Logout Enter your choice: 1 --- Create New Internship --- Enter Title: HR Assistant Enter Description: Human resources operations Enter Preferred Major: HR Management Enter Level (BASIC, INTERMEDIATE, ADVANCED): BASIC Enter Number of Slots (1-10): 5 Internship created successfully with ID: INT005 It is now pending approval from the Career Center.  === Company Representative Menu === 1. Create Internship 2. View My Internships 3. Edit Internship 4. Remove Internship 5. View Applications 6. Approve Application 7. Reject Application 8. Toggle Internship Visibility 9. Change Password 0. Logout Enter your choice: 3 --- Your Created Internships --- ID: INT002 | Title: Data Analyst Intern | Status: APPROVED | Slots: 0/2 ID: INT004 | Title: Frontend Developer Intern | Status: APPROVED | Slots: 0/4 ID: INT005 | Title: HR Ass | Status: PENDING | Slots: 0/5 Enter the ID of the internship you want to edit: INT005 Editing Internship: HR Ass Press Enter at any prompt to keep the current value. Enter new Title (Current: HR Ass): Enter new Description (Current: Human resources operations): Enter new Preferred Major (Current: HR Management): Business Enter new Level (BASIC, INTERMEDIATE, ADVANCED) (Current: BASIC): BASIC Enter new Number of Slots (1-10) (Current: 5): 3 Internship updated successfully.  === Company Representative Menu === 1. Create Internship 2. View My Internships 3. Edit Internship 4. Remove Internship 5. View Applications 6. Approve Application 7. Reject Application 8. Toggle Internship Visibility 9. Change Password 0. Logout Enter your choice: 4 --- Your Created Internships --- ID: INT002 | Title: Data Analyst Intern | Status: APPROVED | Slots: 0/2 ID: INT004 | Title: Frontend Developer Intern | Status: APPROVED | Slots: 0/4 ID: INT005 | Title: HR Ass | Status: PENDING | Slots: 0/3 Enter the ID of the internship you want to remove: INT005 Are you sure you want to remove 'HR Ass'? This will also remove all its applications. (Y/N): Y Internship removed successfully.
21 Career Centre Staff Internship Opportunity Approval	1. 2 2. A	 === Staff Menu === 1. Authorize Company Representative 2. Approve/Reject Internship 3. Process Withdrawal Request 4. Generate Report 5. Change Password 0. Logout Enter your choice: 2 --- Pending Internships --- ID: INT003, Title: Backend Developer Enter Internship ID to process: INT003 Approve (A) or Reject (R)? A Internship approved and is now open.
22 Toggle Internship Opportunity Visibility	1. 8 2. INT002	 === Company Representative Menu === 1. Create Internship 2. View My Internships 3. Edit Internship 4. Remove Internship 5. View Applications 6. Approve Application 7. Reject Application 8. Toggle Internship Visibility 9. Change Password 0. Logout Enter your choice: 8 Enter the Internship ID to toggle visibility: INT002 Visibility for internship 'Data Analyst Intern' is now OFF
23 Career Centre Staff Internship Opportunity Management	1. 3 2. 1bb2b8cb-f6e4-4fcb-bd24-ce9225c569e5 3. R	 === Staff Menu === 1. Authorize Company Representative 2. Approve/Reject Internship 3. Process Withdrawal Request 4. Generate Report 5. Change Password 0. Logout Enter your choice: 3 --- Pending Withdrawal Requests --- ID: 1bb2b8cb-f6e4-4fcb-bd24-ce9225c569e5, Student: Lim Yi Xuan Enter Request ID to process: 1bb2b8cb-f6e4-4fcb-bd24-ce9225c569e5 Approve (A) or Reject (R)? R Withdrawal request rejected.

24 Generate and Filter Internship Opportunities	1. 4 2. Approved 3. 4 4. Computer Science 5. 4 6. ADVANCED 7. 4 8. Approved Computer Science Basic	 

```
=== Staff Menu ===
1. Authorize Company Representative
2. Approve/Reject Internship
3. Process Withdrawal Request
4. Generate Report
5. Change Password
0. Logout
Enter your choice: 4

--- Generate Internship Report ---
Enter filter criteria. Press Enter to skip a filter.
Filter by Status (PENDING, APPROVED, REJECTED, FILLED): APPROVED
Filter by Preferred Major:
Filter by Level (BASIC, INTERMEDIATE, ADVANCED):

--- Report Results ---
ID: INT001 | Title: Software Engineer Intern | Status: APPROVED | Major: Computer Science
ID: INT002 | Title: Data Analyst Intern | Status: APPROVED | Major: Data Science & AI
ID: INT003 | Title: Backend Developer | Status: APPROVED | Major: Computer Engineering
ID: INT004 | Title: Frontend Developer Intern | Status: APPROVED | Major: Computer Science

4 records found.

=== Staff Menu ===
1. Authorize Company Representative
2. Approve/Reject Internship
3. Process Withdrawal Request
4. Generate Report
5. Change Password
0. Logout
Enter your choice: 4

--- Generate Internship Report ---
Enter filter criteria. Press Enter to skip a filter.
Filter by Status (PENDING, APPROVED, REJECTED, FILLED):
Filter by Preferred Major: Computer Science
Filter by Level (BASIC, INTERMEDIATE, ADVANCED):

--- Report Results ---
ID: INT001 | Title: Software Engineer Intern | Status: APPROVED | Major: Computer Science
ID: INT004 | Title: Frontend Developer Intern | Status: APPROVED | Major: Computer Science

2 records found.
```

```
=== Staff Menu ===
1. Authorize Company Representative
2. Approve/Reject Internship
3. Process Withdrawal Request
4. Generate Report
5. Change Password
0. Logout
Enter your choice: 4

--- Generate Internship Report ---
Enter filter criteria. Press Enter to skip a filter.
Filter by Status (PENDING, APPROVED, REJECTED, FILLED):
Filter by Preferred Major:
Filter by Level (BASIC, INTERMEDIATE, ADVANCED): ADVANCED

--- Report Results ---
ID: INT003 | Title: Backend Developer | Status: APPROVED | Major: Computer Engineering

1 records found.

=== Staff Menu ===
1. Authorize Company Representative
2. Approve/Reject Internship
3. Process Withdrawal Request
4. Generate Report
5. Change Password
0. Logout
Enter your choice: 4

--- Generate Internship Report ---
Enter filter criteria. Press Enter to skip a filter.
Filter by Status (PENDING, APPROVED, REJECTED, FILLED): APPROVED
Filter by Preferred Major: Computer Science
Filter by Level (BASIC, INTERMEDIATE, ADVANCED): BASIC

--- Report Results ---
ID: INT004 | Title: Frontend Developer Intern | Status: APPROVED | Major: Computer Science

1 records found.
```

7. Reflection

As a team, developing the Internship Placement Management System was a comprehensive learning experience that allowed us to apply our full knowledge of object-oriented design and programming principles in a realistic system. All of us have worked collaboratively to transform a textual problem statement into a functioning Java application, including multiple roles, business rules, and data flows through a Command Line Interface. At the beginning, the greatest challenge was translating the system requirements into a coherent Boundary–Control–Entity architecture. Initially, we all found it difficult to decide which responsibilities belonged to control classes and which ones to the entity classes. After implementing the system with BCE discipline, it was clear where each concern belonged: the entity's own state, the manager's own transactions, and the boundaries mediate I/O.

Centralising multi-entity workflows into single methods, such as acceptPlacement() and WithdrawalRequest(), eliminated inconsistencies common in console projects and made the codebase easier to reason about. The experience has also shown that putting persistence behind interfaces is not just an “enterprise” concern; repositories would simplify testing and future-proof the storage layer without altering business logic.

Another challenge was making sure that shared data was consistent across modules. Updating of internship slots and cascading the withdrawals after approvals were to be done automatically by the system. We have learnt the value of state validation, encapsulation, and explicit method contracts in preventing side effects by debugging these dependencies

Through this project, each of us has gained knowledge and increased our understanding of concepts such as polymorphism and inheritance. We reused authentication logic by creating an abstract User superclass with specialised subclasses, making future extensibility straightforward. In our team, each of us has focused on specific modules, synchronising interfaces through peer reviews and Git commits, which has strengthened collaboration and version control practices. Overall, completing this project gave us practical insights into teamwork, communication, and truly helped us learn that problems can be solved working together in a structured manner, reinforcing that good design is iterative and that clean architecture leads to maintainable and robust software.

8. Additional Features Implemented

Colored CLI

The interface utilizes the **JAnsi library** to implement a **Colored CLI**, replacing standard monochrome consoles with a clear visual hierarchy. **Bold Blue** and **Cyan** distinguish navigation headers, **Yellow** highlights input prompts, and **Green/Red** provide immediate operational feedback, allowing users to instantly recognize system status and required actions.

Persistent Database

Data longevity is secured through a **CSV-based Persistent Database** that serializes complex object relationships across sessions. Specialized constructors, particularly within the Internship model, accurately reconstitute historical data (such as original IDs and creation dates) without overwriting records. This layer also preserves critical transaction states, ensuring workflows like ApplicationStatus and withdrawal requests remain intact upon restart.

9. Further Enhancements

Three modest enhancements would yield disproportionately beneficial results. First, extracting repositories and injecting them into managers would complete SRP and DIP, shrink classes, and unlock fast, isolated tests. Second, adding strategy seams for eligibility and report filters/sorters would make policies and analytics open for extension while keeping managers closed to modification. Third, replacing plaintext password comparison with a hashing adapter (IHasher with Sha3Hasher or BCryptHasher) would improve security posture with minimal code churn. If the project later targets a multi-user environment, moving to a small relational store (for example, H2/JPA) behind repository interfaces would address concurrency and integrity at scale without changing the application layer or CLI boundary.

Github Link: SC2002-SCE4