Software Requirements Specification

for

CareerKey

Version 1.0 approved

Prepared by:

Muhammad Shahis {BCS-F22-E02}

Muhammad Saqib {BCS-F22-E26}

Nayyab Gull {BCS-F22-E09}

Organization:

University Of Mianwali

Department of Computer Science & IT

Date Created: [08-Jan-26]

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Initial Draft	08-01-2026	Initial document creation	1.0
First Review			
Technical Review			
Final Version			

1. Introduction

CareerKey is a blockchain and AI-based platform designed to digitalize degree attestation and provide intelligent career guidance for students. The system enables secure, transparent, and tamper-proof verification of academic credentials through blockchain technology, while an AI engine analyzes student profiles to recommend suitable job opportunities. By integrating universities, HEC, and career services into a single platform, CareerKey bridges the gap between education and employment.

1.1 Purpose

This document specifies the software requirements for **Career Key**, a comprehensive platform that combines block chain-based academic degree verification with intelligent career recommendation services. The system provides a secure, immutable mechanism for issuing and verifying educational credentials while offering personalized job suggestions to students. This SRS covers the complete system including student, university, and HEC portals, as well as the public verification interface.

1.2 Document Conventions

- ***Font Style:*** *Times New Roman or Arial*
- ***Font Size:***
- ***Headings:*** *18 pt (bold)*
- ***Sub-headings:*** *14 pt (bold)*
- ***Body Text:*** *11 pt*
- ***Text Color:*** *Black*
- ***Page Size:*** *A4*
- ***Line Spacing:*** *1*
- ***Text Alignment:*** *Justified*
- ***Section Titles:*** *Numbered format (e.g., 1.0, 1.1, 2.0)*
- ***Margins:*** *1 inch on all sides*

1.3 Intended Audience and Reading Suggestions

This document is intended for:

- **Developers & Technical Team:** Focus on Sections 2.4, 2.5, 3, 4, and 5 for implementation details.

- **Project Managers & Team Leads:** Review Sections 1.4, 2.1, 2.2, and 5 for project scope and constraints.

- **University & HEC Administrators:** Read Sections 1.4, 2.2, 2.3, and 4.2 to understand system functionality.

- **Clients & Stakeholders:** Begin with Sections 1.1, 1.4, and 2.1 for overview.

- **Testers & QA Team:** Concentrate on Sections 4 and 5 for verification criteria.

1.4 Product Scope

CareerKey is a web-based platform that addresses credential fraud through blockchain technology while assisting students in career development. The system enables HEC-registered universities to issue digitally stamped degrees stored immutably on IPFS and blockchain. Students can request verification, track progress, and receive career recommendations. Employers can instantly verify credentials via a public portal. Benefits include reduced verification time, elimination of fake degrees, and improved student-employer matching. The system aligns with HEC's digital transformation goals and corporate needs for reliable credential verification.

1.5 References

1. HEC Digital Credentials Framework, 2023

2. Ethereum Whitepaper & Developer Documentation

3. IPFS (InterPlanetary File System) Documentation

4. Web3.js Library Documentation

5. "Blockchain for Academic Credentials: A Systematic Literature Review" - Journal of Educational Technology Systems

6. CareerKey Project Charter v1.0

7. Pakistan Personal Data Protection Bill (Draft)

2. Overall Description

2.1 Product Perspective

CareerKey is a new, self-contained product that integrates two previously separate domains: academic credential management and career development. It replaces manual, paper-based verification processes with an automated, secure digital system. The product exists within a larger ecosystem that includes university student databases, HEC national records, blockchain networks, IPFS storage, payment gateways, and job market data sources. While standalone, it will interface with existing university systems for data validation. The diagram below shows the system context:

2.2 Product Functions

- **For Students:** Registration, profile management, degree verification requests with payment, request tracking, verified degree download, career recommendations, job search.

- **For Universities:** Student request review and approval/rejection, request forwarding to HEC, institutional record keeping.

- **For HEC:** University onboarding, final verification decisions, blockchain/IPFS degree issuance, credential management, system oversight.

- **For Public/Employers:** Instant degree verification via QR code or hash without login.

- **System Core:** Blockchain transaction management, IPFS document storage, smart contract execution, payment processing, recommendation algorithms.

2.3 User Classes and Characteristics

1. Students: Primary users. Varied technical expertise. Frequency: Medium to high during degree verification and job search. Need simple, guided interfaces.

2. University Administrators: Moderate technical skills. Frequency: Daily for request processing. Require efficient workflow tools.

3. HEC Officers: High security/privilege level. Technical understanding of blockchain preferred. Frequency: Regular. Need comprehensive oversight capabilities.

4. Employers/Public Verifiers: No login required. Simple, instant verification needs. Frequency: Occasional.

5. System Administrators: Highest technical expertise. Full system access. Frequency: Regular maintenance.

2.4 Operating Environment

- **Client Side:** Modern browsers (Chrome 90+, Firefox 88+, Safari 14+), JavaScript enabled, optional Web3 wallet (MetaMask) for admin functions.

- **Server Side:** Cloud hosting (AWS/Azure), Node.js runtime, Express.js framework, MongoDB database.

- **Blockchain:** Ethereum testnet (Sepolia) for development, potentially mainnet for production.

- **IPFS:** Pinata service or dedicated IPFS node for persistent storage.

- **Payment Gateway:** JazzCash/EasyPaisa/Stripe integration.

- **APIs:** RESTful APIs for internal communication, external APIs for job data (optional).

2.5 Design and Implementation Constraints

1. Technical Constraints:

- Must use Ethereum-based blockchain and IPFS for core functionality

- MERN stack required for web application

- Gas fee optimization essential for cost management

- IPFS pinning service required for permanent storage
- Responsive design must work on mobile browsers

2. Regulatory Constraints:

- HEC maintains final approval authority (implemented in smart contract logic)
- Compliance with Pakistan data protection laws for CNIC and personal data
- Payment gateway must be State Bank of Pakistan approved

3. Project Constraints:

- Must be completed within academic project timeline
- Open-source tools preferred due to budget limitations
- Team's existing blockchain expertise influences technology choices

4. User Environment Constraints:

- Must function with varying internet speeds across Pakistan
- No requirement for specialized hardware
- Browser compatibility with major engines (WebKit, Gecko, Blink)

2.6 User Documentation

- **Online User Manuals:** Separate guides for Students, University Admins, and HEC Officers
- **Context-Sensitive Help:** Integrated help system within application
- **Video Tutorials:** Short demonstration videos for key workflows
- **API Documentation:** For developers integrating with the system
- **Admin Guide:** For system administrators
- **Format:** Web-based HTML with downloadable PDF versions

2.7 Assumptions and Dependencies

Assumptions:

1. Universities have accurate digital records of their alumni
2. HEC maintains current national student databases
3. Users have basic internet access and digital literacy

4. Blockchain network remains accessible and stable

5. IPFS pinning service maintains availability

Dependencies:

1. Ethereum blockchain network availability

2. IPFS node/pinning service uptime

3. Third-party payment gateway integration

4. HEC approval and cooperation for university onboarding

5. University willingness to adopt the new system

3. External Interface Requirements

3.1 User Interfaces

- **Style:** Clean, professional design with HEC branding elements
- **Consistency:** Uniform navigation across all portals
- **Responsive Design:** Mobile-friendly on screens ≥320px width
- **Components:**
  - **Student Portal:** Dashboard, profile editor, request tracker, career recommendations
  - **University Portal:** Request queue, student verification interface, reporting
  - **HEC Portal:** Dashboard, final approval interface, blockchain management
  - **Public Portal:** Simple upload/verify interface with result display
- **Accessibility:** WCAG 2.1 Level AA compliance for users with disabilities
- **Error Handling:** Clear, user-friendly error messages with resolution suggestions

3.2 Hardware Interfaces

- No specialized hardware requirements
- Standard webcam for QR code scanning (optional enhancement)
- Standard printers for degree document printing
- Minimum 2GB RAM for client devices
- Minimum 1024×768 screen resolution

3.3 Software Interfaces

1. Blockchain Interface: Web3.js/Ethers.js library for Ethereum interaction

2. IPFS Interface: ipfs-http-client or Pinata SDK for file storage

3. Payment Gateway: REST API integration with chosen provider

4. Database: Mongoose ODM for MongoDB interaction

5. Email Service: SMTP/API-based service (SendGrid, etc.) for notifications

6. University Systems: CSV/Excel import for batch student verification (manual process)

3.4 Communications Interfaces

- **HTTP/HTTPS:** Primary protocol for web interfaces (HTTPS mandatory for all transactions)

- **REST APIs:** JSON-based APIs for frontend-backend communication

- **WebSocket:** Optional for real-time notification updates

- **Email:** SMTP for user notifications and alerts

- **Blockchain:** JSON-RPC for Ethereum network communication

- **IPFS:** HTTP API for file storage/retrieval

- **Security:** TLS 1.2+ for all external communications

4. System Features

4.1 User Registration and Authentication

4.1.1 Description and Priority: Allows users to create accounts, log in, and manage profiles. High priority as it's the entry point for all users.

4.1.2 Stimulus/Response Sequences:

- Student enters email/password → System creates account → Email verification sent

- User enters credentials → System validates → Grants appropriate portal access

- Admin adds university → System sends invitation → University admin completes registration

4.1.3 Functional Requirements:

- **FR-1**: System shall allow student self-registration with email, password, and basic details. (H)

- **FR-2:** System shall require email verification before account activation. (H)

- **FR-3:** System shall support role-based login (Student, University Admin, HEC Admin). (H)

- **FR-4:** System shall allow password reset via email. (M)

- **FR-5:** System shall maintain login sessions with 30-minute inactivity timeout. (M)

- **FR-6:** System shall encrypt all passwords using bcrypt before storage. (H)

- **FR-7:** System shall allow students to create and edit profiles (education, skills, interests). (M)

- **FR-8:** System shall allow HEC to register and approve universities. (H)

4.2 Degree Verification Workflow System

4.2.1 Description and Priority: Manages the complete degree verification process from request to blockchain issuance. Highest priority as core functionality.

4.2.2 Stimulus/Response Sequences:

- Student submits verification request → Payment processed → Request queues at university

- University admin reviews → Accepts/Rejects → If accepted, forwards to HEC

- HEC officer reviews → Approves/Rejects → If approves, executes blockchain workflow

- System stores on IPFS → Records on blockchain → Notifies student → Degree available for download

4.2.3 Functional Requirements:

- **FR-9:** System shall allow students to submit verification requests with CNIC, roll number, and payment. (H)

- **FR-10:** System shall integrate with payment gateway for fee processing. (H)

- **FR-11:** System shall queue requests for university review with student details. (H)

- **FR-12:** System shall allow university admins to accept or reject requests with reason. (H)

- **FR-13:** System shall automatically forward accepted requests to HEC portal. (H)

- **FR-14:** System shall allow HEC officers to view, approve, or reject requests. (H)

- **FR-15:** System shall execute 4-step verification process upon HEC approval: (H)
  - Apply digital stamp to degree PDF
  - Upload to IPFS and receive hash
  - Record transaction on blockchain with IPFS hash
  - Update student record with verification status

- **FR-16:** System shall notify students at each workflow stage (email/SMS). (M)

- **FR-17:** System shall allow students to track request status in real-time. (M)

- **FR-18:** System shall allow HEC to update or revoke degrees with blockchain record. (M)

- **FR-19:** System shall maintain complete audit log of all verification actions. (H)

4.3 Career Recommendation Engine

4.3.1 Description and Priority: Provides personalized career suggestions based on student profiles. Medium priority as secondary feature.

4.3.2 Stimulus/Response Sequences:

- Student enters career preferences → System analyzes profile → Returns matched jobs

- Student browses recommendations → Clicks on job → Views details and application options

- System periodically updates job database → Recalculates recommendations

4.3.3 Functional Requirements:

- **FR-20:** System shall analyze student profiles (degree, skills, interests) for job matching. (M)

- **FR-21:** System shall allow keyword-based job searches. (M)

- **FR-22:** System shall display personalized job recommendations on student dashboard. (M)

- **FR-23:** System shall provide job details (description, requirements, company, salary range). (M)

- **FR-24:** System shall allow filtering of jobs by category, location, experience level. (L)

- **FR-25:** System shall maintain a database of job opportunities (initially manual entry, API later). (M)

4.4 Blockchain and IPFS Integration

4.4.1 Description and Priority: Provides the immutable storage foundation for verified degrees. Highest priority as core technology.

4.4.2 Stimulus/Response Sequences:

- System prepares verified degree → Uploads to IPFS → Receives content hash

- System calls smart contract → Passes student ID and IPFS hash → Transaction mined

- System receives transaction hash → Stores with student record

- Verifier provides hash → System retrieves from IPFS → Displays verified document

4.4.3 Functional Requirements:

- **FR-26:** System shall encrypt degree documents before IPFS storage. (H)

- **FR-27:** System shall use pinning service to ensure IPFS file persistence. (H)

- **FR-28:** System shall interact with Ethereum smart contract for degree recording. (H)

- **FR-29:** Smart contract shall store: student ID, university ID, IPFS hash, timestamp, HEC signature. (H)

- **FR-30:** System shall handle blockchain transaction confirmation waiting periods. (M)

- **FR-31:** System shall retry failed blockchain transactions with appropriate logging. (M)

- **FR-32:** System shall maintain fallback mechanism if blockchain network is unavailable. (M)

- **FR-33:** Smart contract shall grant HEC admin rights to update records. (H)

4.5 Public Verification Portal

4.5.1 Description and Priority: Allows anyone to verify degree authenticity without login. High priority for employer usability.

4.5.2 Stimulus/Response Sequences:

- User uploads QR code or enters hash → System queries blockchain → Retrieves IPFS hash

- System fetches document from IPFS → Validates digital signature → Displays verification result

- Result shows: Verified/Not Verified, student details (if verified), issuing authority, date

4.5.3 Functional Requirements:

- **FR-34:** System shall provide public web page for degree verification without login. (H)

- **FR-35:** System shall accept QR code image upload or manual hash entry. (H)

- **FR-36:** System shall query blockchain to verify hash existence and retrieve details. (H)

- **FR-37:** System shall fetch and display degree document from IPFS if verified. (H)

- **FR-38:** System shall clearly indicate verification status (GREEN for valid, RED for invalid). (H)

- **FR-39:** System shall display minimal student info (name, degree, university, year) when valid. (M)

- **FR-40:** System shall not store any verification query data for privacy. (H)

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- **Response Time:** Dashboard loading < 3 seconds, verification result < 5 seconds
- **Throughput:** Support 100 concurrent users, 50 simultaneous verification requests
- **Blockchain:** Transaction confirmation within 60 seconds (testnet)
- **IPFS:** Document upload/download within 10 seconds for ≤10MB files
- **Availability:** 99% uptime during business hours (9 AM - 5 PM PKT)
- **Scalability:** Support addition of 50+ universities and 100,000+ students

5.2 Safety Requirements

- No physical safety requirements (software-only product)
- Financial safety: Secure payment processing with transaction rollback capability
- Data safety: Regular encrypted backups with disaster recovery plan
- System shall prevent data loss during blockchain transaction failures

5.3 Security Requirements

- **Authentication:** Strong password policies (min 8 chars, mixed case, numbers)
- **Authorization:** Role-based access control with principle of least privilege
- **Data Encryption:** TLS for transmission, AES-256 for sensitive data at rest
- **Blockchain Security:** Private key management via secure vault (not plaintext storage)
- **Input Validation:** All user inputs validated and sanitized
- **Audit Trail:** Immutable log of all sensitive operations (login attempts, approvals, changes)
- **Session Management:** Secure session handling with invalidation on logout
- **Compliance:** Adherence to HEC security policies and data protection regulations

5.4 Software Quality Attributes

- **Reliability:** Mean time between failures > 720 hours
- **Maintainability:** Modular code with documentation, change implementation within 5 days
- **Usability:** 90% of users can complete verification request without help
- **Portability:** Function across Windows, macOS, Linux, iOS, Android browsers

- **Testability:** Unit test coverage > 80% for critical modules
- **Interoperability:** Standard JSON APIs for potential integration with other systems
- **Recoverability:** System restore within 4 hours of failure

5.5 Business Rules

1. Only HEC-approved universities may issue degrees through the system
2. Students can only request verification for their own degrees
3. Universities can only verify their own alumni
4. HEC has final authority on all degree verifications
5. Verified degrees cannot be modified; only HEC can revoke with new blockchain transaction
6. Payment must be completed before university review begins
7. Each degree gets unique QR code and blockchain hash
8. Public verification shows only authenticity status, not full personal data
9. Career recommendations based only on user-provided profile data
10. System administrators cannot modify blockchain records directly

6. Other Requirements

- **Internationalization:** English as primary language, Urdu translation optional for future
- **Legal:** Terms of service and privacy policy compliant with Pakistani law
- **Database Requirements:** MongoDB cluster with daily backups, 1TB initial storage
- **Reporting:** HEC dashboard with statistics (verifications per university, time metrics, revenue)
- **Monitoring:** System health dashboard with blockchain/IPFS connectivity status
- **Deployment:** Docker containerization for consistent environments
- **Testing:** User acceptance testing with real students and university staff
- **Training:** Conduct workshops for HEC and university administrators
- **Maintenance:** 6-month warranty period with bug fixes and minor enhancements

Appendix A: Glossary

Term	Definition
Blockchain	Distributed digital ledger that records transactions immutably
IPFS	InterPlanetary File System - decentralized storage protocol
HEC	Higher Education Commission of Pakistan
Smart Contract	Self-executing contract with terms written in code on blockchain
Gas Fee	Transaction cost on Ethereum blockchain
Immutable	Cannot be changed or altered after creation
Digital Stamp	Cryptographic signature verifying document authenticity
QR Code	Quick Response code containing verification data
Hash	Unique fixed-length string generated from data using cryptographic function
Web3	JavaScript library for interacting with Ethereum blockchain
MERN Stack	MongoDB, Express.js, React, Node.js - web development stack
PIN	Persist IPFS Now - keeping IPFS files accessible permanently

Appendix B: Analysis Models

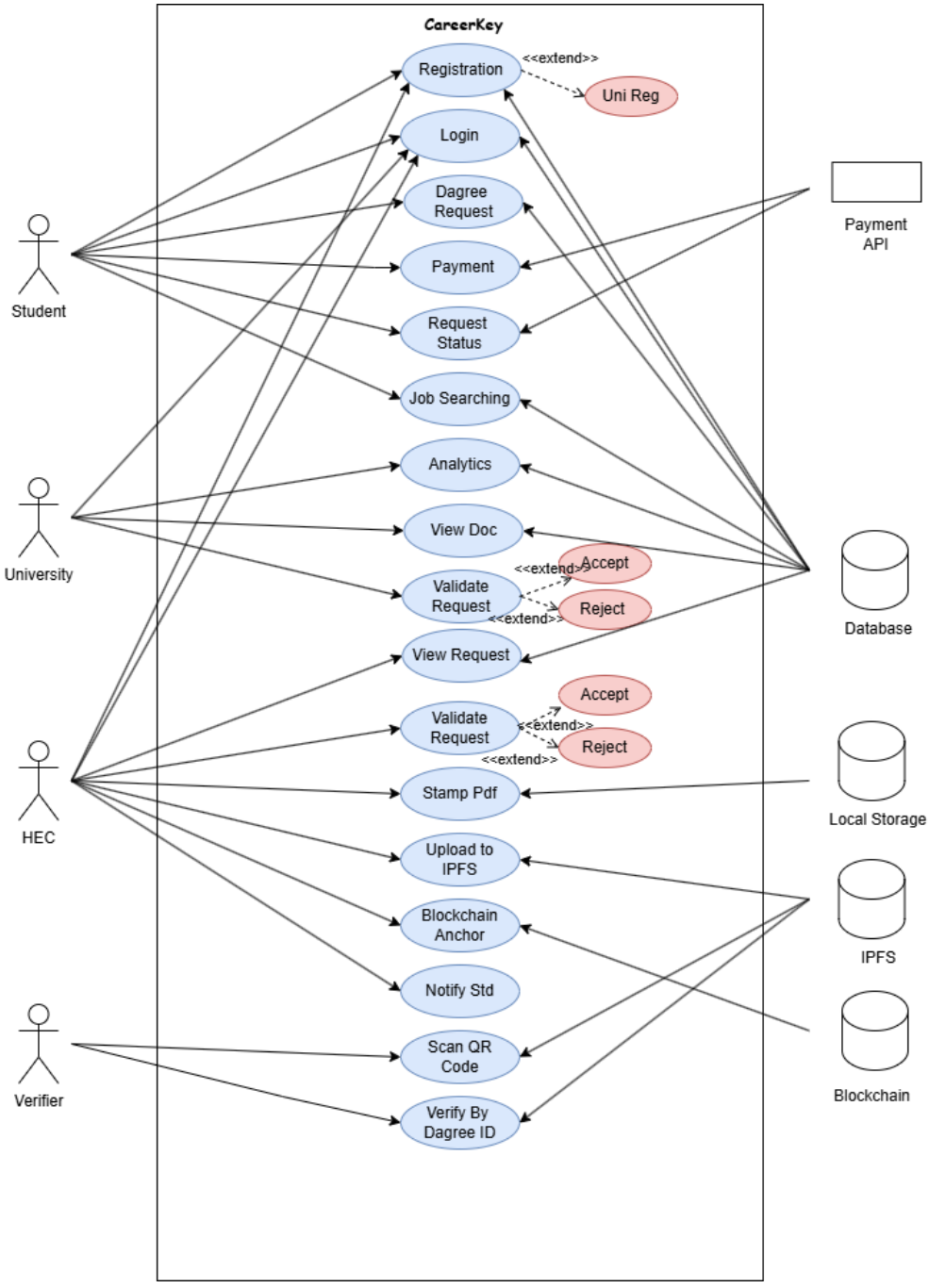
This appendix presents the analysis models used to understand, visualize, and validate the functional and data requirements of the **CareerKey System**. These models help in identifying system actors, workflows, data relationships, and interactions among system components.

B.1 Use Case Model

The Use Case Model illustrates the interactions between users and the CareerKey system. It identifies the system's primary actors and the services provided to them.

Actors

- **Student** – Requests degree attestation and receives career recommendations.
- **University** – Verifies student academic records and submits degree data.
- **HEC (Higher Education Commission)** – Approves or rejects degree attestations.
- **Verifer** – Juat verify dagree by scaning QR Code and via dagree id.

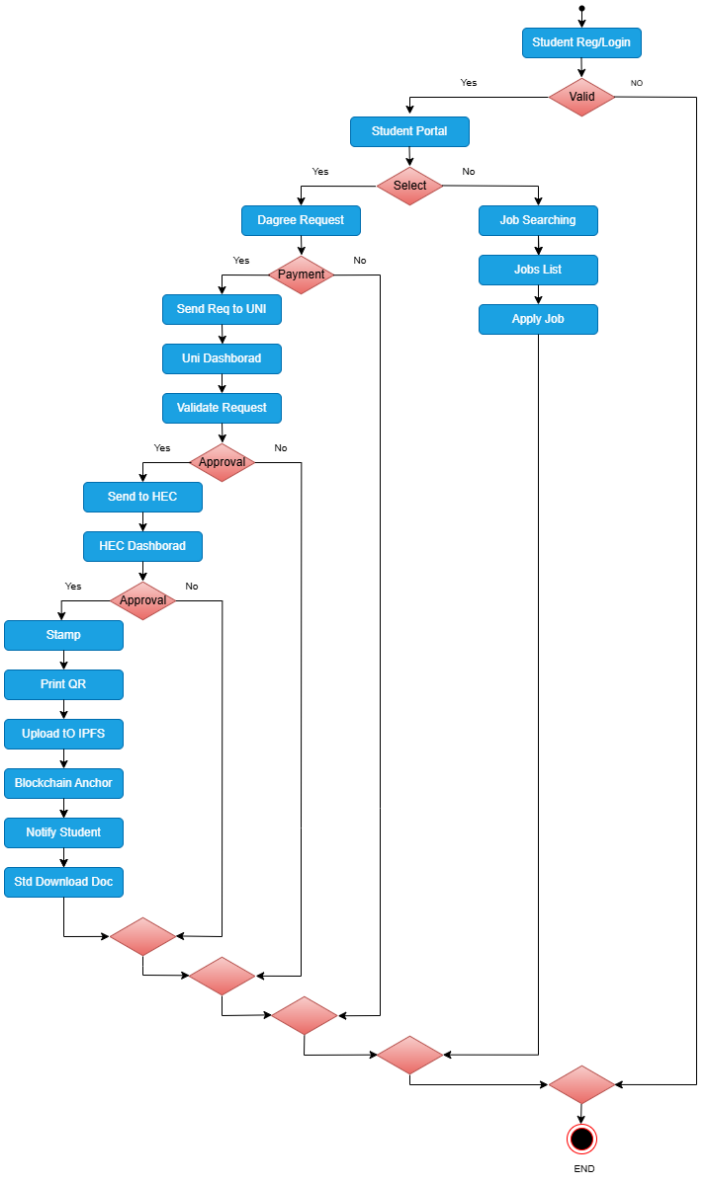


B.2 Activity Diagram

The Activity Diagram represents the workflow of the CareerKey system, showing the sequence of actions performed during degree attestation and career recommendation processes.

Key Activities

- Student logs in using CNIC.
- Student selects a service (Degree Attestation or Career Recommendation).
- University verifies academic data.
- HEC reviews and approves attestation requests.
- Degree hash is stored on the blockchain.
- QR code and transaction ID are generated.
- Student downloads verified documents.
- AI engine analyzes student profiles and recommends jobs.



B.3 Sequence Diagram

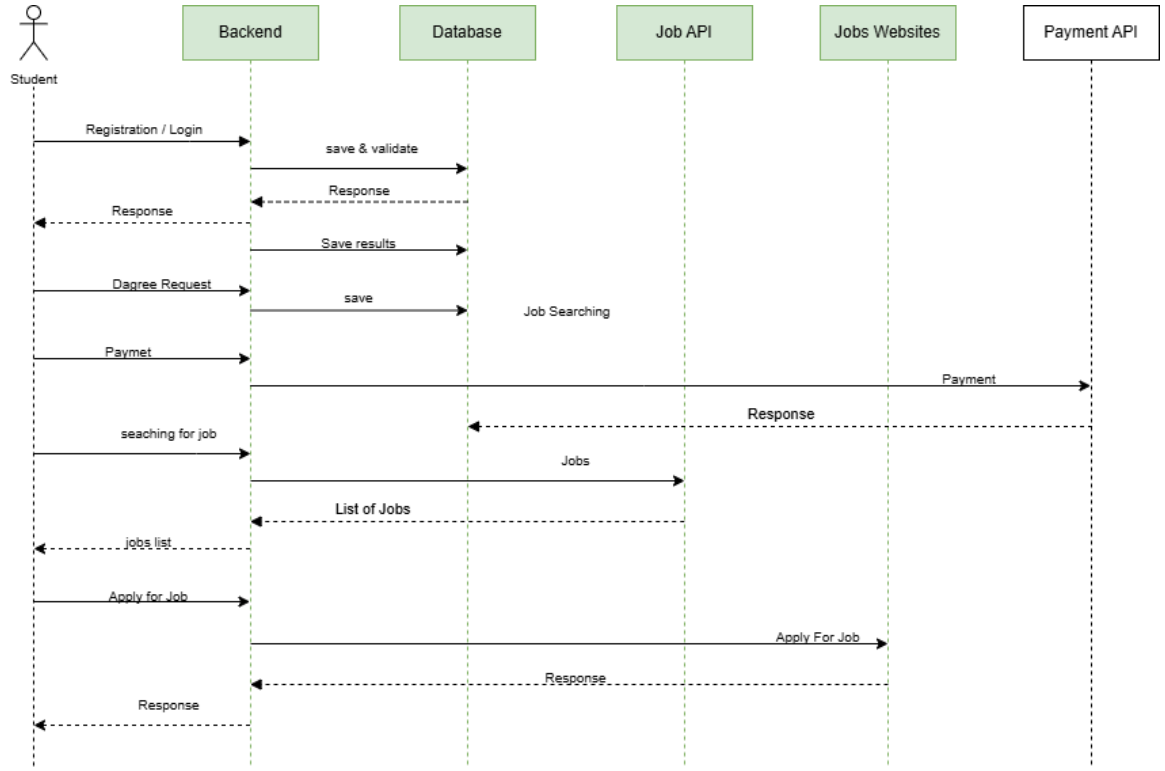
The Sequence Diagram illustrates the time-ordered interaction among system components.

Main Interactions

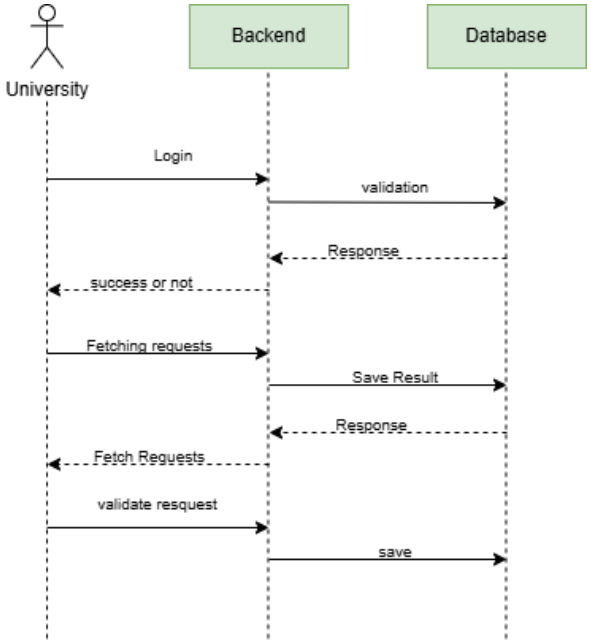
- Student sends attestation request to the system.
- System forwards request to the University.
- University submits verified data to HEC.
- HEC approves the attestation.
- System interacts with the blockchain to store the degree hash.
- AI engine processes career data and returns job recommendations.

This model helps in identifying message flow and dependencies among actors.

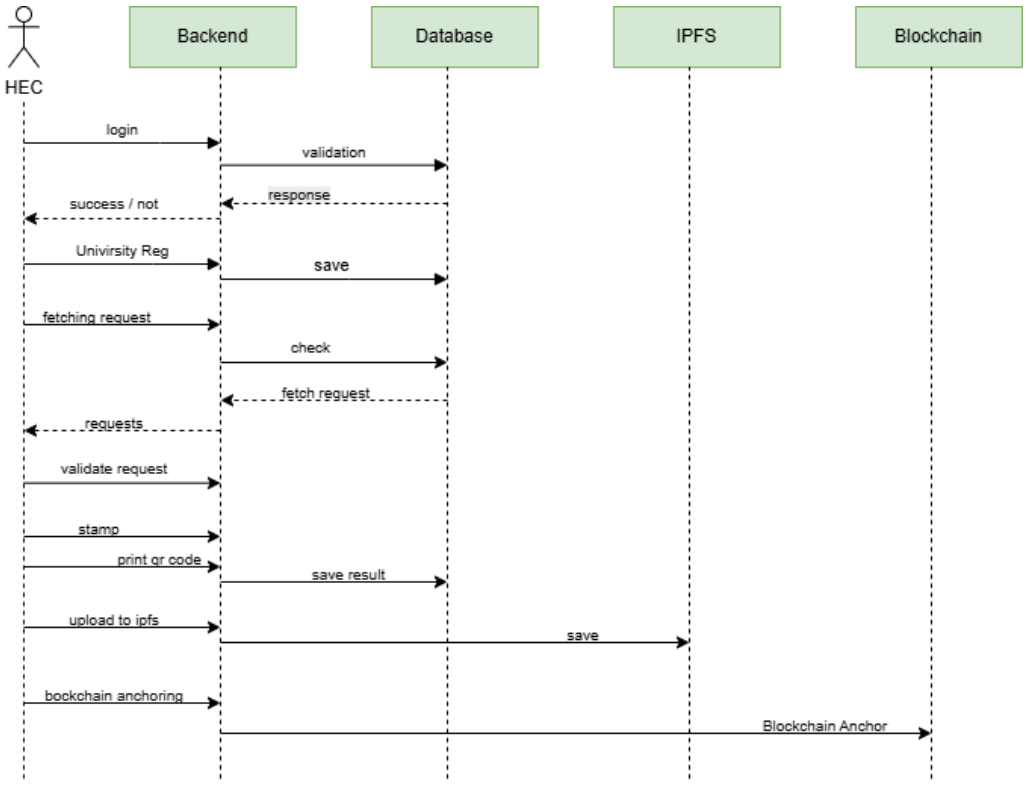
Student Sequance Diagram



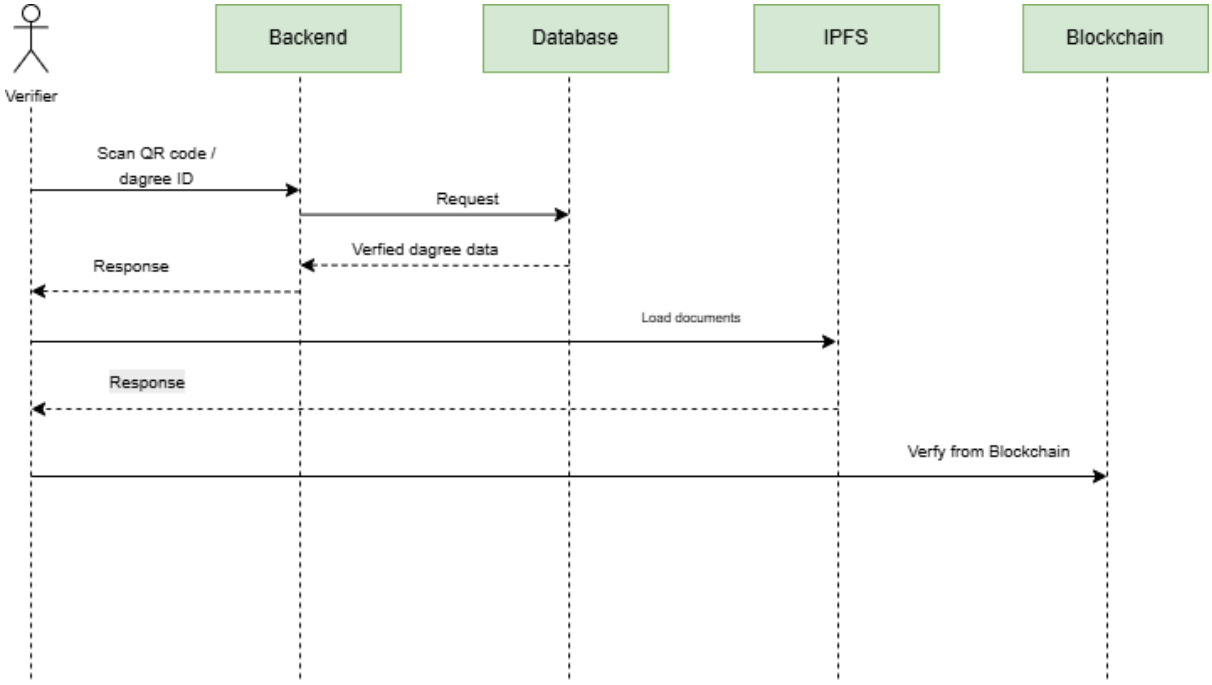
University Sequance Diagram



HEC Sequance Diagram



Verfier Sequance Diagram



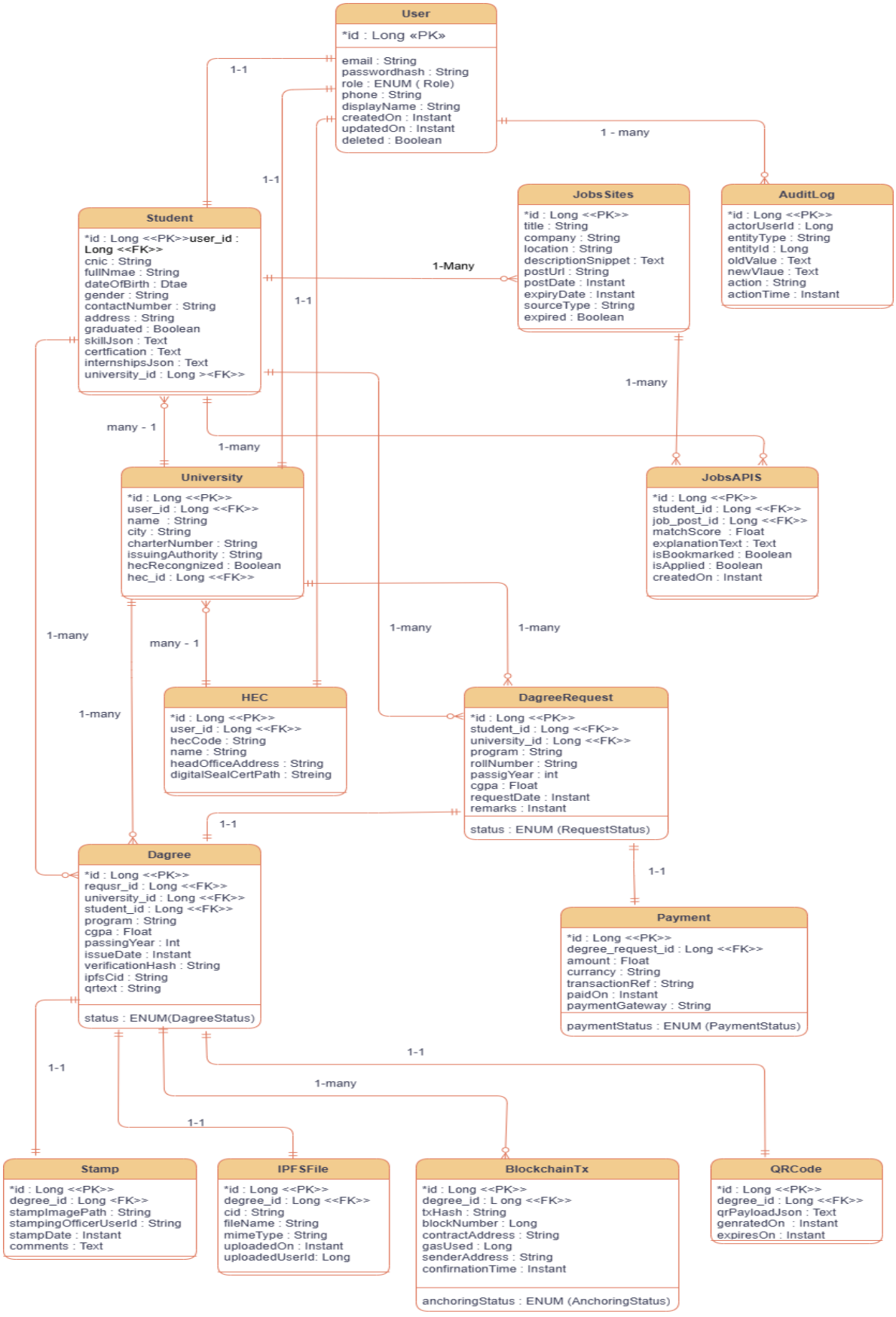
B.4 Class Diagram

The Class Diagram defines the static structure of the system by showing classes, attributes, and relationships.

Key Classes

- Student
- University
- Degree
- AttestationRequest
- HEC
- BlockchainRecord
- CareerProfile
- JobRecommendation

Relationships such as association, aggregation, and inheritance are used to represent system structure.


User
*id : Long «PK»
email : String
passwordhash : String
role : ENUM (Role)
phone : String
displayName : String
createdOn : Instant
updatedOn : Instant
deleted : Boolean
1-1
1 - many
1-1
Student
*id : Long <<PK>>user_id : Long <<FK>>
cnic : String
fullNmae : String
dateOfBirth : Dtae
gender : String
contactNumber : String
address : String
graduated : Boolean
skillJson : Text
certfication : Text
internshipsJson : Text
university_id : Long ><FK>>
1-Many
JobsSites
*id : Long <<PK>>
title : String
company : String
location : String
descriptionSnippet : Text
postUrl : String
postDate : Instant
expiryDate : Instant
sourceType : String
expired : Boolean
AuditLog
*id : Long <<PK>>
actorUserId : Long
entityType : String
entityId : Long
oldValue : Text
newVlaue : Text
action : String
actionTime : Instant
1-1
1-many
many - 1
1-many
University
*id : Long <<PK>>
user_id : Long <<FK>>
name : String
city : String
charterNumber : String
issuingAuthority : String
hecRecongnized : Boolean
hec_id : Long <<FK>>
JobsAPIS
*id : Long <<PK>>
student_id : Long <<FK>>
job_post_id : Long <<FK>>
matchScore : Float
explanationText : Text
isBookmarked : Boolean
isApplied : Boolean
createdOn : Instant
1-many
many - 1
1-many
1-many
1-many
HEC
*id : Long <<PK>>
user_id : Long <<FK>>
hecCode : String
name : String
headOfficeAddress : String
digitalSealCertPath : Streing
DagreeRequest
*id : Long <<PK>>
student_id : Long <<FK>>
university_id : Long <<FK>>
program : String
rollNumber : String
passigYear : int
cgpa : Float
requestDate : Instant
remarks : Instant
status : ENUM (RequestStatus)
1-1
1-1
Dagree
*id : Long <<PK>>
requsr_id : Long <<FK>>
university_id : Long <<FK>>
student_id : Long <<FK>>
program : String
cgpa : Float
passingYear : Int
issueDate : Instant
verificationHash : String
ipfsCid : String
qrtext : String
status : ENUM(DagreeStatus)
Payment
*id : Long <<PK>>
degree_request_id : Long <<FK>>
amount : Float
currancy : String
transactionRef : String
paidOn : Instant
paymentGateway : String
paymentStatus : ENUM (PaymentStatus)
1-1
1-1
1-many
1-1
Stamp
*id : Long <<PK>>
degree_id : Long <FK>>
stampImagePath : String
stampingOfficerUserId : String
stampDate : Instant
comments : Text
IPFSFile
*id : Long <<PK>>
degree_id : Long <<FK>>
cid : String
fileName : String
mimeType : String
uploadedOn : Instant
uploadedUserId: Long
BlockchainTx
*id : Long <<PK>>
degree_id : L ong <<FK>>
txHash : String
blockNumber : Long
contractAddress : String
gasUsed : Long
senderAddress : String
confirnationTime : Instant
anchoringStatus : ENUM (AnchoringStatus)
QRCode
*id : Long <<PK>>
degree_id : Long <<FK>>
qrPayloadJson : Text
genratedOn : Instant
expiresOn : Instant

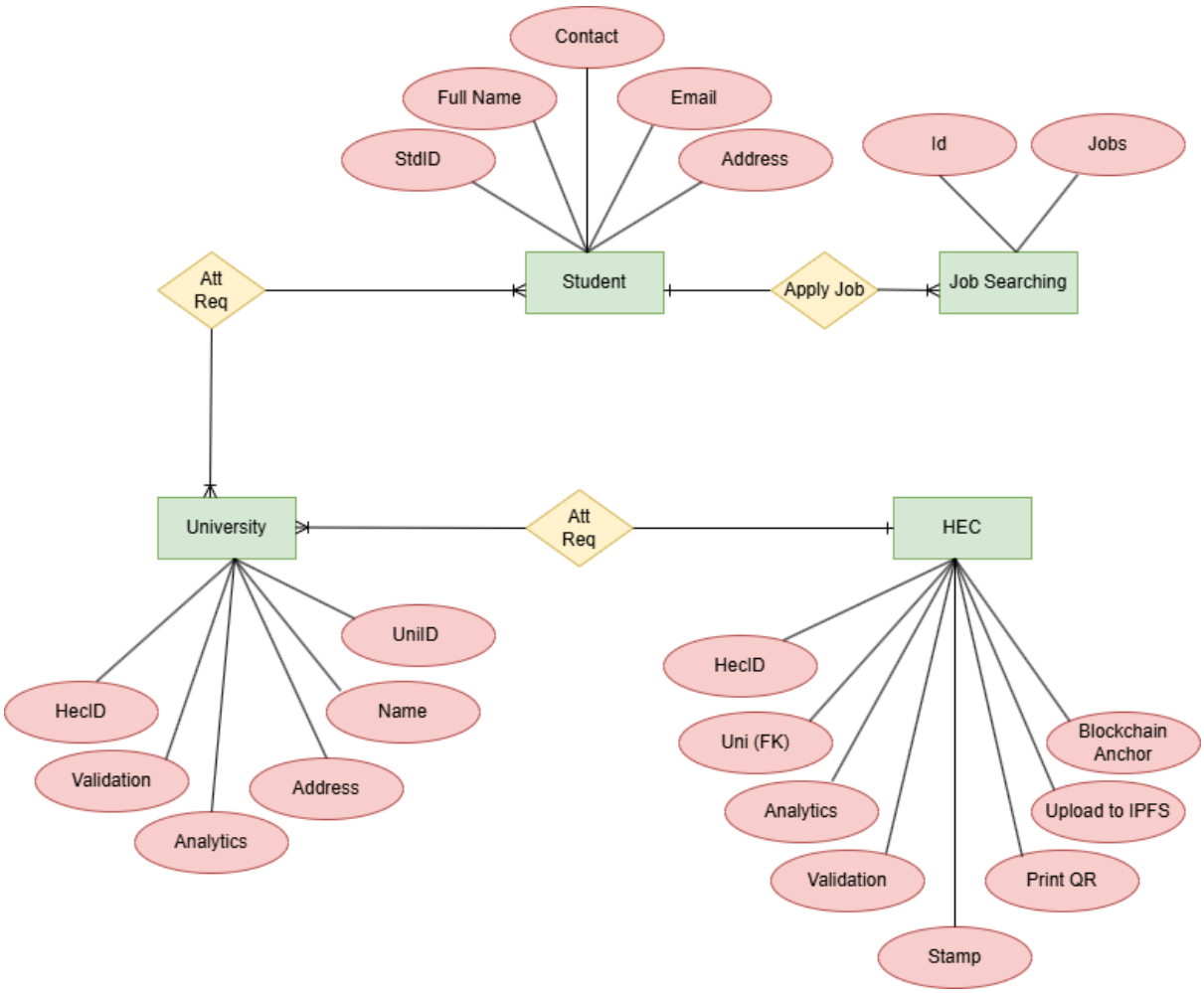
B.5 Entity Relationship (ER) Diagram

The ER Diagram represents the database structure and relationships among entities.

Core Entities

- Student
- University
- Verfier
- HEC

The ER model ensures data consistency, integrity, and proper normalization of the database.



Appendix C: To Be Determined List

At this stage of the **CareerKey** project, some requirements remain to be finalized, including the choice of the final Ethereum deployment network, selection of external job platform APIs, detailed AI model configuration for career recommendations, payment gateway integration specifics, smart contract security audit approach, and final deployment and scalability strategy. These TBD items will be tracked and resolved in later phases of the project and updated in future revisions of the SRS.