SC2002 OBJECT ORIENTED DESIGN & PROGRAMMING

INTERNSHIP PLACEMENT MANAGEMENT SYSTEM

Report of Project Structure Design & Functionality

AY25/26 Sem 1 | SMAC, Group 6

DATE OF SUBMISSION: 18 NOVEMBER 2025

Github Link:https://github.com/trigo-09/SC2002_OODP

Declaration of Original Work for SC2002/CE2002/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 honored the principles of academic integrity and have upheld Student Code of Academic Conducting 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 (SC2002/CE2002CZ2002)	Lab Group	Signature /Date
Viswanathan Meganathan	SC2002	SMAC	14/11/25
Soeren Kjellerup Skovborg	SC2002	SMAC	14/11/25
Cristian Chye RenHui	SC2002	SMAC	14/11/25
Sean Macinnes	SC2002	SMAC	14/11/25
Lim Cheng Kiat	SC2002	SMAC	Cheng 14/11/25

1. DESIGN CONSIDERATIONS

1.1 OVERVIEW

The Internship Placement Management System (IPMS) is a java-based command-line application designed to streamline Internship management in NTU by being a centralised platform among students,company representatives and career staff. It manages posting of internships, creation of applications among many others. The system enforces role-based access, ensures secure data handling and remains extensible by following Object-Oriented Design Principles.

1.2 DESIGN APPROACH

During the design process we followed key OOD principles by first identifying key entities, properties and interactions from system requirements. From there we modelled the program with class and sequence diagrams to create a strong layered approach with low coupling ensuring high cohesion within each component and modularity in layer.

- Boundary layer
  Interacts with users by presenting menus, displaying results and collecting inputs
- Controller layer
  Each user has their own controller which orchestrates the relevant workflow of the users by coordinating with different services, repository and inputs allowing encapsulation of workflow to be specific to the user.
- Service/Repository layer
  There are 4 different services; Request,Authentication,Application and Internship which contains the necessary business logic and enforced system rules for its own entity, ensuring modularity. The Repository acts as Single-Source of Truth(SSOT) which consolidates data into one centralised, authoritative form which ensures data consistency.
- Entity layer
  Contains core domain classes such as User. These store state and basic behaviours modelling the required Objects

All these layers are tied together by a SystemController which serves as the central coordinator. This approach makes the system more readable, prioritises code reusability and in turn makes the system maintainable. We focused on utilising core principles like abstraction and inheritance to make the system highly flexible: a new user type or methods can be easily implemented without much changes to the core logic, keeping the system scalable.

1.3 HIGHLIGHTS OF SOME DESIGN DECISIONS

User factory: Using the factory pattern, hides the object creation logic from the client and abstracts the object creation process, allowing for flexibility and maintainability.
Internship builder: Using the builder increases readability and avoids long constructors with numerous arguments that could lead to bugs.
Password hashing: Using password hashing with the SHA-256 algorithm improves security by storing credentials in encrypted text rather than plain text.
Exception handling: We created multiple custom exceptions to help output concise error messages to the user when the exceptions are caught in the user interface.
Enum based state control: ApplicationState, InternshipLevel and UserRole are examples of system states where enums were used to minimise errors, improve readability and documentation.
Persistent Data Storage: All modifications to critical data are backed up to system.dat during run time, ensuring data integrity even in unexpected events like system crash.

1.4 APPLIED DESIGN PRINCIPAL (SOLID)

1.4.1 Single Responsibility Principle (SRP)

We used SRP by ensuring that each class has a single clearly defined purpose.
For Example, AuthenticationService has the sole purpose of handling authentication logic and the ApplicationService only includes methods that relate to applications. This makes the system easier to test, debug and extend.

1.4.2 Open/Closed Principle (OCP)

We adhered to this principle by allowing new functionality to be added without altering existing code. This is done by having services depend on interfaces like Irepository instead of a concrete implementation, allowing for new data sources to be added. Similarly we use an abstract user class that new user roles extend, minimising the need to adjust existing code.

1.4.3 Liskov Substitution Principle (LSP)

The LSP is clearly reflected in our user hierarchy where all subclasses of user (Student, Staff, CompanyRep) can be substituted for each other wherever a generic user class is expected which can be seen during our login phase where for authentication User type is used. Another instance where this principle was applied, was in the change of password method. The base controller contains the implementation and any subclass of BaseController(StudentController, StaffController, RepController) will be able to change password without violating parent's contract.

1.4.4 Interface Segregation Principle (ISP)

The interface segregation principle is used to ensure that clients are never forced to implement methods they don't use, this is achieved in our system by having role-specific controllers such as *StudentController* and *StaffController*. This ensures only the relevant methods and action exposed to the clients.

1.4.5 Dependency Injection Principle (DIP)

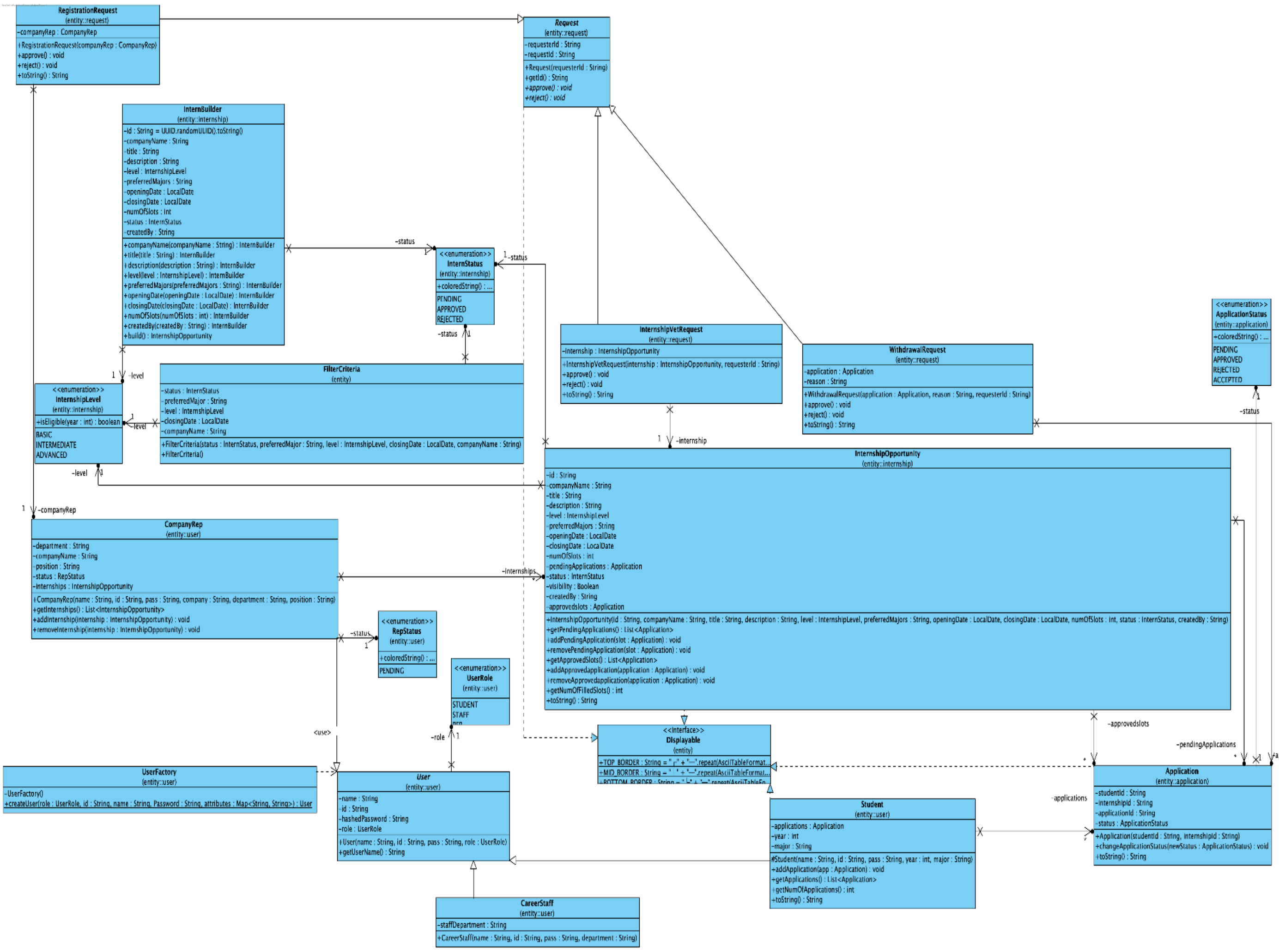
We apply the dependency injection principle by having our services rely on interfaces instead of concrete implementations, and then later injecting the needed dependencies. This is done to make the system more maintainable and extendable. It is implemented with ApplicationService, InternshipService and RequestService that all take IRepository as input.

All things considered, our architecture remained modular, maintainable, and scalable thanks to the application of the SOLID principle. We created a system that can change over time with little chance of disrupting current functionality by separating duties, using abstractions, and guaranteeing extensibility through inheritance and interfaces.

2. DETAILED UML CLASS DIAGRAM

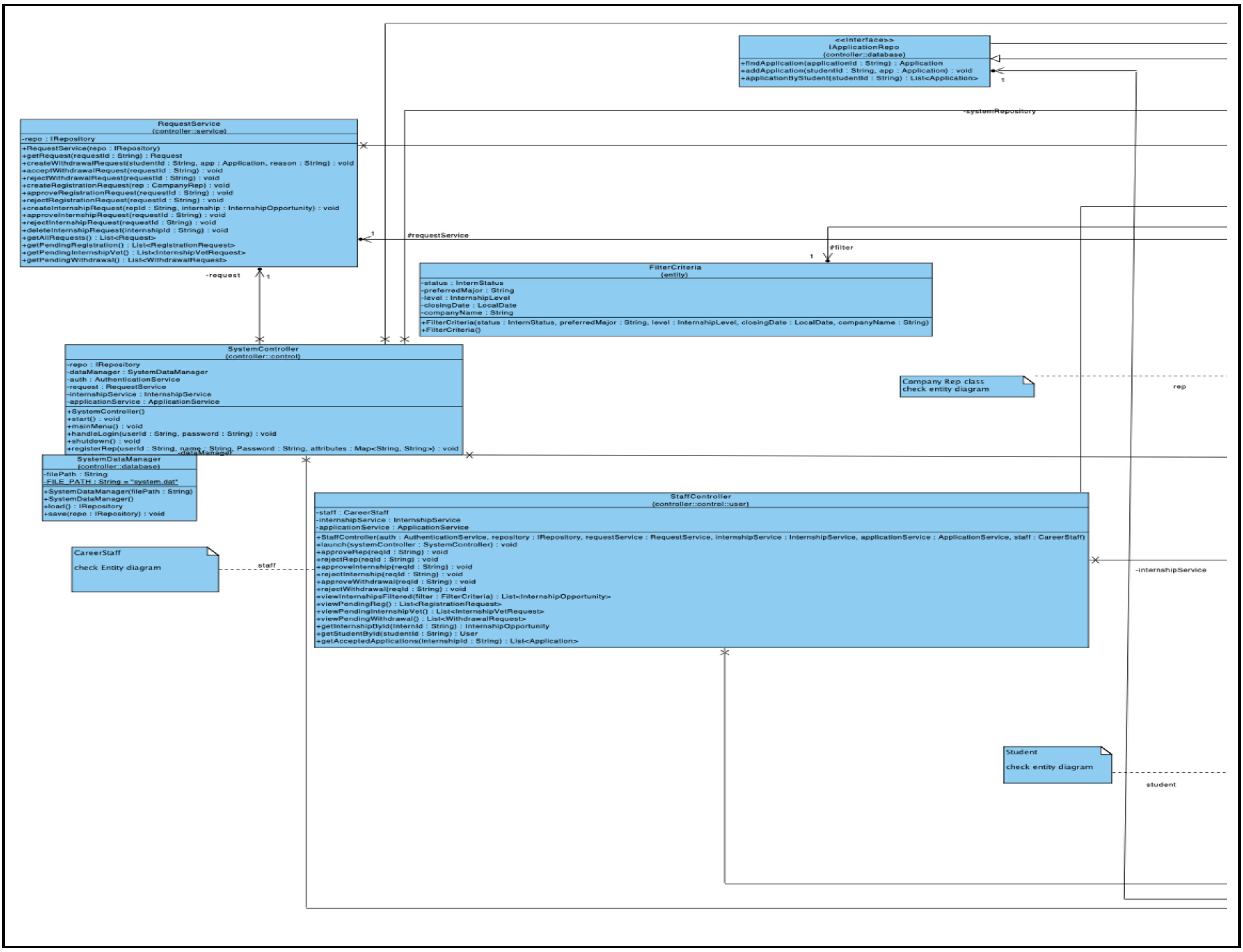
Please Refer<UML diagram> for further details. The diagram had to be artificially split to be placed in the report. You can view the diagram in the folder attached with the report for greater readability.

2.1 ENTITY DIAGRAM

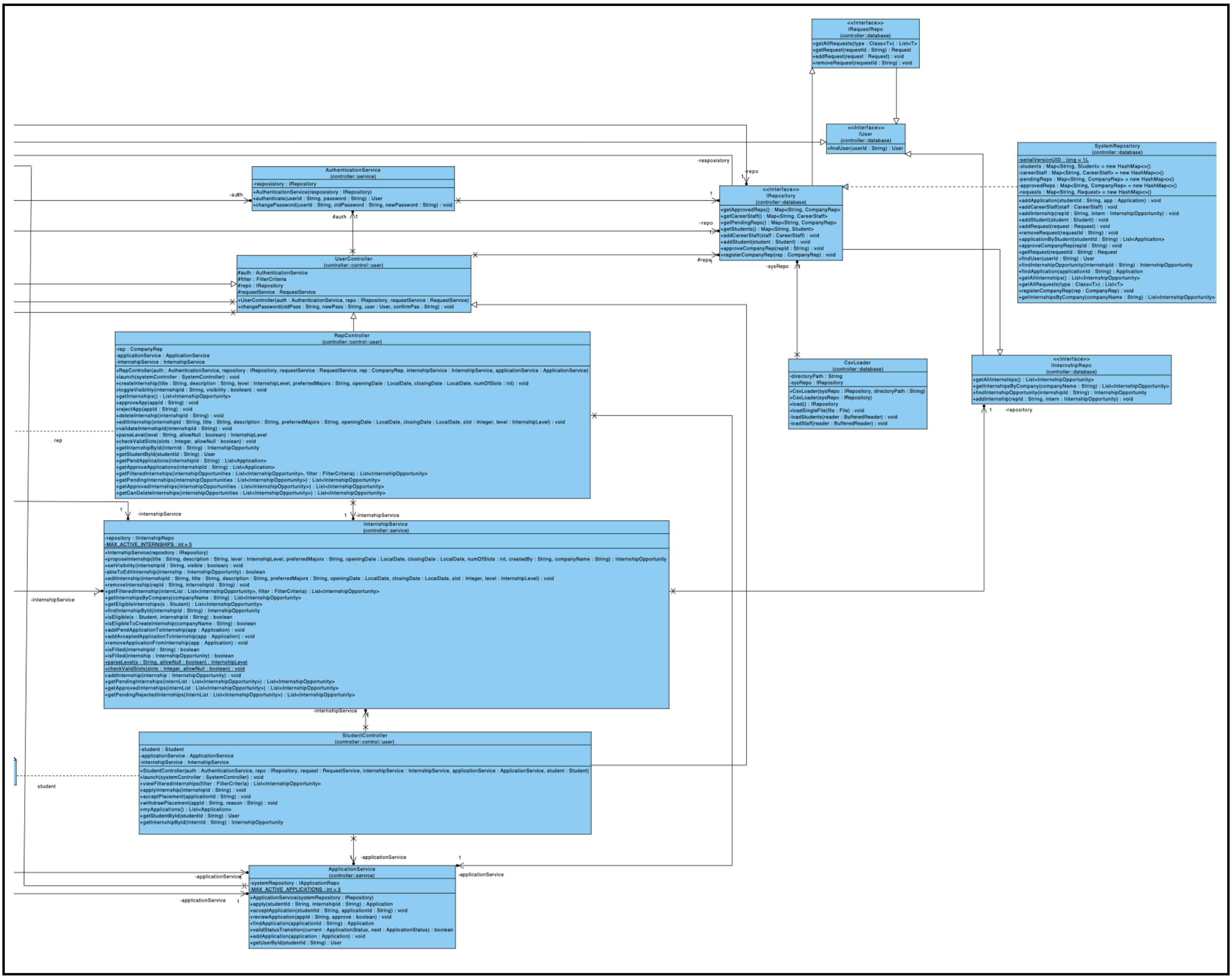

RegistrationRequest
(entity::request)
-companyRep : CompanyRep
+RegistrationRequest(companyRep : CompanyRep)
+approve() : void
+reject() : void
+toString() : String
Request
(entity::request)
-requesterId : String
-requestId : String
+Request(requesterId : String)
+getId() : String
+approve() : void
+reject() : void
InternBuilder
(entity::internship)
-id : String = UUID.randomUUID().toString()
-companyName : String
-title : String
-description : String
-level : InternshipLevel
-preferredMajors : String
-openingDate : LocalDate
-closingDate : LocalDate
-numOfSlots : Int
-status : InternStatus
-createdBy : String
+companyName(companyName : String) : InternBuilder
+title(title : String) : InternBuilder
+description(description : String) : InternBuilder
+level(level : InternshipLevel) : InternBuilder
+preferredMajors(preferredMajors : String) : InternBuilder
+openingDate(openingDate : LocalDate) : InternBuilder
+closingDate(closingDate : LocalDate) : InternBuilder
+numOfSlots(numOfSlots : int) : InternBuilder
+createdBy(createdBy : String) : InternBuilder
+build() : InternshipOpportunity
-status
1
<<enumeration>>
InternStatus
(entity::internship)
+coloredString() : ...
PENDING
APPROVED
REJECTED
1 -status
-status 1
InternshipVetRequest
(entity::request)
-internship : InternshipOpportunity
+InternshipVetRequest(internship : InternshipOpportunity, requesterId : String)
+approve() : void
+reject() : void
+toString() : String
WithdrawalRequest
(entity::request)
-application : Application
-reason : String
+WithdrawalRequest(application : Application, reason : String, requesterId : String)
+approve() : void
+reject() : void
+toString() : String
<<enumeration>>
ApplicationStatus
(entity::application)
+coloredString() : ...
PENDING
APPROVED
REJECTED
ACCEPTED
1
-status
1 -level
<<enumeration>>
InternshipLevel
(entity::internship)
+isEligible(year : int) : boolean
BASIC
INTERMEDIATE
ADVANCED
1
-level
FilterCriteria
(entity)
-status : InternStatus
-preferredMajor : String
-level : InternshipLevel
-closingDate : LocalDate
-companyName : String
+FilterCriteria(status : InternStatus, preferredMajor : String, level : InternshipLevel, closingDate : LocalDate, companyName : String)
+FilterCriteria()
-level 1
1 -internship
InternshipOpportunity
(entity::internship)
-id : String
-companyName : String
-title : String
-description : String
-level : InternshipLevel
-preferredMajors : String
-openingDate : LocalDate
-closingDate : LocalDate
-numOfSlots : int
-pendingApplications : Application
-status : InternStatus
-visibility : Boolean
-createdBy : String
-approvedslots : Application
+InternshipOpportunity(id : String, companyName : String, title : String, description : String, level : InternshipLevel, preferredMajors : String, openingDate : LocalDate, closingDate : LocalDate, numOfSlots : int, status : InternStatus, createdBy : String)
+getPendingApplications() : List<Application>
+addPendingApplication(slot : Application) : void
+removePendingApplication(slot : Application) : void
+getApprovedSlots() : List<Application>
+addApprovedapplication(application : Application) : void
+removeApprovedapplication(application : Application) : void
+getNumOfFilledSlots() : int
+toString() : String
1 -companyRep
CompanyRep
(entity::user)
-department : String
-companyName : String
-position : String
-status : RepStatus
-internships : InternshipOpportunity
+CompanyRep(name : String, id : String, pass : String, company : String, department : String, position : String)
+getInternships() : List<InternshipOpportunity>
+addInternship(internship : InternshipOpportunity) : void
+removeInternship(internship : InternshipOpportunity) : void
-internships
*
-status
1
<<enumeration>>
RepStatus
(entity::user)
+coloredString() : ...
PENDING
<<enumeration>>
UserRole
(entity::user)
STUDENT
STAFF
REP
<use>
-role 1
<<Interface>>
Displayable
(entity)
+TOP_BORDER : String = "┌" + "─".repeat(AsciiTableFormat...
+MID_BORDER : String = "├" + "─".repeat(AsciiTableFormat...
+BOTTOM_BORDER : String = "└" + "─".repeat(AsciiTableFo...
-approvedslots
-pendingApplications
*
*
1
-application
UserFactory
(entity::user)
-UserFactory()
+createUser(role : UserRole, id : String, name : String, Password : String, attributes : Map<String, String>) : User
User
(entity::user)
-name : String
-id : String
-hashedPassword : String
-role : UserRole
+User(name : String, id : String, pass : String, role : UserRole)
+getUserName() : String
-applications
*
Student
(entity::user)
-applications : Application
-year : int
-major : String
#Student(name : String, id : String, pass : String, year : int, major : String)
+addApplication(app : Application) : void
+getApplications() : List<Application>
+getNumOfApplications() : int
+toString() : String
Application
(entity::application)
-studentId : String
-internshipId : String
-applicationId : String
-status : ApplicationStatus
+Application(studentId : String, internshipId : String)
+changeApplicationStatus(newStatus : ApplicationStatus) : void
+toString() : String
CareerStaff
(entity::user)
-staffDepartment : String
+CareerStaff(name : String, id : String, pass : String, department : String)

(Diagram rearranged to fit in report)

2.2 CONTROL DIAGRAM

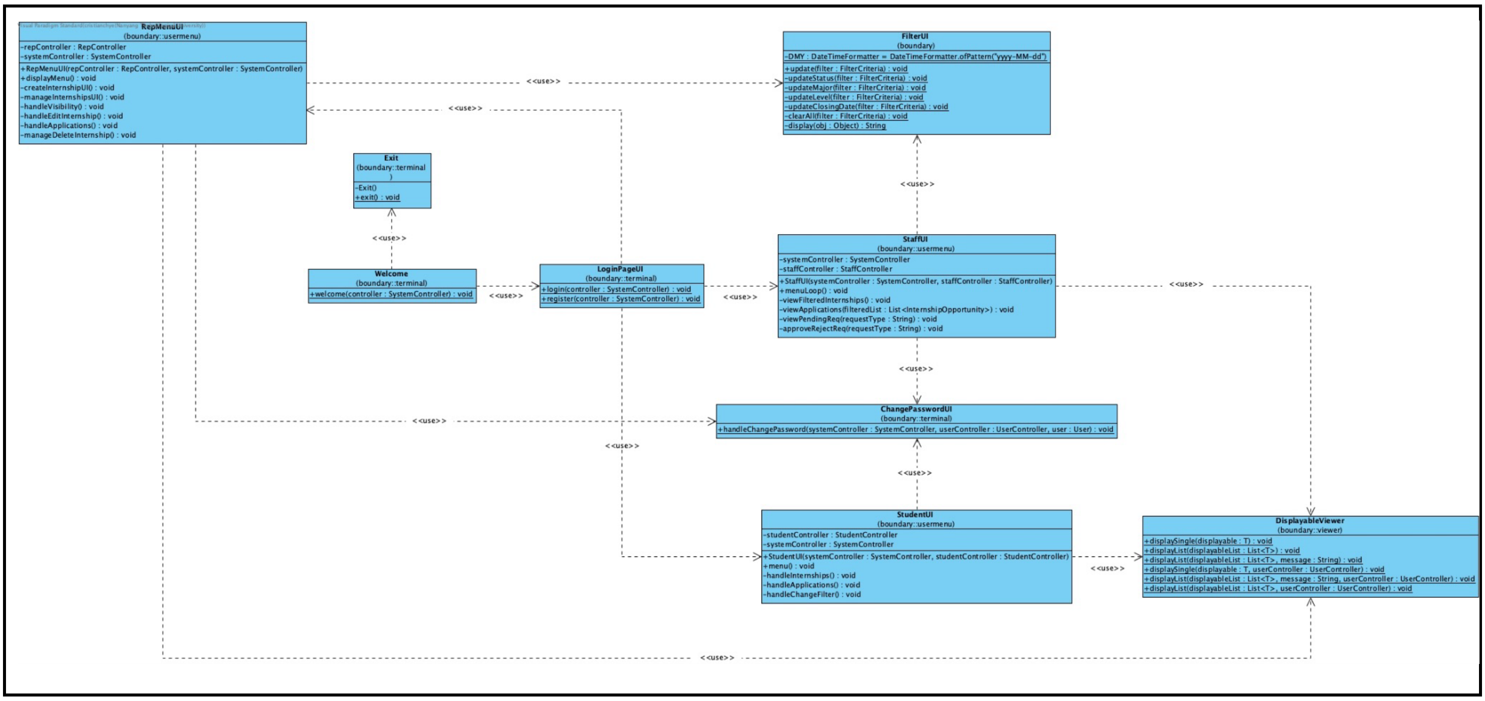

<<Interface>>
IApplicationRepo
(controller::database)
+findApplication(applicationId : String) : Application
+addApplication(studentId : String, app : Application) : void
+applicationByStudent(studentId : String) : List<Application>
1
-systemRepository
RequestService
(controller::service)
-repo : IRepository
+RequestService(repo : IRepository)
+getRequest(requestId : String) : Request
+createWithdrawalRequest(studentId : String, app : Application, reason : String) : void
+acceptWithdrawalRequest(requestId : String) : void
+rejectWithdrawalRequest(requestId : String) : void
+createRegistrationRequest(rep : CompanyRep) : void
+approveRegistrationRequest(requestId : String) : void
+rejectRegistrationRequest(requestId : String) : void
+createInternshipRequest(repId : String, internship : InternshipOpportunity) : void
+approveInternshipRequest(requestId : String) : void
+rejectInternshipRequest(requestId : String) : void
+deleteInternshipRequest(internshipId : String) : void
+getAllRequests() : List<Request>
+getPendingRegistration() : List<RegistrationRequest>
+getPendingInternshipVet() : List<InternshipVetRequest>
+getPendingWithdrawal() : List<WithdrawalRequest>
1
#requestService
#filter
1
-request
1
FilterCriteria
(entity)
-status : InternStatus
-preferredMajor : String
-level : InternshipLevel
-closingDate : LocalDate
-companyName : String
+FilterCriteria(status : InternStatus, preferredMajor : String, level : InternshipLevel, closingDate : LocalDate, companyName : String)
+FilterCriteria()
SystemController
(controller::control)
-repo : IRepository
-dataManager : SystemDataManager
-auth : AuthenticationService
-request : RequestService
-internshipService : InternshipService
-applicationService : ApplicationService
+SystemController()
+start() : void
+mainMenu() : void
+handleLogin(userId : String, password : String) : void
+shutdown() : void
+registerRep(userId : String, name : String, Password : String, attributes : Map<String, String>) : void
-dataManager
Company Rep class
check entity diagram
rep
SystemDataManager
(controller::database)
-filePath : String
-FILE_PATH : String = "system.dat"
+SystemDataManager(filePath : String)
+SystemDataManager()
+load() : IRepository
+save(repo : IRepository) : void
StaffController
(controller::control::user)
-staff : CareerStaff
-internshipService : InternshipService
-applicationService : ApplicationService
+StaffController(auth : AuthenticationService, repository : IRepository, requestService : RequestService, internshipService : InternshipService, applicationService : ApplicationService, staff : CareerStaff)
+launch(systemController : SystemController) : void
+approveRep(reqId : String) : void
+rejectRep(reqId : String) : void
+approveInternship(reqId : String) : void
+rejectInternship(reqId : String) : void
+approveWithdrawal(reqId : String) : void
+rejectWithdrawal(reqId : String) : void
+viewInternshipsFiltered(filter : FilterCriteria) : List<InternshipOpportunity>
+viewPendingReg() : List<RegistrationRequest>
+viewPendingInternshipVet() : List<InternshipVetRequest>
+viewPendingWithdrawal() : List<WithdrawalRequest>
+getInternshipById(InternId : String) : InternshipOpportunity
+getStudentById(studentId : String) : User
+getAcceptedApplications(internshipId : String) : List<Application>
CareerStaff
check Entity diagram
staff
-internshipService
Student
check entity diagram
student

(1st half of the Control UML)

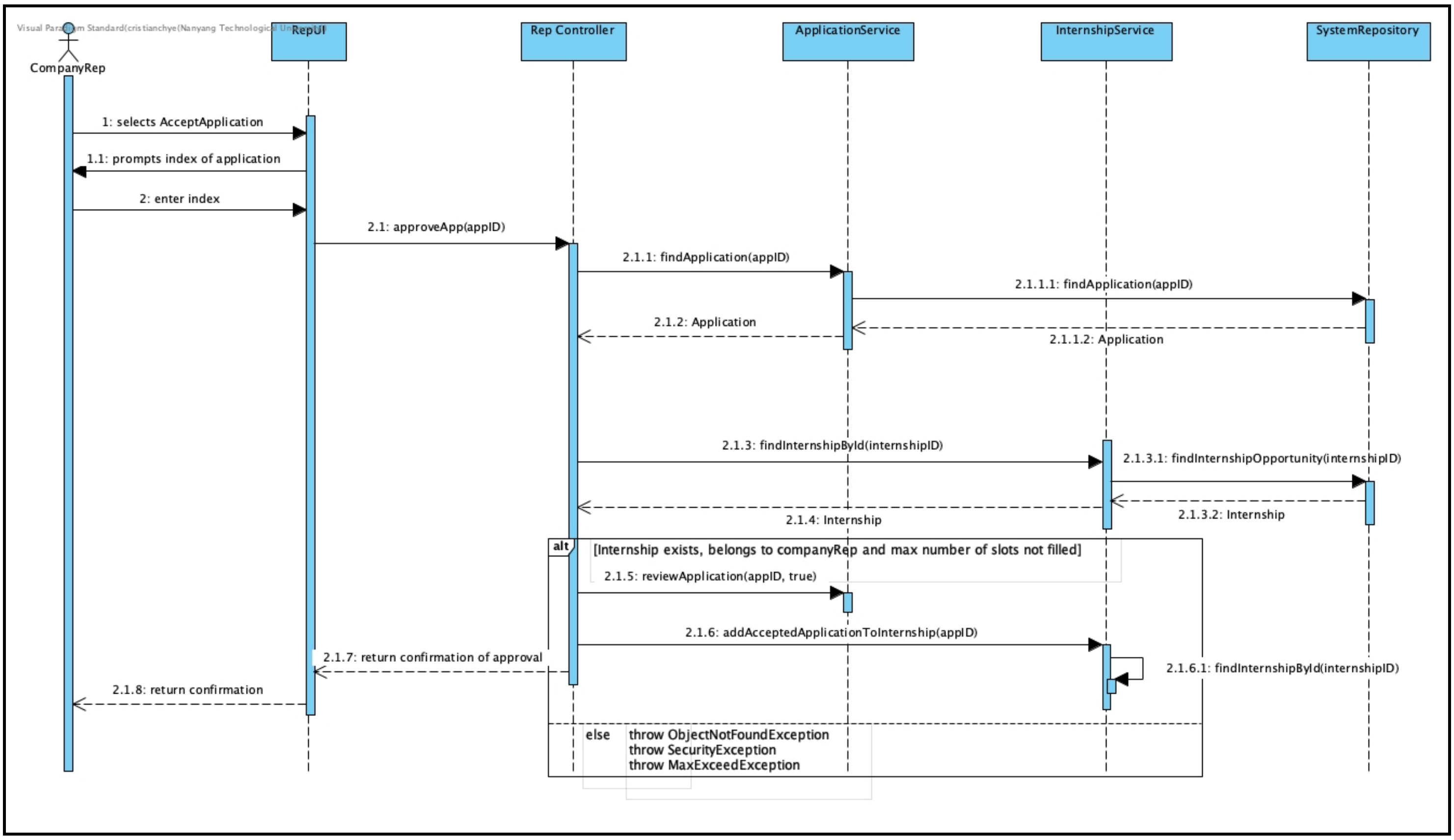

<<Interface>>
IRequestRepo
(controller::database)
+getAllRequests(type : Class<T>) : List<T>
+getRequest(requestId : String) : Request
+addRequest(request : Request) : void
+removeRequest(requestId : String) : void
<<Interface>>
IUser
(controller::database)
+findUser(userId : String) : User
SystemRepository
(controller::database)
-serialVersionUID : long = 1L
-students : Map<String, Student> = new HashMap<>()
-careerStaff : Map<String, CareerStaff> = new HashMap<>()
-pendingReps : Map<String, CompanyRep> = new HashMap<>()
-approvedReps : Map<String, CompanyRep> = new HashMap<>()
-requests : Map<String, Request> = new HashMap<>()
+addApplication(studentId : String, app : Application) : void
+addCareerStaff(staff : CareerStaff) : void
+addInternship(repId : String, intern : InternshipOpportunity) : void
+addStudent(student : Student) : void
+addRequest(request : Request) : void
+removeRequest(requestId : String) : void
+applicationByStudent(studentId : String) : List<Application>
+approveCompanyRep(repId : String) : void
+getRequest(requestId : String) : Request
+findUser(userId : String) : User
+findInternshipOpportunity(internshipId : String) : InternshipOpportunity
+findApplication(applicationId : String) : Application
+getAllInternships() : List<InternshipOpportunity>
+getAllRequests(type : Class<T>) : List<T>
+registerCompanyRep(rep : CompanyRep) : void
+getInternshipsByCompany(companyName : String) : List<InternshipOpportunity>
-resposistory
-repo
1
<<Interface>>
IRepository
(controller::database)
+getApprovedReps() : Map<String, CompanyRep>
+getCareerStaff() : Map<String, CareerStaff>
+getPendingReps() : Map<String, CompanyRep>
+getStudents() : Map<String, Student>
+addCareerStaff(staff : CareerStaff) : void
+addStudent(student : Student) : void
+approveCompanyRep(repId : String) : void
+registerCompanyRep(rep : CompanyRep) : void
1
-repo
1
#repo
-sysRepo
1
AuthenticationService
(controller::service)
-resposistory : IRepository
+AuthenticationService(resposistory : IRepository)
+authenticate(userId : String, password : String) : User
+changePassword(userId : String, oldPassword : String, newPassword : String) : void
-auth
#auth
1
UserController
(controller::control::user)
#auth : AuthenticationService
#filter : FilterCriteria
#repo : IRepository
#requestService : RequestService
+UserController(auth : AuthenticationService, repo : IRepository, requestService : RequestService)
+changePassword(oldPass : String, newPass : String, user : User, confirmPas : String) : void
CsvLoader
(controller::database)
-directoryPath : String
-sysRepo : IRepository
+CsvLoader(sysRepo : IRepository, directoryPath : String)
+CsvLoader(sysRepo : IRepository)
+load() : IRepository
+loadSingleFile(file : File) : void
-loadStudents(reader : BufferedReader) : void
-loadStaff(reader : BufferedReader) : void
<<Interface>>
IInternshipRepo
(controller::database)
+getAllInternships() : List<InternshipOpportunity>
+getInternshipsByCompany(companyName : String) : List<InternshipOpportunity>
+findInternshipOpportunity(internshipId : String) : InternshipOpportunity
+addInternship(repId : String, intern : InternshipOpportunity) : void
1
-repository
RepController
(controller::control::user)
-rep : CompanyRep
-applicationService : ApplicationService
-internshipService : InternshipService
+RepController(auth : AuthenticationService, repository : IRepository, requestService : RequestService, rep : CompanyRep, internshipService : InternshipService, applicationService : ApplicationService)
+launch(systemController : SystemController) : void
+createInternship(title : String, description : String, level : InternshipLevel, preferredMajors : String, openingDate : LocalDate, closingDate : LocalDate, numOfSlots : int) : void
+toggleVisibility(internshipId : String, visibility : boolean) : void
+getInternships() : List<InternshipOpportunity>
+approveApp(appId : String) : void
+rejectApp(appId : String) : void
+deleteInternship(internshipId : String) : void
+editInternship(internshipId : String, title : String, description : String, preferredMajors : String, openingDate : LocalDate, closingDate : LocalDate, slot : Integer, level : InternshipLevel) : void
+validateInternshipId(internshipId : String) : void
+parseLevel(level : String, allowNull : boolean) : InternshipLevel
+checkValidSlots(slots : Integer, allowNull : boolean) : void
+getInternshipById(InternId : String) : InternshipOpportunity
+getStudentById(studentId : String) : User
+getPendApplications(internshipId : String) : List<Application>
+getApproveApplications(internshipId : String) : List<Application>
+getFilteredInternships(internshipOpportunities : List<InternshipOpportunity>, filter : FilterCriteria) : List<InternshipOpportunity>
+getPendingInternships(internshipOpportunities : List<InternshipOpportunity>) : List<InternshipOpportunity>
+getApprovedInternships(internshipOpportunities : List<InternshipOpportunity>) : List<InternshipOpportunity>
+getCanDeleteInternships(internshipOpportunities : List<InternshipOpportunity>) : List<InternshipOpportunity>
rep
1
-internshipService
1
-internshipService
InternshipService
(controller::service)
-repository : IInternshipRepo
-MAX_ACTIVE_INTERNSHIPS : int = 5
+InternshipService(repository : IRepository)
+proposeInternship(title : String, description : String, level : InternshipLevel, preferredMajors : String, openingDate : LocalDate, closingDate : LocalDate, numOfSlots : int, createdBy : String, companyName : String) : InternshipOpportunity
+setVisibility(internshipId : String, visible : boolean) : void
-ableToEditInternship(internship : InternshipOpportunity) : boolean
+editInternship(internshipId : String, title : String, description : String, preferredMajors : String, openingDate : LocalDate, closingDate : LocalDate, slot : Integer, level : InternshipLevel) : void
+removeInternship(repId : String, internshipId : String) : void
+getFilteredInternship(internList : List<InternshipOpportunity>, filter : FilterCriteria) : List<InternshipOpportunity>
+getInternshipsByCompany(companyName : String) : List<InternshipOpportunity>
+getEligibleInternships(s : Student) : List<InternshipOpportunity>
+findInternshipById(internshipId : String) : InternshipOpportunity
+isEligible(s : Student, internshipId : String) : boolean
+isEligibleToCreateInternship(companyName : String) : boolean
+addPendApplicationToInternship(app : Application) : void
+addAcceptedApplicationToInternship(app : Application) : void
+removeApplicationFromInternship(app : Application) : void
+isFilled(internshipId : String) : boolean
+isFilled(internship : InternshipOpportunity) : boolean
+parseLevel(s : String, allowNull : boolean) : InternshipLevel
+checkValidSlots(slots : Integer, allowNull : boolean) : void
+addInternship(internship : InternshipOpportunity) : void
+getPendingInternships(internList : List<InternshipOpportunity>) : List<InternshipOpportunity>
+getApprovedInternships(internList : List<InternshipOpportunity>) : List<InternshipOpportunity>
+getPendingRejectedInternships(internList : List<InternshipOpportunity>) : List<InternshipOpportunity>
-internshipService
1
-internshipService
StudentController
(controller::control::user)
-student : Student
-applicationService : ApplicationService
-internshipService : InternshipService
+StudentController(auth : AuthenticationService, repo : IRepository, request : RequestService, internshipService : InternshipService, applicationService : ApplicationService, student : Student)
+launch(systemController : SystemController) : void
+viewFilteredInternships(filter : FilterCriteria) : List<InternshipOpportunity>
+applyInternship(internshipId : String) : void
+acceptPlacement(applicationId : String) : void
+withdrawPlacement(appId : String, reason : String) : void
+myApplications() : List<Application>
+getStudentById(studentId : String) : User
+getInternshipById(InternId : String) : InternshipOpportunity
student
1
-applicationService
1
-applicationService
-applicationService
-applicationService
1
ApplicationService
(controller::service)
-systemRepository : IApplicationRepo
-MAX_ACTIVE_APPLICATIONS : int = 3
+ApplicationService(systemRepository : IRepository)
+apply(studentId : String, internshipId : String) : Application
+acceptApplication(studentId : String, applicationId : String) : void
+reviewApplication(appId : String, approve : boolean) : void
+findApplication(applicationId : String) : Application
+validStatusTransition(current : ApplicationStatus, next : ApplicationStatus) : boolean
+addApplication(application : Application) : void
+getUserById(studentId : String) : User

(2nd half of the Control UML diagram)

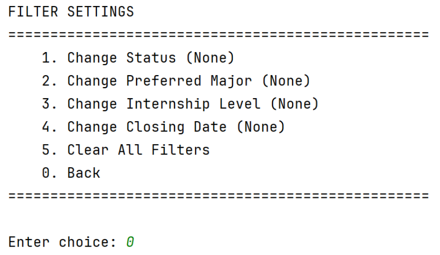
2.3 BOUNDARY DIAGRAM


Visual Paradigm Standard(cristianchye(Nanyang Technological University))
RepMenuUI
(boundary::usermenu)
-repController : RepController
-systemController : SystemController
+RepMenuUI(repController : RepController, systemController : SystemController)
+displayMenu() : void
-createInternshipUI() : void
-manageInternshipsUI() : void
-handleVisibility() : void
-handleEditInternship() : void
-handleApplications() : void
-manageDeleteInternship() : void
FilterUI
(boundary)
-DMY : DateTimeFormatter = DateTimeFormatter.ofPattern("yyyy-MM-dd")
+update(filter : FilterCriteria) : void
-updateStatus(filter : FilterCriteria) : void
-updateMajor(filter : FilterCriteria) : void
-updateLevel(filter : FilterCriteria) : void
-updateClosingDate(filter : FilterCriteria) : void
-clearAll(filter : FilterCriteria) : void
-display(obj : Object) : String
Exit
(boundary::terminal
)
-Exit()
+exit() : void
Welcome
(boundary::terminal)
+welcome(controller : SystemController) : void
LoginPageUI
(boundary::terminal)
+login(controller : SystemController) : void
+register(controller : SystemController) : void
StaffUI
(boundary::usermenu)
-systemController : SystemController
-staffController : StaffController
+StaffUI(systemController : SystemController, staffController : StaffController)
+menuLoop() : void
-viewFilteredInternships() : void
-viewApplications(filteredList : List<InternshipOpportunity>) : void
-viewPendingReq(requestType : String) : void
-approveRejectReq(requestType : String) : void
ChangePasswordUI
(boundary::terminal)
+handleChangePassword(systemController : SystemController, userController : UserController, user : User) : void
StudentUI
(boundary::usermenu)
-studentController : StudentController
-systemController : SystemController
+StudentUI(systemController : SystemController, studentController : StudentController)
+menu() : void
-handleInternships() : void
-handleApplications() : void
-handleChangeFilter() : void
DisplayableViewer
(boundary::viewer)
+displaySingle(displayable : T) : void
+displayList(displayableList : List<T>) : void
+displayList(displayableList : List<T>, message : String) : void
+displaySingle(displayable : T, userController : UserController) : void
+displayList(displayableList : List<T>, message : String, userController : UserController) : void
+displayList(displayableList : List<T>, userController : UserController) : void
<<use>>

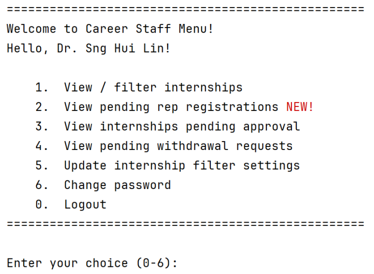
3. DETAILED UML SEQUENCE DIAGRAM


Visual Paradigm Standard(cristianchye(Nanyang Technological University))
CompanyRep
RepUI
Rep Controller
ApplicationService
InternshipService
SystemRepository
1: selects AcceptApplication
1.1: prompts index of application
2: enter index
2.1: approveApp(appID)
2.1.1: findApplication(appID)
2.1.1.1: findApplication(appID)
2.1.1.2: Application
2.1.2: Application
2.1.3: findInternshipById(internshipID)
2.1.3.1: findInternshipOpportunity(internshipID)
2.1.3.2: Internship
2.1.4: Internship
alt
[Internship exists, belongs to companyRep and max number of slots not filled]
2.1.5: reviewApplication(appID, true)
2.1.6: addAcceptedApplicationToInternship(appID)
2.1.6.1: findInternshipById(internshipID)
2.1.7: return confirmation of approval
2.1.8: return confirmation
else
throw ObjectNotFoundException
throw SecurityException
throw MaxExceedException

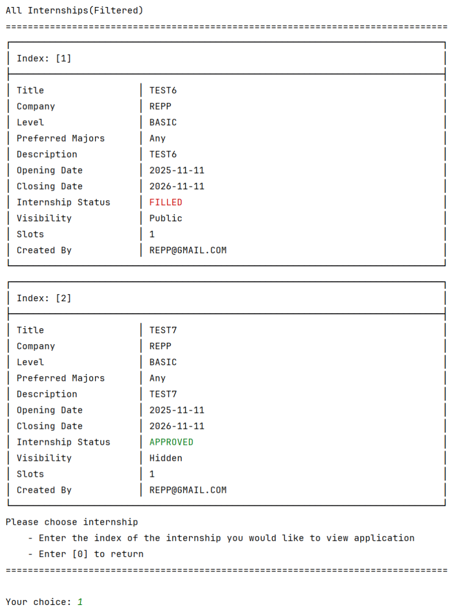
4. TEST CASES AND RESULT



```
FILTER SETTINGS
==================================================
    1. Change Status (None)
    2. Change Preferred Major (None)
    3. Change Internship Level (None)
    4. Change Closing Date (None)
    5. Clear All Filters
    0. Back
==================================================

Enter choice: 0
```

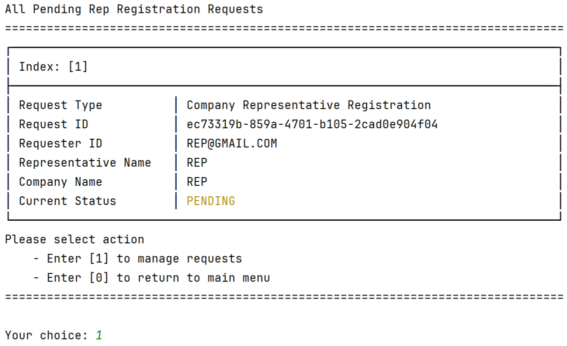


```
==================================================
Welcome to Career Staff Menu!
Hello, Dr. Sng Hui Lin!

    1.  View / filter internships
    2.  View pending rep registrations NEW!
    3.  View internships pending approval
    4.  View pending withdrawal requests
    5.  Update internship filter settings
    6.  Change password
    0.  Logout
==================================================

Enter your choice (0-6):
```

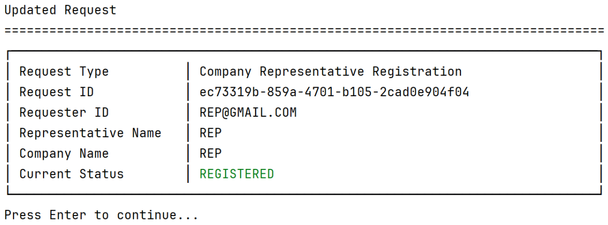


```
All Internships(Filtered)
==============================================================================

┌──────────────────────────────────────────────────────────────────────────┐
│ Index: [1]                                                               │
├──────────────────────────────────────────────────────────────────────────┤
│ Title                  │ TEST6                                           │
│ Company                │ REPP                                            │
│ Level                  │ BASIC                                           │
│ Preferred Majors       │ Any                                             │
│ Description            │ TEST6                                           │
│ Opening Date           │ 2025-11-11                                      │
│ Closing Date           │ 2026-11-11                                      │
│ Internship Status      │ FILLED                                          │
│ Visibility             │ Public                                          │
│ Slots                  │ 1                                               │
│ Created By             │ REPP@GMAIL.COM                                  │
└──────────────────────────────────────────────────────────────────────────┘

┌──────────────────────────────────────────────────────────────────────────┐
│ Index: [2]                                                               │
├──────────────────────────────────────────────────────────────────────────┤
│ Title                  │ TEST7                                           │
│ Company                │ REPP                                            │
│ Level                  │ BASIC                                           │
│ Preferred Majors       │ Any                                             │
│ Description            │ TEST7                                           │
│ Opening Date           │ 2025-11-11                                      │
│ Closing Date           │ 2026-11-11                                      │
│ Internship Status      │ APPROVED                                        │
│ Visibility             │ Hidden                                          │
│ Slots                  │ 1                                               │
│ Created By             │ REPP@GMAIL.COM                                  │
└──────────────────────────────────────────────────────────────────────────┘
Please choose internship
    - Enter the index of the internship you would like to view application
    - Enter [0] to return
==============================================================================

Your choice: 1
```

Staff can view all internships based on their filter and see applications of chosen internships

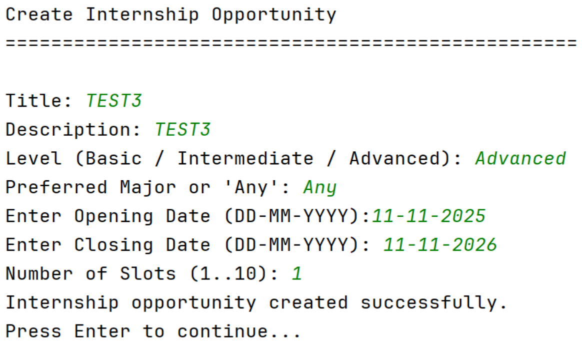


```
All Pending Rep Registration Requests
==============================================================================

┌──────────────────────────────────────────────────────────────────────────┐
│ Index: [1]                                                               │
├──────────────────────────────────────────────────────────────────────────┤
│ Request Type           │ Company Representative Registration             │
│ Request ID             │ ec73319b-859a-4701-b105-2cad0e904f04            │
│ Requester ID           │ REP@GMAIL.COM                                   │
│ Representative Name    │ REP                                             │
│ Company Name           │ REP                                             │
│ Current Status         │ PENDING                                         │
└──────────────────────────────────────────────────────────────────────────┘

Please select action
    - Enter [1] to manage requests
    - Enter [0] to return to main menu
==============================================================================

Your choice: 1
```

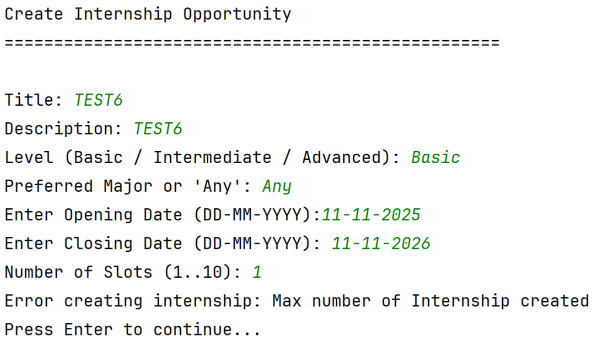


```
Updated Request
==============================================================================
┌────────────────────────────────────────────────────────────────────────────┐
│ Request Type           │ Company Representative Registration               │
│ Request ID             │ ec73319b-859a-4701-b105-2cad0e904f04              │
│ Requester ID           │ REP@GMAIL.COM                                     │
│ Representative Name    │ REP                                               │
│ Company Name           │ REP                                               │
│ Current Status         │ REGISTERED                                        │
└────────────────────────────────────────────────────────────────────────────┘
Press Enter to continue...
```

Staff can see all pending requests(internship vetting, registration, withdrawals) and manage (approve or reject) them.

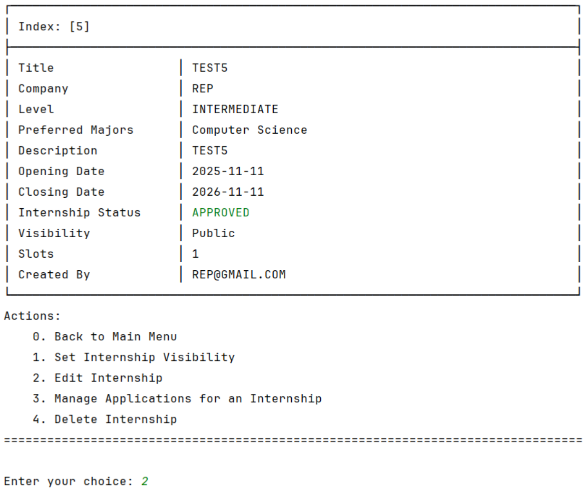


```
Create Internship Opportunity
==================================================

Title: TEST3
Description: TEST3
Level (Basic / Intermediate / Advanced): Advanced
Preferred Major or 'Any': Any
Enter Opening Date (DD-MM-YYYY):11-11-2025
Enter Closing Date (DD-MM-YYYY): 11-11-2026
Number of Slots (1..10): 1
Internship opportunity created successfully.
Press Enter to continue...
```

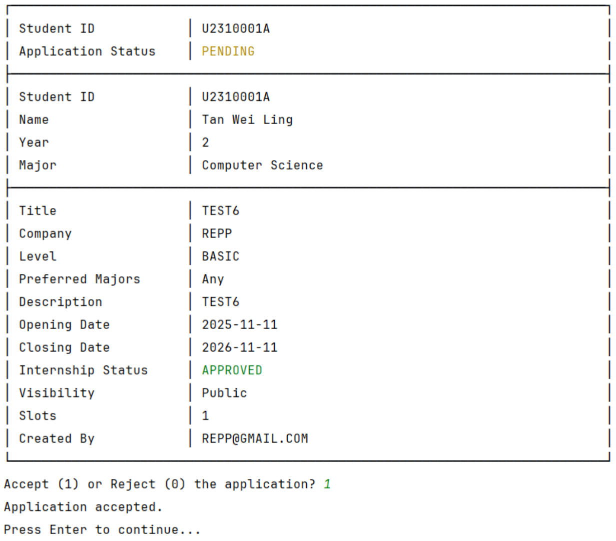


```
Create Internship Opportunity
==================================================

Title: TEST6
Description: TEST6
Level (Basic / Intermediate / Advanced): Basic
Preferred Major or 'Any': Any
Enter Opening Date (DD-MM-YYYY):11-11-2025
Enter Closing Date (DD-MM-YYYY): 11-11-2026
Number of Slots (1..10): 1
Error creating internship: Max number of Internship created
Press Enter to continue...
```

Company representatives can create maximum of 5 internships

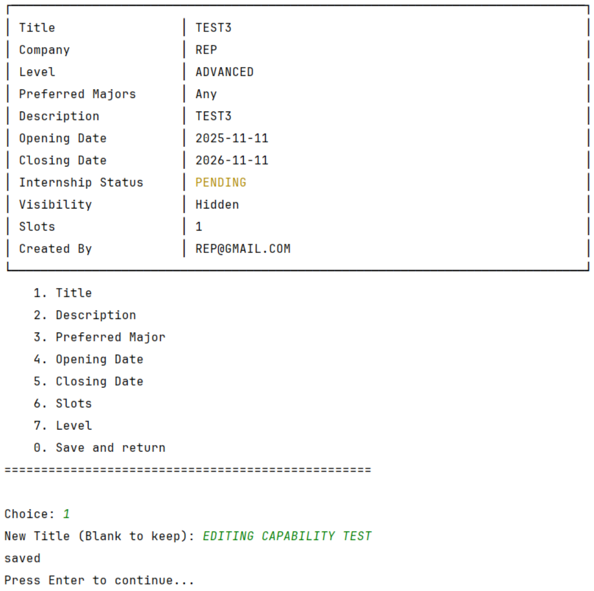


```
| Index: [5]                                                  |
|                                                             |
| Title                 | TEST5                               |
| Company               | REP                                 |
| Level                 | INTERMEDIATE                        |
| Preferred Majors      | Computer Science                    |
| Description           | TEST5                               |
| Opening Date          | 2025-11-11                          |
| Closing Date          | 2026-11-11                          |
| Internship Status     | APPROVED                            |
| Visibility            | Public                              |
| Slots                 | 1                                   |
| Created By            | REP@GMAIL.COM                       |

Actions:
    0. Back to Main Menu
    1. Set Internship Visibility
    2. Edit Internship
    3. Manage Applications for an Internship
    4. Delete Internship
================================================================================

Enter your choice: 2
```

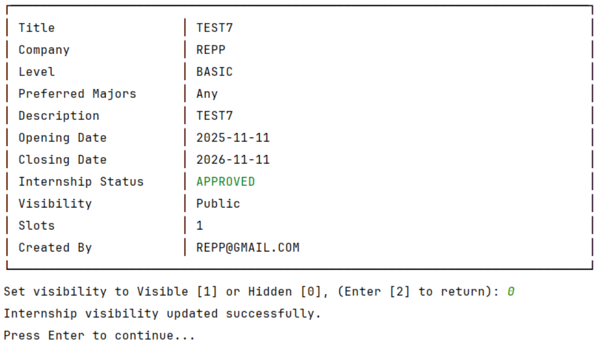


```
| Student ID            | U2310001A                             |
| Application Status    | PENDING                               |
|                                                               |
| Student ID            | U2310001A                             |
| Name                  | Tan Wei Ling                          |
| Year                  | 2                                     |
| Major                 | Computer Science                      |
|                                                               |
| Title                 | TEST6                                 |
| Company               | REPP                                  |
| Level                 | BASIC                                 |
| Preferred Majors      | Any                                   |
| Description           | TEST6                                 |
| Opening Date          | 2025-11-11                            |
| Closing Date          | 2026-11-11                            |
| Internship Status     | APPROVED                              |
| Visibility            | Public                                |
| Slots                 | 1                                     |
| Created By            | REPP@GMAIL.COM                        |

Accept (1) or Reject (0) the application? 1
Application accepted.
Press Enter to continue...
```

Company representatives can manage (view, toggle visibility, edit, manage applications, delete) their internships

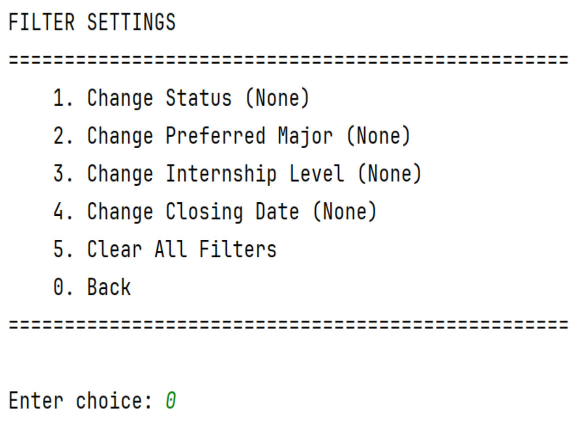


```
| Title                 | TEST3                               |
| Company               | REP                                 |
| Level                 | ADVANCED                            |
| Preferred Majors      | Any                                 |
| Description           | TEST3                               |
| Opening Date          | 2025-11-11                          |
| Closing Date          | 2026-11-11                          |
| Internship Status     | PENDING                             |
| Visibility            | Hidden                              |
| Slots                 | 1                                   |
| Created By            | REP@GMAIL.COM                       |

    1. Title
    2. Description
    3. Preferred Major
    4. Opening Date
    5. Closing Date
    6. Slots
    7. Level
    0. Save and return
==================================================

Choice: 1
New Title (Blank to keep): EDITING CAPABILITY TEST
saved
Press Enter to continue...
```

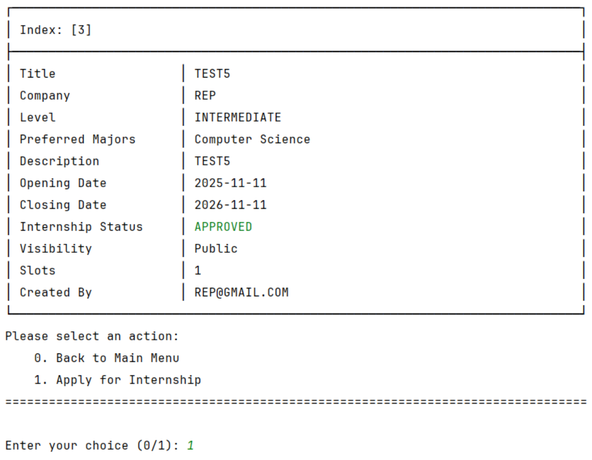


```
| Title                 | TEST7                               |
| Company               | REPP                                |
| Level                 | BASIC                               |
| Preferred Majors      | Any                                 |
| Description           | TEST7                               |
| Opening Date          | 2025-11-11                          |
| Closing Date          | 2026-11-11                          |
| Internship Status     | APPROVED                            |
| Visibility            | Public                              |
| Slots                 | 1                                   |
| Created By            | REPP@GMAIL.COM                      |

Set visibility to Visible [1] or Hidden [0], (Enter [2] to return): 0
Internship visibility updated successfully.
Press Enter to continue...
```

Editing of PENDING internships and toggling visibility of APPROVED internships



```
FILTER SETTINGS
=================================================
    1. Change Status (None)
    2. Change Preferred Major (None)
    3. Change Internship Level (None)
    4. Change Closing Date (None)
    5. Clear All Filters
    0. Back
=================================================


Enter choice: 0
```



```
| Index: [3]                                                |
|                                                           |
| Title                | TEST5                              |
| Company              | REP                                |
| Level                | INTERMEDIATE                       |
| Preferred Majors     | Computer Science                   |
| Description          | TEST5                              |
| Opening Date         | 2025-11-11                         |
| Closing Date         | 2026-11-11                         |
| Internship Status    | APPROVED                           |
| Visibility           | Public                             |
| Slots                | 1                                  |
| Created By           | REP@GMAIL.COM                      |

Please select an action:
    0. Back to Main Menu
    1. Apply for Internship
================================================================================

Enter your choice (0/1): 1
```

Students can change passwords, update internship filters, view internships and apply to maximum 3 eligible ones.

5. REFLECTION

5.1 TRADEOFFS AND AREAS OF IMPROVEMENT

During development we made several conscious decisions and tradeoffs to make sure we delivered a simple and maintainable program within the scope of the course. While these tradeoffs helped us deliver a well structured program, it left room for future improvements to the system.

Security: We decided to go with a simple implementation of password hashing with SHA-256 to demonstrate a working system. For a production level program further improvements should be made to increase security. Implementing a stronger hashing algorithm as well as adding salting to password would further safeguard .

Architecture: The decision to use a strong layered approach increased initial development time and added extra boilerplate code, but follows OO principles, allowing reusability, low coupling and easy teamwork.

Data persistence: Our system uses a simple repository using Java object serialization (.dat files). This allows the system to be lightweight and easy to maintain, while allowing persistence across multiple runs. However, we acknowledge that it is not scalable or concurrent as a full database solution but it is sufficient for a CLI system.

Interface: Keeping the interface in the command line allowed us to focus more on the core logic of the program but sacrificed user friendliness. Making Boundary layer unnecessarily complicated and rigid, moving to web-based or GUI would greatly enhance the experience.

Features: Some user features were intentionally overlooked to keep the system simple and functional. Future enhancements includes:

- Tagging internships like “software” or “finance” would help for search queries.
- Adding the ability for students to save internships they are interested in to a “wishlist”
- Notifications when there is an update on an internship, application or request.
- Forgot password/User Id which would require external library usage

Validation and error handling: More input validation and custom error messages in the interface could also be implemented. We already have quite a lot, but to have a complete robust system would require very strict validation of input considering all edge cases.

5.2 STRUGGLES

We initially had trouble moving on as we were too keen on perfecting the UML class diagram and getting it done, but we realised that it not feasible and its natural for UML to constantly get updated as the project progress. We did not communicate clearly early on which lead to few merge conflicts and redundant copies of method that did the same thing, however we manage to catch it early on due to our weekly meeting. We decide moving on, we will clearly state which portion we will take and wat methods we would require, which create help to de-conflicts issues early on and promoted better collaboration.

5.2 LESSONS LEARNED

We saw how applying OODP concepts like SOLID greatly simplifies development and collaboration. We also appreciated the value of design patterns like factories, as well as the roles of abstract classes and interfaces during our debugging phase as minor changes didn’t result in our code “breaking”. On top of that, UML diagrams and GitHub helped us coordinate effectively as a team on a large codebase. Overall, the project has developed our technical skills and ability to collaborate as a software development team.