TABLE OF CONTENT

INTRODUCTION

In today’s competitive job market, students and professionals face significant challenges in securing employment, accessing skill-building resources, and connecting with industry opportunities. Traditional career services operate in silos, leaving job seekers, employers, and educators without a unified system to bridge these gaps. To address these challenges, we present Career Hive – an innovative Django-based platform designed to revolutionize career development through technology.

Career Hive provides a seamless, integrated solution for students, companies, institutions, and professionals. By leveraging Python Django and SQLite, the platform offers real-time access to jobs, internships, courses, and networking opportunities. Users can effortlessly connect with employers, enroll in skill-building programs, and engage with industry experts – all within a single ecosystem.

The system empowers students with driven job matching, one-click applications, and personalized course recommendations. Companies benefit from streamlined recruitment tools to post jobs, track applicants, and identify top talent. Institutions gain powerful analytics to align curricula with industry demands, while professionals contribute through mentorship and community webinars.

Key Innovations

- Smart Matching: Django-powered algorithms link students with relevant jobs/internships.
- Centralized Learning: Institutions upload courses (videos/PDFs/assignments) with payment integration.
- Community Hub: Professionals host webinars and share industry insights.
- Admin Control: Real-time monitoring of users, complaints, and system analytics.

Built for scalability, the platform’s SQLite database ensures fast performance for early-stage deployment, with a future shift to PostgreSQL for enterprise expansion. By integrating these modules, Career Hive transforms fragmented career services into a dynamic, user-centric ecosystem – reducing unemployment gaps and fostering lifelong professional growth

Chapter – 2

PROJECT OUTLINE

TITLE:

CARRIER HIVE

DEFINITION

Career Hive is an advanced digital platform designed to streamline career development by connecting students, professionals, companies, and institutions. It provides a unified ecosystem where users can access job opportunities, skill-building resources, and industry networking tools.

Built on Django and SQLite, with future scalability to PostgreSQL, Career Hive simplifies the process of job hunting, course enrolment, and professional engagement. It bridges the gap between education and employment, ensuring users have access to the resources they need to grow in their careers.

OBJECTIVES

1. Seamless Job & Internship Matching

Enable students and professionals to easily discover relevant job and internship opportunities through an efficient recommendation system.

2. Centralized Learning & Skill Development

Provide a learning hub where institutions can upload courses (videos, PDFs, assignments) with integrated payment options, allowing users to upskill effectively.

3. Simplified Job Application Process

Offer a one-click application feature that allows users to apply for multiple jobs effortlessly, improving job-seeking efficiency.

4. Enhanced Networking & Community Engagement

Establish a community hub where professionals can share insights, host webinars, and mentor students to enhance career development.

5. Advanced Recruitment Tools for Companies

Enable recruiters to post job listings, track applications, and filter top candidates efficiently, reducing hiring time.

6. Institutional Insights & Analytics

Provide institutions with data-driven insights to align academic programs with industry demands, ensuring better employment outcomes for students.

7. Robust Admin Control & System Monitoring

Ensure platform integrity with real-time monitoring of user activity, complaints, and system performance for smooth operations.

8. Paperless & Automated Processes

Replace traditional recruitment and course enrolment paperwork with a fully digital platform, streamlining operations for users and organizations.

9. Encouraging Lifelong Learning & Career Growth

Support continuous professional development by offering users ongoing access to career-building resources and mentorship.

10. Real-Time Notifications & Updates

Keep users informed with instant notifications on job postings, course availability, and industry events, ensuring they never miss an opportunity.

11. Feedback & Platform Enhancement

Implement a feedback system to allow users to review courses, job postings, and networking events, driving continuous improvements.

12. Multi-Stakeholder Integration

Create a unified career ecosystem that benefits students, job seekers, recruiters, educators, and industry professionals, bridging the gap between education and employment.

SCOPE

1. Geographical Scope

- **Local and National Coverage**: Initially serving students, institutions, and employers within a specific region, with plans for nationwide expansion.
- **Remote Access**: A web-based platform ensures accessibility for students, educators, and employers, even in remote areas.
- **Scalability**: The platform architecture supports expansion into international markets.

2. Target Users

- **Students**: Primary users seeking job opportunities, internships, and skill-building courses.
- **Institutions**: Universities and colleges managing courses, tracking student progress, and aligning curricula with industry needs.
- **Companies**: Employers posting job openings, internships, and recruiting talent.
- **Professionals**: Industry experts engaging with students through mentorship, webinars, and networking opportunities.
- **Administrators**: Managing user accounts, monitoring platform activity, and handling complaints.

3. Functional Scope

- **Job & Internship Management**: Posting, applying, and tracking job opportunities.
- **Course Management**: Institutions can upload learning materials, assignments, and integrate payment options.
- **Application Tracking**: Real-time status updates for students and employers on job and internship applications.
- **Webinar Hosting & Community Engagement**: Professionals conduct webinars, mentorship sessions, and industry discussions.
- **Feedback & Ratings**: Users provide feedback on job listings, courses, and professional interactions to improve platform quality.

4. Technological Scope

- **Web Platform**: A responsive, Django-based platform ensuring accessibility across devices.
- **Database**: SQLite for early-stage data management, with a planned transition to PostgreSQL for scalability.
- **Authentication & Security**: Role-based access control ensuring data privacy and secure login.
- **Payment Processing**: Integrated transactions for course enrollments and premium platform features.

5. Business & Economic Scope

- **Revenue Models**: Monetization strategies include premium features, recruitment fees, and institutional licensing.
- **Employer Partnerships**: Collaboration with companies to create direct hiring pipelines.
- **Institutional Value**: Helps universities improve placement rates and better align academic programs with job market demands.

6. Social Impact Scope

- **Career Accessibility**: Democratizing access to jobs, internships, and professional development.
- **Education-Industry Alignment**: Bridging the gap between academic learning and real-world employment.
- **Community Building**: Fostering industry connections between students, professionals, and educators.

7. Future Scope

- **Mobile Application**: Expanding accessibility through native mobile apps.
- **Advanced Analytics:** Employment trend dashboards for institutions and employers.
- **Corporate Training Integration**: Upskilling and reskilling programs for professionals.

8. Sustainability Scope

- **Paperless Systems**: Digital documentation, reducing administrative waste.
- **Remote Readiness**: Supporting virtual job applications, career counselling, and professional development.

<u>MODULE EXPLANATION</u>

1. Admin Module

The admin is responsible for managing and verifying users, institutions, companies, and communities.

Features:

- **User Verification**: Approve or reject registrations of students, institutions, companies, and communities.
- **Job & Internship Management**: View job details, including company name, job descriptions, and status.
- **Course Management**: Monitor courses uploaded by institutions.
- **Complaint Handling**: Review and address complaints submitted by users.
- **Feedback Management**: View feedback from users and take necessary actions.

2. Company Module

The Company module allows organizations to post and manage job and internship opportunities.

Features:

- **Job Management**: Add, update, or delete job postings.

- **Internship Management**: Add, update, or delete internship postings.
- **Application Tracking**: View and manage applicants who have applied for jobs or internships.

3. Institution Module

Institutions can create and manage learning resources for students.

Features:

- **Course Management**: Create, update, and delete courses.
- **Content Upload**: Upload course materials such as PDFs, videos, and assignments.
- **Payment Approval**: Approve course payments made by students.
- **Student Tracking**: View and manage enrolled students.

4. Student Module

Students can explore career opportunities and enrol in courses.

Features:

- **Course Enrolment**: Enrol in courses and make payments.
- **Content Access**: Download PDFs, watch videos, and access assignments.
- **Job & Internship Applications**: Apply for jobs and internships.
- **Application Tracking**: View the status of job and internship applications.
- **Feedback & Complaints**: Submit complaints and provide feedback.
- **Webinar Access**: View and watch community-provided webinars.

5. Community Module

The Community module allows professionals and institutions to share knowledge.

Features:

- **Webinar Creation**: Create and manage webinars for students and institutions.

SOFTWARE CONFIGURATION

It is very important to select the appropriate software so that the software works properly.
Below is the software that is required to make the new system:

System Specification

A Software Requirements Specification (SRS) document ensures that Career Hive’s development aligns with user needs and industry expectations. It bridges the gap between clients (institutions, employers), end-users (students, professionals), and developers.

Key Aspects of the SRS:

- **Client Needs Identification** – Defining platform goals, job search automation, and skill development features.
- **User Requirement Analysis** – Understanding student expectations, employer demands, and institutional requirements.
- **Software Development Planning** – Translating requirements into Django-based implementation with PostgreSQL for future scalability.

SYSTEM REQUIREMENTS

H/W REQUIREMENTS

PROCESSOR	: INTEL DUAL CORE or above
HARD DISK	: 160 GB
INPUT DEVICES	: KEYBOARD, MOUSE
OUTPUT DEVICES	: MONITOR

S/W REQUIREMENTS

- OPERATING SYSTEM: WINDOWS 7
- FRONT END: HTML,CSS ,PYTHON,JAVA
- BACK END: SQLITE2
- FRAMEWORK: DJANGO (PYTHON-BASED WEB FRAMEWORK)
- PLATFORM USED: VISUAL STUDIO CODE (VS CODE)

Chapter – 3

SYSTEM ANALYSIS

INTRODUCTION

System analysis is the process of gathering data, diagnosing problems, and defining the requirements for a new system. In software development, structural analysis plays a crucial role in ensuring that the system meets user needs and operates efficiently. During this phase, information is collected through various methods such as document analysis, user feedback, and observation of existing processes.

For Career Hive, system analysis helps define the platform's objectives, assess existing career development challenges, and determine the best approach to bridge the gap between students, institutions, employers, and professionals. The project focuses on revitalizing and simplifying career services to provide an efficient, user-friendly experience.

A preliminary study was conducted using fact-finding techniques such as surveys, comparisons with traditional career services, and evaluations of current job-matching platforms. The findings helped shape the platform's core functionalities, ensuring that it meets the needs of users while enhancing career development opportunities.

Objectives of System Analysis:

- **Identifying the Need**: Understanding career-related challenges faced by students, institutions, and employers.
- **Analysing the Existing and Proposed System**: Evaluating traditional job placement methods and designing an improved digital solution.
- **Evaluating Feasibility**: Assessing economic, technical, and operational feasibility for successful implementation.
- **Performing Economic and Technical Analysis**: Ensuring cost-effectiveness and efficient technology adoption.
- **Identifying Hardware & Software Requirements**: Defining the technology stack needed for smooth operations.
- **Allocating Functions to Hardware & Software**: Ensuring optimal system performance and scalability.

IDENTIFICATION OF NEED

System analysis involves breaking down the entire career development process to identify interrelated components and define the platform's role. Career Hive is designed to provide an integrated career ecosystem that connects students with job opportunities, internships, learning resources, and industry professionals.

The current career services landscape is fragmented, with separate platforms for job searching, learning, and networking. Career Hive eliminates these inefficiencies by combining job matching, skills development, and professional networking into a single platform.

DEFINITION

Career Hive is an innovative career development platform designed to help students and professionals access jobs, internships, learning resources, and industry connections. The system integrates AI-driven job matching, course enrollment features, and networking tools to create a seamless career ecosystem.

Built using Django and SQLite, with plans for a future transition to PostgreSQL, the platform ensures scalability and high performance. By leveraging technology, Career Hive simplifies the job-seeking process, enhances institutional insights, and improves recruitment efficiency.

OBJECTIVES

1. **Streamline Job & Internship Matching**

To provide students with AI-powered job recommendations and allow them to apply for opportunities effortlessly.

2. **Enhance Learning & Skill Development**

To offer a centralized learning hub where institutions can upload courses (videos, PDFs, assignments) and provide skill-building programs.

3. **Simplify Job Applications**

To enable students and professionals to apply for jobs with one click, reducing application complexity.

4. **Improve Employer Recruitment**

To provide recruiters with a streamlined system to post jobs, track applications, and identify top candidates efficiently.

5. **Strengthen Networking & Community Engagement**

To create a professional community where experts can host webinars, mentorship sessions, and career guidance programs

6. **Provide Institutional Insights**

To equip educational institutions with data-driven analytics, helping them align their curricula with industry needs and improve student employability.

7. **Ensure Platform Security & Control**

To implement role-based authentication, allowing administrators to manage users, monitor activities, and handle complaints in real time.

8. **Digitize & Automate Processes**

To replace manual job search and career guidance methods with an automated, AI-driven approach that saves time for students and recruiters.

9. **Increase Accessibility & Career Opportunities**

To make career development resources accessible to students in remote locations, ensuring equal opportunities for all.

10. **Implement Real-Time Notifications & Tracking**

To keep users updated on job postings, course availability, and networking events through real-time notifications.

11. **Introduce Feedback & Rating Mechanisms**

To enable users to rate job postings, courses, and community interactions, ensuring continuous platform improvement.

12. **Create a Unified Career Ecosystem**

To connect students, institutions, employers, and professionals within a single platform, fostering collaboration and career growth.

EXISTING SYSTEM

The existing career development and job search system relies on manual processes, fragmented platforms, and limited accessibility. Students and job seekers must navigate multiple websites, career offices, and institutions to find job opportunities, internships, and skill-building resources. Institutions struggle to track student progress and placement outcomes, while employers face challenges in identifying the right talent.

Disadvantages of the Existing System:

- **Time-consuming** – Job seekers must visit multiple platforms to find opportunities, apply, and track applications.

- **Inefficient Career Guidance** – Students lack structured resources for career planning and professional networking.
- **Manual and Unstructured Process** – Institutions and recruiters rely on traditional methods like email communication and spreadsheets.
- **Limited Accessibility** – Many students, especially in rural areas, have limited access to career services.

PROPOSED SYSTEM

The proposed Career Hive platform overcomes these limitations by integrating job matching, course management, and networking features into a single digital ecosystem. It enables students, institutions, employers, and professionals to interact efficiently, reducing the reliance on traditional job search methods.

Advantages of the Proposed System:

- **Less Manual Effort** – AI-powered job recommendations and automated application tracking streamline the process.
- **Efficient System** – Centralized job postings, course enrollments, and networking opportunities improve accessibility.
- **Enhanced Institutional Support** – Schools and colleges can track student progress and align curricula with job market trends.
- **Time-Efficient** – Users receive real-time notifications about job postings, applications, and professional events.
- **Career Accessibility for All** – Web-based and mobile-ready, ensuring access for students in rural and urban areas alike.

FEASIBILTY STUDY

Before implementation, a feasibility study is conducted to assess the practicality, cost-effectiveness, and long-term sustainability of Career Hive. This ensures the platform is viable, beneficial, and scalable for all stakeholders.

Steps in Feasibility Analysis:

- **Formation of a Project Team** – Defining roles and responsibilities of developers, designers, and stakeholders.
- **Preparing the System Flowchart** – Mapping platform workflows, including job matching, course enrolment, and community features.
- **Enumerating Candidate Systems** – Evaluating alternative career service models and their limitations.

- **Identifying the Best Fit System** – Choosing Career Hive as the most effective, scalable, and user-friendly solution.

Economic Feasibility

- Reduces costs associated with manual career services and traditional job search methods.
- Increases efficiency and accuracy in job matching and skill-building resources.
- Offers potential revenue through premium features, employer partnerships, and institutional subscriptions.

Technical Feasibility

- Django-based web application ensures scalability and security.
- AI-powered recommendation system enhances job matching accuracy.
- Secure authentication mechanisms protect user data.
- Cloud-based hosting and database management improve performance and reliability.

Operational Feasibility

- User-friendly interface reduces training needs for students, institutions, and employers.
- Seamless onboarding ensures easy adoption by all stakeholders.
- Integrated feedback system enables continuous improvements.

System Design

System design is a critical phase that translates performance requirements into design specifications, outlining the solution and approach for the new system. It involves logical and physical stages, focusing on data structure, software architecture, and procedural details. The design process aims to create a representation of the software, providing understanding and procedure details necessary for implementation. In the design phase, database tables, input screens, and output reports are created, ensuring user-friendliness and avoiding redundancy. Effective design is essential for accurately translating system requirements into a software product.

Data flow diagrams (DFD)

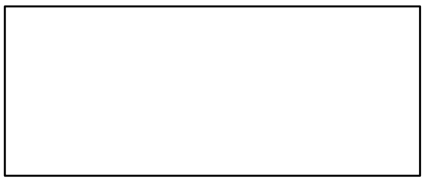
DFD is a network that describes the flow of data throughout the system. A dataflow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flow charts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually "say" things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to

CEO. That's why DFDs remain so popular after all these years. While they work well for data flow software and systems, they are less applicable nowadays to visualizing interactive, real time or database-oriented software or systems.

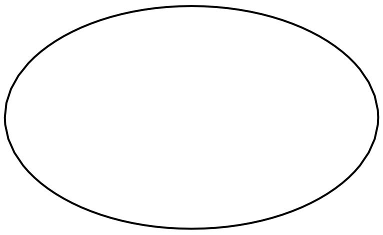
Using any convention's DFD rules or guidelines, the symbols depict the four components of data flow diagrams.

1. **External entity**: an outside system that sends or receives data, communicating with the system being diagrammed. They are the sources and destinations of information entering or leaving the system. They might be an outside organization or person, a computer system or a business system. They are also known as terminators, sources and sinks or actors. They are typically drawn on the edges of the diagram.
2. **Process**: any process that changes the data, producing an output. It might perform computations, or sort data based on logic, or direct the data flow based on business rules. A short label is used to describe the process, such as "Submit payment".
3. **Data store**: files or repositories that hold information for later use, such as a database table or a membership form. Each data store receives a simple label, such as "Orders."
4. **Data flow:** the route that data takes between the external entities, processes and datastores. It portrays the interface between the other components and is shown with arrows, typically labelled with a short data name, like "Billing details".

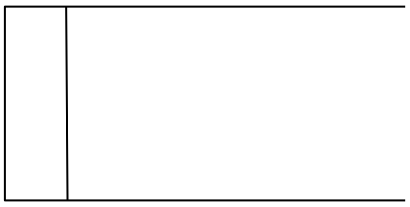
Some Data Flow Diagram Charting Forms:



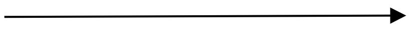
: External source (originator) or receiver



: Transform process



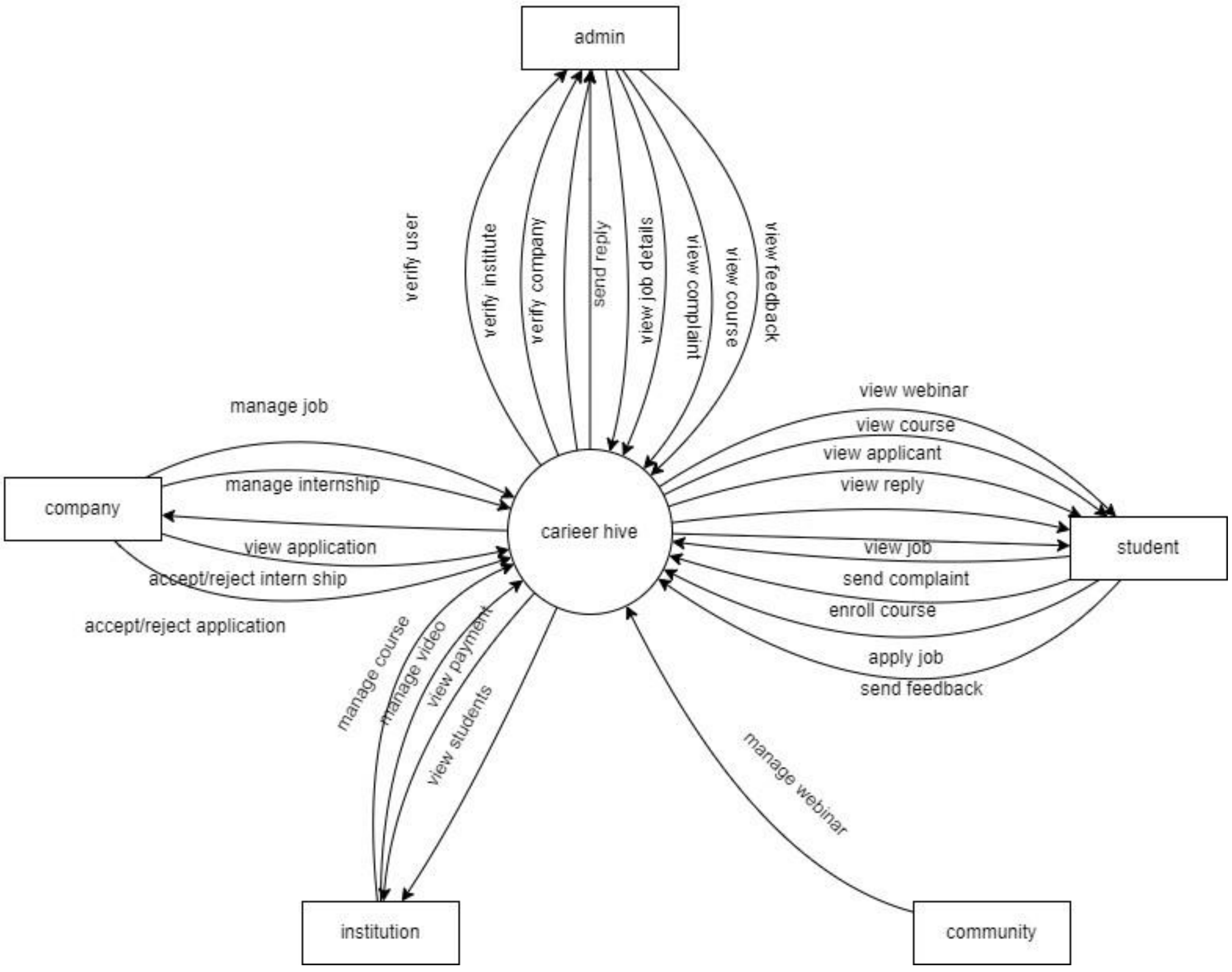
: Data Store

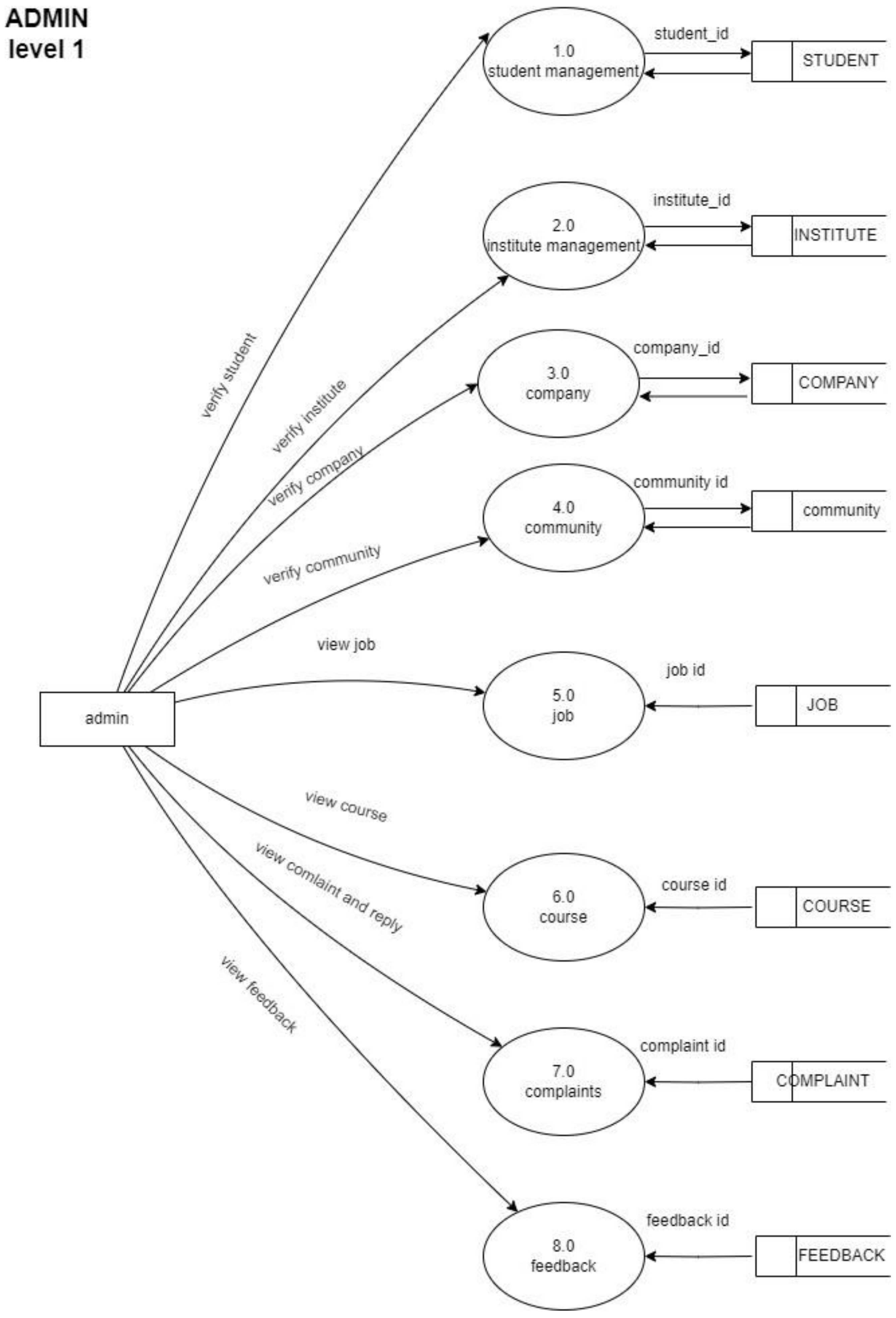


: Data Flow

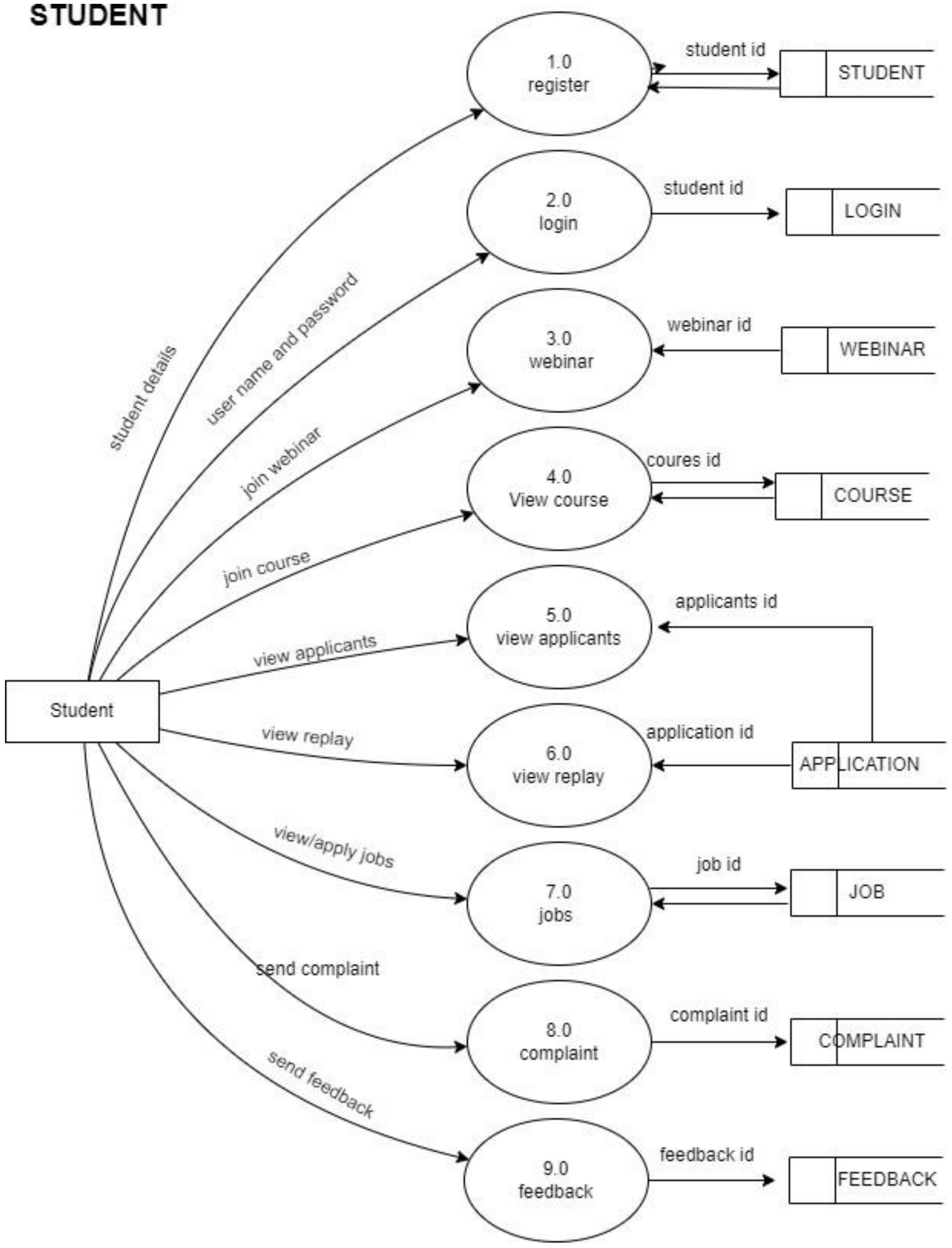
Data Flow Diagrams (DFD)

level 0



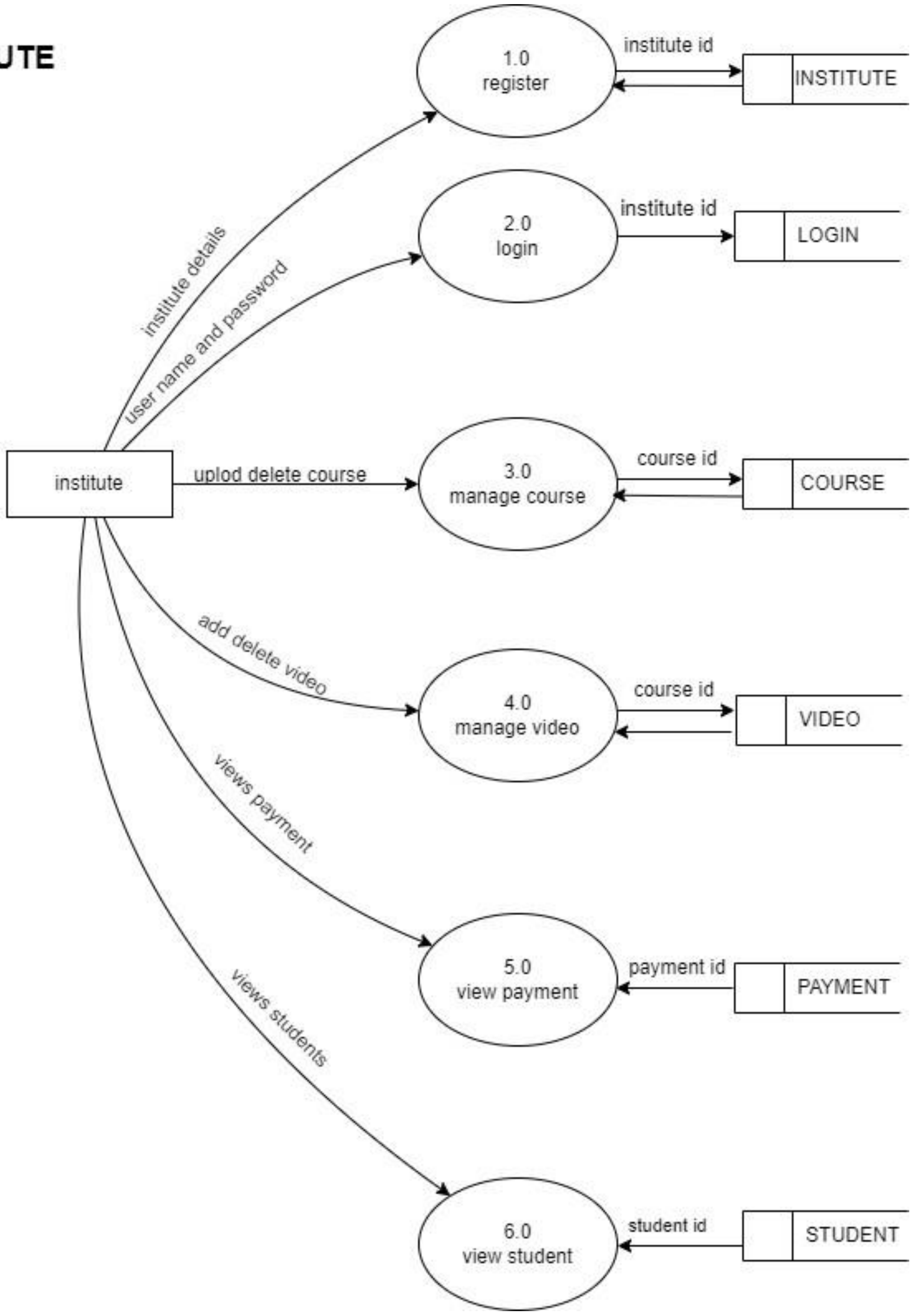

ADMIN
level 1
admin
verify student
verify institute
verify company
verify community
view job
view course
view comlaint and reply
view feedback
1.0
student management
2.0
institute management
3.0
company
4.0
community
5.0
job
6.0
course
7.0
complaints
8.0
feedback
student_id
institute_id
company_id
community id
job id
course id
complaint id
feedback id
STUDENT
INSTITUTE
COMPANY
community
JOB
COURSE
COMPLAINT
FEEDBACK

STUDENT

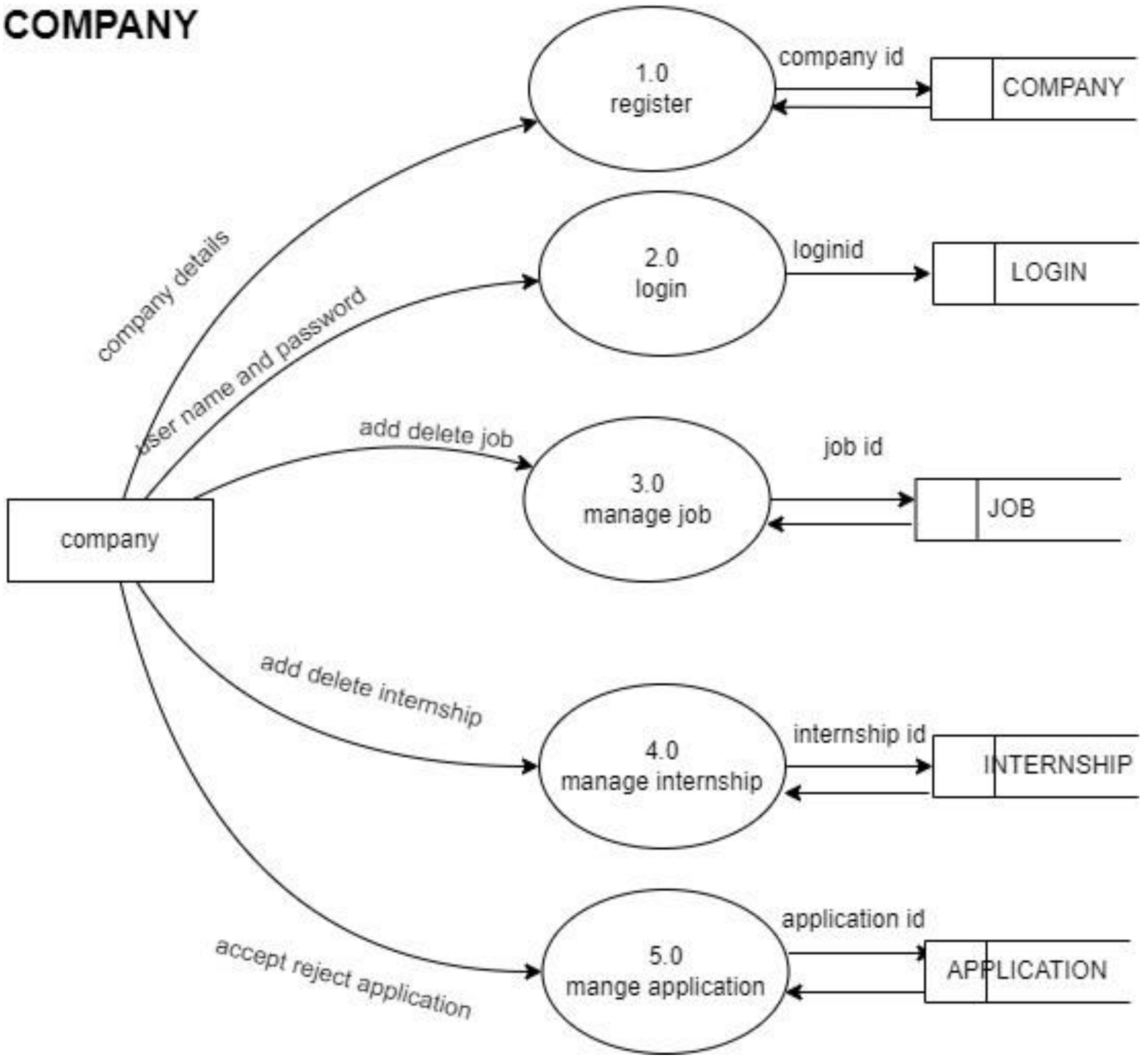


1.0
register
2.0
login
3.0
webinar
4.0
View course
5.0
view applicants
6.0
view replay
7.0
jobs
8.0
complaint
9.0
feedback
Student
student details
user name and password
join webinar
join course
view applicants
view replay
view/apply jobs
send complaint
send feedback
student id
STUDENT
student id
LOGIN
webinar id
WEBINAR
coures id
COURSE
applicants id
application id
APPLICATION
job id
JOB
complaint id
COMPLAINT
feedback id
FEEDBACK

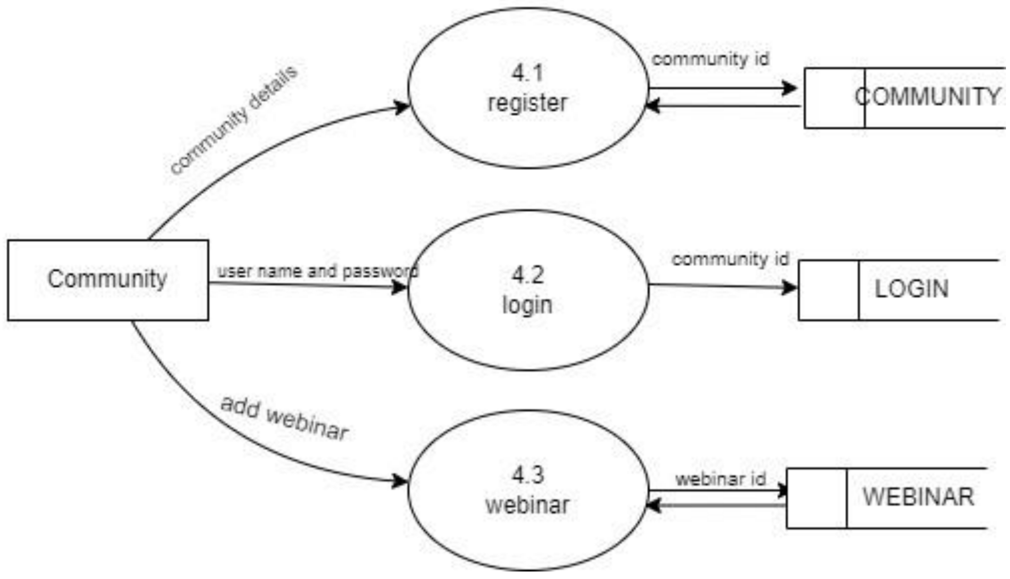
INSTITUTE


institute
institute details
user name and password
uplod delete course
add delete video
views payment
views students
1.0 register
2.0 login
3.0 manage course
4.0 manage video
5.0 view payment
6.0 view student
institute id
institute id
course id
course id
payment id
student id
INSTITUTE
LOGIN
COURSE
VIDEO
PAYMENT
STUDENT

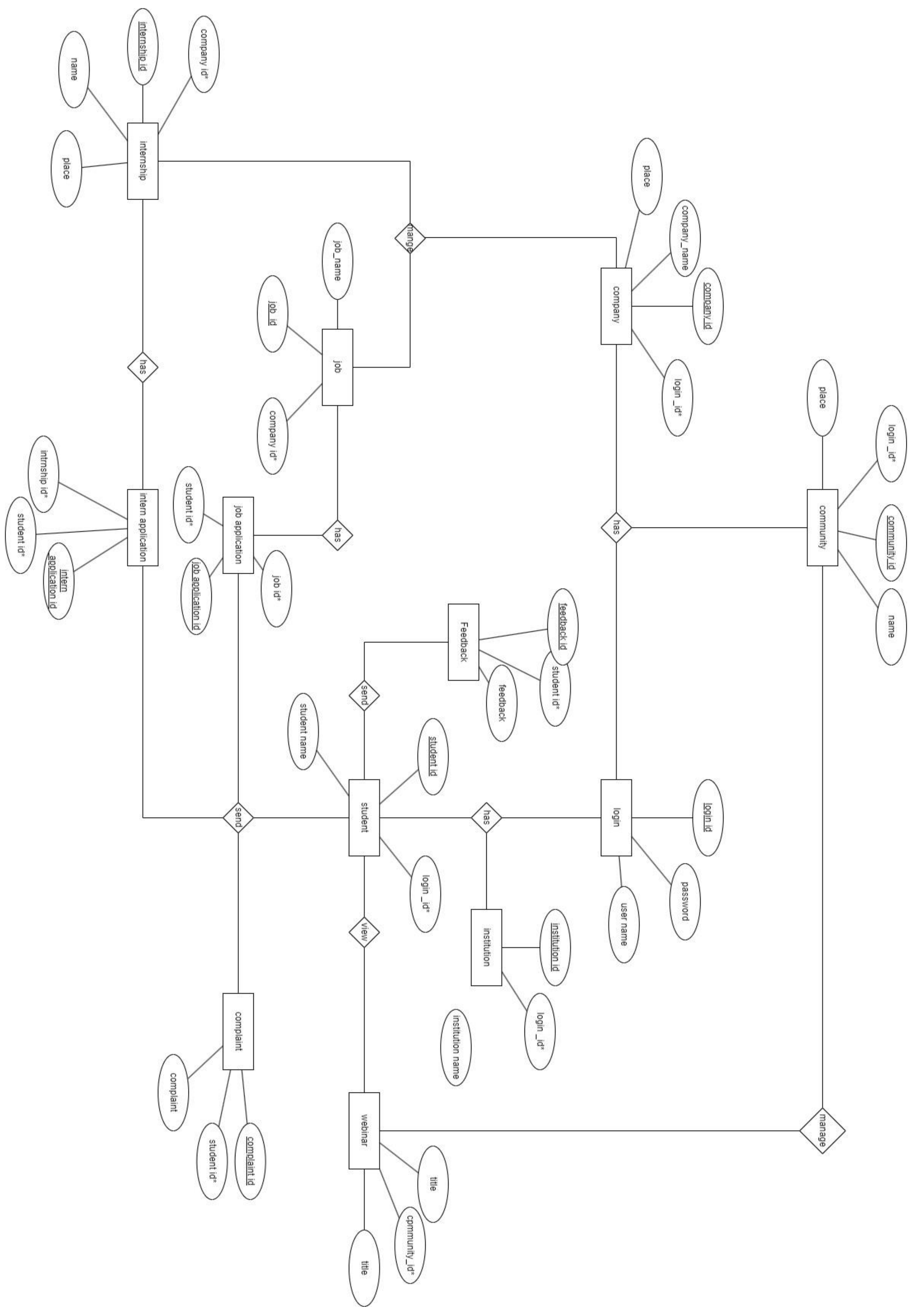
COMPANY


company details
user name and password
add delete job
add delete internship
accept reject application
company
1.0
register
2.0
login
3.0
manage job
4.0
manage internship
5.0
mange application
company id
loginid
job id
internship id
application id
COMPANY
LOGIN
JOB
INTERNSHIP
APPLICATION

COMMUNITY
level 1



ER DIAGRAM



internship
internship id
name
company id*
place
mange
company
place
company_name
company id
login _id*
job
job_name
job id
company id*
has
intern application
intrnship id*
student id*
intern application id
job application
student id*
job application id
job id*
has
community
place
login _id*
community id
name
has
Feedback
feedback id
student id*
feedback
send
student
student name
student id
login _id*
send
has
login
login id
password
user name
institution
institution id
login _id*
institution name
view
complaint
complaint
student id*
complaint id
webinar
title
cpmmunity_id*
title
manage

Chapter - 4

SYSTEM DESIGN

INTRODUCTION

System design is the process of defining the Architecture, modules, interfaces, and data for a system to satisfy specified requirements. System design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of system analysis, systems architecture and systems engineering.

TYPES OF SYSTEM DESIGN

- **Logical Design**: Logical design pertains to an abstract representation of the data flow, inputs and outputs of the system. It describes the inputs (source), outputs (destinations), databases (data stores), procedures (data flows) all in a format that meets the user requirements.
- **Physical Design**: Physical design relates to the actual input and outputs processes of the system. It focuses on how data is entered into a system, verified, processed, and displayed as output.
- **Architectural Design**: It is also known as high level design that focuses on the design of system architecture. It describes the structure and behavior of the system. It defines the structure and relationship between various modules of system development process.
- **Detailed Design**: It follows Architectural design and focuses on development of each module.
- **Conceptual Data Modelling**: It is representation of organization data which includes all the major entities and relationship. System analyses develop a conceptual data model for the current system that supports the scope and requirement for the proposed system.

DATABASE DESIGN

A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently. The general objective is to make information access easy, quick, inexpensive and flexible for users. The general theme behind a database is to integrate all information. Database design is recognized as a standard of management information system and is available virtually for every computer system. In database design several specific objectives are considered:

- Ease of learning and use
- Controlled redundancy
- Data independence
- More information at low cost
- Accuracy and integrity
- Recovery from failure

- Privacy and security

INPUT DESIGN

Input design is a part of the overall design. The input methods can be broadly classified. Internal controls must be established for monitoring the number of inputs and for ensuring that the data are valid. The basic steps involved in input design are:

- Review input requirements.
- Decide how the input data flow will be implemented.
- Decide the source document.
- Prototype online input screens.
- Design the input screens.

The quality of the system input determines the quality of the system output. Input specifications describe the way data enter the system for processing. Input design features can ensure the reliability of the system and produce result.

OUTPUT DESIGN

A quality is one, which meets the requirements of end user and present the information clearly. In any system results of processing are communicated to the user and to the other system through outputs. In the outputs design it is determined how the information is to be displayed for immediate need. It is the most important and direct source information is to user. Efficient and intelligent output design improves the system's relationships with the user and helps in decision making. The objectives of the output design are to convey the information of all the past activities, current status and to emphasis important events. The output generally refers to the results and information that is generated from the system. Outputs from computes are required primarily to communicate the result of processing to the users. Output also provides a means of storage by coping the results for later reference in consolation. There is a chance that some of the end users will not actually operate the input data or information through workstations but will see the output from the system.

Two phases of the output design are:

- Output Definitions
- Output Specification

Output definitions consider the type of outputs contents, its frequency and its volume, the appropriate outputs media is determined for output. Once the media is chosen, the details specification of output documents are carried out. The nature of output required from the proposed system is determined during logical design stage. It takes the outline of the output from the logical design and produces output as specified during the logical design phase. In a project, when designing the output, the system must accomplish the following:

- Determine the information to present.
- Decide whether to display, speak the information and select the output medium.
- Arrange the information in acceptable format.
- Decide how to distribute the output to the intended receipt.

Thus, by following the above specification, a high-quality output can be generated. Outputs from compute system are required primarily to communicate thee result of processing to users. Computer output is the most important and direct source of information to the user. Efficiency, intelligible output should improve the system's relationship with the user and help in decision making. The output devices to consider depend on factors as compatibility of the device with the system, response time requirements, expected print quality, number of copies needed etc.

NORMALIZATION

Designing a database is complete task and the normalization theory is a useful aid in the design process. The process of normalization is concerned with transformation of conceptual schema into computer representation form. There will be need for most databases to grow by adding new attributes and new relations. The data will be used new ways. Tuple will be added and deleted. Information stored may undergo updating also. New association may also be added. In such situation the performance of a database is entirely depend upon its design. A bad database design may lead to certain undesirable things like:

- Repetition of information
- Inability to represent certain information.
- Loss of information

To minimize these anomalies, normalization may be used. If the database is in a normalized from, the data can be growing without, in most cases, forcing the rewriting application programs. This is important because of the excessive and growing cost of maintaining an organization's application programs and its data from the disrupting effects of database growth. As the quality of application programs increase, the cost of maintaining the without normalization will rise to prohibitive levels, A normalized database can also encompass many related activities of an organization thereby minimizing the need for rewriting the applications of programs. Thus, normalization helps one attain a good database design and there by ensures continued efficiency of database. We can define the procedure as the successive reduction of a given collection of relations to some more desirable from. This procedure is reversible. That is, it is always possible to take the output from the procedure and convert them back into input. In this process, no information is lost. So, it is also called "no lose decomposition.

FIRST NORMAL FORM: A relation is first normal form (INF) if and all its attributes are based on single domain. The objective of normalization a table is to remove its repeating groups and ensure that all entries of the resulting table have at most single value.

SECOND NORMAL FORM: A table is said to be second normal form (2NF), when it is in INF and every attribute in the record is functionally dependent upon the whole key, and not just a part of the key.

THIRD NORMAL FORM: A table is in third normal form (3NF). When it is in 2NF and every non-key attribute is functionally dependent on just the primary key.

TABLE DESIGN

1. Login

Field	Type	Size	Constraints
id	Auto	-	Primary Key
username	CharField	255	Unique, Required
password	CharField	255	Required
usertype	CharField	50	Required

2. Student

Field	Type	Size	Constraints
id	Auto	-	Primary Key
LOGIN	ForeignKey	-	References Login, Required
first name	CharField	100	Required
last name	CharField	100	Required
place	CharField	100	Optional
pincode	CharField	10	Optional
phone	CharField	15	Optional
email	CharField	255	Unique, Required
image	ImageField	-	Optional

3. Company

Field	Type	Size	Constraints
id	Auto	-	Primary Key
LOGIN	ForeignKey	-	References Login, Required
company_name	CharField	150	Required
place	CharField	100	Optional
pincode	CharField	10	Optional
phone	CharField	15	Optional
email	CharField	255	Unique, Required
status	CharField	50	Optional (Default: 'Pending')

4. INSTITUTION

Field	Type	Size	Constraints
id	Auto	-	Primary Key
LOGIN	ForeignKey	-	References Login, Required
first_name	CharField	100	Required
email	CharField	255	Unique, Required
phone	CharField	15	Optional
place	CharField	100	Optional
pin	CharField	10	Optional
post	CharField	100	Optional
district	CharField	100	Optional
state	CharField	100	Optional

5. Community

Field	Type	Size	Constraints
id	Auto	-	Primary Key
LOGIN	ForeignKey	-	References Login, Required
name	CharField	150	Required
place	CharField	100	Optional
pincode	CharField	10	Optional
phone	CharField	15	Optional
email	CharField	255	Unique, Required

6. Courses

Field	Type	Size	Constraints
id	Auto	-	Primary Key
Instructor	ForeignKey	-	References Instructor, Required
course_name	CharField	150	Required
description	CharField	500	Optional
duration	CharField	50	Optional
cover_photo	CharField	255	Optional
fees	CharField	50	Optional
syllabus_pdf	CharField	255	Optional
status	CharField	20	Default: 'Pending'

7. Course_videos

Field	Type	Size	Constraints
id	Auto	-	Primary Key
COURSE	ForeignKey	-	References Courses, Required
video	CharField	255	Required
title	CharField	150	Required
description	CharField	500	Optional

8. Course_material

Field	Type	Size	Constraints
Id	Auto	-	Primary Key
COURSE	ForeignKey	-	References Courses, Required
course_materials	CharField	255	Required
Title	CharField	150	Required
Description	CharField	500	Optional

9. Payment

Field	Type	Size	Constraints
id	Auto	-	Primary Key
COURSE	ForeignKey	-	References Courses, Required
Student	ForeignKey	-	References Student, Required
amount	CharField	20	Required
date	DateField	-	Auto Now
status	CharField	20	Optional

10. Job

Field	Type	Size	Constraints
id	Auto	-	Primary Key
COMPANY	ForeignKey	-	References Company, Required
job_name	CharField	150	Required
place	CharField	100	Optional
description	CharField	500	Optional
skills_required	CharField	255	Optional
available	CharField	50	Optional

11. Internship

Field	Type	Size	Constraints
id	Auto	-	Primary Key
COMPANY	ForeignKey	-	References Company, Required
name	CharField	150	Required
place	CharField	100	Optional
description	CharField	500	Optional

12. Job Application

Field	Type	Size	Constraints
id	Auto	-	Primary Key
Job	ForeignKey	-	References Job, Required
STUDENT	ForeignKey	-	References Student, Required
status	CharField	50	Optional

13. Intern Application

Field	Type	Size	Constraints
id	Auto	-	Primary Key
INTERNSHIP	ForeignKey	-	References Internship, Required
STUDENT	ForeignKey	-	References Student, Required
status	CharField	50	Optional

15. Webinar

Field	Type	Size	Constraints
id	Auto	-	Primary Key
COMMUNITY	ForeignKey	-	References Community, Required
title	CharField	150	Required
description	CharField	500	Optional
duration	CharField	50	Optional
link	CharField	255	Optional

16. Event

Field	Type	Size	Constraints
id	Auto	-	Primary Key
COMMUNITY	ForeignKey	-	References Community, Required
title	CharField	150	Required
description	CharField	500	Optional
date	CharField	20	Optional
time	CharField	20	Optional

17. Feedback

Field	**Type**	**Size**	**Constraints**
id	Auto	-	Primary Key
STUDENT	ForeignKey	-	References Student, Required
feedback	CharField	500	Optional
date	DateField	-	Auto Now

18. Complaint

Field	**Type**	**Size**	**Constraints**
id	Auto	-	Primary Key
STUDENT	ForeignKey	-	References Student, Required
complaint	CharField	500	Optional
date	DateField	-	Auto Now
reply	CharField	500	Optional

PROCESS DESIGN

Process design outlines the structure of data and program components for building a computerbased system. It considers the architectural style, component structure and properties, and interrelationships among components.Specialists like database designers and system architects often handle this task for large, complex systems. The system architect selects an appropriate architectural style based on system requirements.Architectural design involves creating data architecture, deriving architectural structure representations, and developing an architecture model. This model includes data architecture, program structure, component properties, and technical specifications for implementation.The design phase takes software requirements as input and produces design specifications as output. System design also involves creating layouts for input and reports for output.

STRUCTURED DESIGN

Structured design deals with the data-flow in the system. It partitions a program into hierarchy of modules. The modules are organized in a top-down manner and the details will be at the bottom. The structured Design begins with a system specification that identifies inputs and outputs that described the functional of the Table

INPUT DESIGN

Input design is a part of the overall design. The input methods can be broadly classified. Internal controls must be established for monitoring the number of inputs and for ensuring that the data are valid. The basic steps involved in input design are:

- Review input requirements
- Decide how the input data flow will be implemented.
- Decide the source document.
- Prototype online input screens.
- Design the input screens.

The quality of the system input determines the quality of the system output. Input specifications describe the way data enter the system for processing. Input design features can ensure the reliability of the system od produce result.

OUTPUT DESIGN

Process design outlines the structure of data and program components for building a computerbased system. It considers the architectural style, component structure and properties, and interrelationships among components.Specialists like database designers and system architects often handle this task for large, complex systems. The system architect selects an appropriate architectural style based on system requirements.Architectural design involves creating data architecture, deriving architectural structure representations, and developing an architecture model. This model includes data architecture, program structure, component properties, and technical specifications for implementation.The design phase takes software requirements as input and produces design specifications as output. System design also involves creating layouts for input and reports for output.

Chapter – 5

CODING

```
"""career_guidance_project URL Configuration


The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/3.2/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import include, path
from . import views


urlpatterns = [
    path('',views.public_home),
    path('login/',views.login),
    path('manage_instructor',views.manage_instructor),
    path('manage_company',views.manage_company),
    path('manage_community',views.manage_community),
    path('manage_students',views.manage_students),


    path('admin_home',views.admin_home),
    path('admin_manage_staff',views.admin_manage_staff),
    path('staff_del/<id>',views.staff_del),
    path('staff_upd/<id>',views.staff_upd),
    path('admin_view_job_details',views.admin_view_job_details),
    path('view_jobs/<id>',views.view_jobs),
    path('admin_manage_company',views.admin_manage_company),
    path('cmpny_del/<id>',views.cmpny_del),
    path('cmpny_upd/<id>',views.cmpny_upd),
    path('admin_view_course_details',views.admin_view_course_details),
    path('admin_view_cmplt',views.admin_view_cmplt),
    path('admin_send_reply/<id>',views.admin_send_reply),
    path('admin_view_feedback',views.admin_view_feedback),
    path('admin_view_rating',views.admin_view_rating),
    path('admin_view_user',views.admin_view_user),
    path('admin_view_instructor',views.admin_view_instructor),
    path('admin_view_company',views.admin_view_company),
    path('admin_view_community',views.admin_view_community),
    path('admin_approve_company/<id>',views.admin_approve_company),
    path('admin_reject_company/<id>',views.admin_reject_company),
    path('admin_approve_instructor/<id>',views.admin_approve_instructor),
    path('admin_reject_instructor/<id>',views.admin_reject_instructor),
    path('admin_approve_community/<id>',views.admin_approve_community),
    path('admin_reject_community/<id>',views.admin_reject_community),
    path('admin_approve_user/<id>',views.admin_approve_user),
    path('admin_reject_user/<id>',views.admin_reject_user),
    path('community_home',views.community_home),
    path('community_manage_webinar',views.community_manage_webinar),
    path('community_manage_event',views.community_manage_event),
    path('edit_webinar/<id>/',views.edit_webinar),
    path('delete_webinar/<id>/',views.delete_webinar),
    path('edit_event/<id>/',views.edit_event),
    path('delete_event/<id>/',views.delete_event),
    path('access_videos/<id>/',views.access_videos),
    path('view_assigments/<id>/',views.view_assigments),
    path('applyjob/<id>/',views.applyjob),
    path('view_job_application',views.view_job_application),
        path('staff_home',views.staff_home),

        path('staff_manage_clg',views.staff_manage_clg),

        path('college_del/<id>',views.college_del),

        path('college_upd/<id>',views.college_upd),

        path('staff_mng_crs/<id>',views.staff_mng_crs),

        path('course_upd/<id1>/<id>',views.course_upd),

        path('course_del/<id1>/<id>',views.course_del),

        path('staff_view_enquiry',views.staff_view_enquiry),

        path('staff_send_reply/<id>',views.staff_send_reply),

        path('staff_view_rating',views.staff_view_rating),

        path('staff_view_feedback',views.staff_view_feedback),

        path('staff_send_noti',views.staff_send_noti),

        path('update_noti/<id>',views.update_noti),

        path('delete_noti/<id>',views.delete_noti),

        path('instructor_home',views.instructor_home,name='instructor_home'),

        path('instructor_manage_courses',views.instructor_manage_courses),

        path('edit_course/<int:id>', views.edit_course, name='edit_course'),

        path('delete_course/<int:id>', views.delete_course, name='delete_course'),

        path('instructor_manage_quizzes/', views.instructor_manage_quizzes,
    name='instructor_manage_quizzes'),

        path('edit_quiz/<int:id>/', views.edit_quiz, name='edit_quiz'),

        path('delete_quiz/<int:id>/', views.delete_quiz, name='delete_quiz'),

        path('instructor_manage_materials/<id>/', views.instructor_manage_materials,
    name='instructor_manage_materials'),

        path('edit_material/<int:id>/', views.edit_material, name='edit_material'),

        path('delete_material/<int:id>/', views.delete_material, name='delete_material'),

        path('instructor_manage_videos/<id>/', views.instructor_manage_videos,
    name='instructor_manage_videos'),

        path('instructor_manage_assignments/<id>/', views.instructor_manage_assignments,
    name='instructor_manage_assignments'),

        path('edit_video/<int:id>', views.edit_video, name='edit_video'),

        path('delete_video/<int:id>', views.delete_video, name='delete_video'),

        path('cmpny_home',views.cmpny_home),

        path('company_view_user',views.company_view_user),

        path('manage_internship',views.manage_internship),
    path('cmp_mng_job',views.cmp_mng_job),
    path('job_upd/<id>',views.job_upd),
    path('job_del/<id>',views.job_del),
    path('delete_student/<id>',views.delete_student),
    path('inter_del/<id>',views.inter_del),
    path('inter_upd/<id>',views.inter_upd),
    path('cmp_view_job',views.cmp_view_job),
    path('cmp_mng_vac/<id>',views.cmp_mng_vac),
    path('vac_upd/<id1>/<id>',views.vac_upd),
    path('vac_del/<id1>/<id>',views.vac_del),
    path('cmpny_view_application',views.cmpny_view_application),
    path('short_list/<id>',views.short_list),
    path('reject_req/<id>',views.reject_req),
    path('user_login',views.user_login),
    path('std_view_college',views.std_view_college),
    path('student_view_courses',views.student_view_courses),
    path('std_view_cmpny',views.std_view_cmpny),
    path('std_view_vacancy',views.std_view_vacancy),
    path('std_send_application',views.std_send_application),
    path('view_notification',views.view_notification),
    path('std_enquiry',views.std_enquiry),
    path('view_enq_rply',views.view_enq_rply),
    path('std_send_rating',views.std_send_rating),
    path('std_send_feedback',views.std_send_feedback),
    path('std_send_complaints',views.std_send_complaints),
    path('view_reply',views.view_reply),
    path('std_reg',views.std_reg),
    path('fetch_course',views.fetch_course),
    path('user_viewprofile',views.user_viewprofile),
    path('user_view_internships',views.user_view_internships),
    path('user_view_jobs',views.user_view_jobs),
    # path("upldcv", views.upldcv),
    # path("careerprediction", views.careerprediction),
    # path("d", views.d),
    path('companyviewappintern/',views.companyviewappintern),
    path('instructor_view_students/',views.instructor_view_students),
    path('companyaccintern/<id>/',views.companyaccintern),
    path('companyrejectintern/<id>/',views.companyrejectintern),
    path('job_accpet/<id>/',views.job_accpet),
    path('job_reject/<id>/',views.job_reject),
    path('user_home/',views.user_home),
    path('user_viewcourses/',views.user_viewcourses),
    path('user_view_cousre_details/<id>/',views.user_view_cousre_details),
    path('user_viewjobs/',views.user_viewjobs),
    path('user_viewinternships/',views.user_viewinternships),
    path('user_profile/',views.user_profile),
    path('make_payment/<int:id>/', views.make_payment, name='make_payment'),
    path('admin_approve_payment/<int:payment_id>/', views.admin_approve_payment,
name='admin_approve_payment'),
    path('view-course/<int:id>/', views.view_course, name='view_course'),
    # path('attendquiz/<id>',views.attendquiz),
    path('attendquiz/<int:course_id>/', views.attend_quiz, name='attend_quiz'),
    path('submit_quiz/', views.submit_quiz, name='submit_quiz'),
    path('apply/<int:vacancy_id>/', views.apply_job, name='apply_job'),
    path('applyintern/<id>',views.applyintern),
    path('viewappintern/',views.viewappintern),
    path('create_assignment/',views.create_assignment),
    path('assignment_list/',views.assignment_list),
    path('user_sent_complaint/',views.user_sent_complaint),
    path('user_sent_feedback/',views.user_sent_feedback),
    path("webinar_list/", views.webinar_list, name="webinar_list"),
{% include 'learning/header.html'%}
<div class="container">
    <div class="row">
        <div class="col-lg-12">
            <center>
                <h1 class="opacity">Manage Assigments</h1>
                <form action="" method="post" class="register" enctype="multipart/form-data">
```

```
                    <input type="text" class="input-field" name="title" placeholder="Title" required /> <br><br>
                    <input type="text" class="input-field" name="description" placeholder="Questions" required />
<br><br>
                    <label for="">Last Date</label>
                    <input type="date" class="input-field" name="end_date" placeholder="Questions" required />
<br><br>


                    <input type="submit" class="btn btn-secondary" value="Add Assigments" name="submit"
class="opacity">
                </form>
            </center>
        </div>
    </div>
</div>


<center>
    <br><br><br><br>
    <div class="tables">
        <h1>Assigments</h1>
        <table class="table">
            <thead>
                <tr>
                    <th>ID</th>
                    <th>Course</th>
                    <th>Title</th>
                    <th>Questions</th>
                    <th>Last Date</th>
                    <th>Actions</th>
                </tr>
            </thead>
            <tbody>
                {% for video in videos %}
                <tr>
                    <td>{{ video.id }}</td>
                    <td>{{ video.COURSE.course_name }}</td>
                    <td>{{ video.title }}</td>
                    <td>{{ video.description }}</td>
                    <td>{{ video.end_date }}</td>

                    <td>
                        <center>
                        <a href="/edit_video/{{video.id}}" class="btn btn-success">Edit</a>
                        <br>
                        <a href="/delete_video/{{video.id}}" class="btn btn-danger">Delete</a>
                        </center>
                    </td>
                </tr>
                {% endfor %}
            </tbody>
        </table>
    </div>
</center>

{% include 'footer.html' %}
{% include 'learning/header.html'%}

<!-- Page Container -->
<div class="container my-5">
    <div class="row">
        <div class="col-lg-12">
            <div class="text-center mb-4">
                <h1 class="display-4 text-primary">Manage Courses</h1>
                <form action="" method="post" class="register shadow p-4 rounded" enctype="multipart/form-
data">
                    {% csrf_token %}
                    <div class="form-group">
                        <input type="text" class="form-control mb-3" name="course_name" placeholder="Course
Name" required />
```

```
                    </div>
                    <div class="form-group">
                        <input type="text" class="form-control mb-3" name="description"
placeholder="Description" required />
                    </div>
                    <div class="form-group">
                        <input type="text" class="form-control mb-3" name="duration" placeholder="Duration"
required />
                    </div>
                    <div class="form-group">
                        <input type="number" class="form-control mb-3" name="fees" placeholder="Fees" required
/>
                    </div>
                    <div class="form-group">
                        <label for="syllabus_pdf">Choose PDF</label>
                        <input type="file" class="form-control mb-3" name="syllabus_pdf" required />
                    </div>
                    <div class="form-group">
                        <label for="cover_photo">Choose Cover Photo</label>
                        <input type="file" class="form-control mb-3" name="cover_photo" required />
                    </div>
                    <div class="text-center">
                        <input type="submit" class="btn btn-primary btn-lg" value="Add Course" name="submit">
                    </div>
                </form>
            </div>
        </div>
    </div>

    <!-- Courses Table -->
    <div class="tables">
        <h2 class="text-center text-primary my-5">Courses</h2>
        <div class="table-responsive">
            <table class="table table-bordered table-striped">
                <thead class="thead-dark">
                    <tr>
                        <th>ID</th>
                        <th>Course Name</th>
                        <th>Description</th>
                        <th>Duration</th>
                        <th>Fees</th>
                        <th>Syllabus</th>
                        <th>Cover Photo</th>
                        <th>Actions</th>
                        <th>Works</th>
                    </tr>
                </thead>
                <tbody>
                    {% for course in courses %}
                    <tr>
                        <td>{{ course.id }}</td>
                        <td>{{ course.course_name }}</td>
                        <td>{{ course.description }}</td>
                        <td>{{ course.duration }}</td>
                        <td>{{ course.fees }}</td>
                        <td><a href="/static/media/{{ course.syllabus_pdf }}" class="btn btn-info btn-sm"
target="_blank">View Syllabus</a></td>
                        <td><img src="/static/media/{{ course.cover_photo }}" alt="{{ course.course_name }}
Cover Photo" width="100" class="img-thumbnail"></td>
                        <td>
                            <div class="text-center">
                                <a href="/edit_course/{{ course.id }}" class="btn btn-success btn-sm">Edit</a>
                                <br><br>
                                <a href="/delete_course/{{ course.id }}" class="btn btn-danger btn-sm">Delete</a>
                            </div>
                        </td>
                        <td>
                            <div class="text-center">
                                <a href="/instructor_manage_videos/{{ course.id }}" class="btn btn-info btn-sm">Add
Videos</a>
```

```
                        <br><br>
                        <!-- <a href="/instructor_manage_assignments/{{ course.id }}" class="btn btn-warning
btn-sm">Add Assignments</a>
                        <br><br> -->
                        <a href="/admin_approve_payment/{{ course.id }}" class="btn btn-success btn-
sm">Payments</a>
                    </div>
                </td>
            </tr>
            {% endfor %}
          </tbody>
        </table>
      </div>
    </div>
</div>

{% include 'footer.html' %}
{% include 'community/header.html' %}

<style>
    /* General Layout Styling */
    .container {
        max-width: 1200px;
        margin: 0 auto;
        padding: 20px;
    }

    h1, h3 {
        text-align:  center;
        font-size:  2.5rem;
        margin-top: 20px;
    }

    .opacity {
```

```
    opacity: 0.9;
}

/* Form Styling */
.register {
    max-width: 700px;
    margin: 0 auto;
    padding: 20px;
    background-color: #f8f8f8;
    border-radius: 10px;
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);
}

.input-field {
    width: 100%;
    padding: 12px;
    margin-bottom: 12px;
    border: 1px solid #ccc;
    border-radius: 8px;
    background-color: #fff;
    font-size: 16px;
    color: #333;
}

.btn-secondary {
    width: 100%;
    padding: 15px;
    background-color: #573b8a;
    color: white;
    border: none;
    border-radius: 8px;
    font-size: 18px;
    cursor: pointer;
    transition: background-color 0.3s ease;
    }

    .btn-secondary:hover {
        background-color: #6d44b8;
    }

    /* Responsive Styling */
    @media (max-width: 767px) {
        .register {
            padding: 15px;
        }

        .input-field, .btn-secondary {
            font-size: 14px;
        }
    }
</style>

<main>
    <div class="container">
        <div class="row">
            <!-- Hero Text Section -->
            <div class="col-lg-12">
                <h1 class="opacity">Manage Events</h1>
                <form action="" method="post" class="register" enctype="multipart/form-data">
                    {% csrf_token %}
                    <input type="text" class="input-field" name="title" placeholder="Event Title" required
/><br><br>
                    <input type="date" class="input-field" name="date" required /><br><br>
                    <input type="time" class="input-field" name="time" required /><br><br>
                    <textarea class="input-field" name="description" placeholder="Event Description" rows="4"
required></textarea><br><br>

                    <input type="submit" class="btn-secondary" value="Add Event" name="submit" />
            </form>
        </div>
    </div>
</div>

<div class="container">
    <h3 class="opacity">Existing Events</h3>
    <table class="table table-bordered table-striped" align="center">
        <thead>
            <tr>
                <th>Event Title</th>
                <th>Date</th>
                <th>Time</th>
                <th>Description</th>
                <th>Actions</th>
            </tr>
        </thead>
        <tbody>
            {% for event in courses %}
                <tr>
                    <td>{{ event.title }}</td>
                    <td>{{ event.date }}</td>
                    <td>{{ event.time }}</td>
                    <td>{{ event.description }}</td>
                    <td>
                        <center>
                        <a href="../edit_event/{{event.id}}" class="btn btn-success">Edit</a>
                        <br><br>
                        <a href="../delete_event/{{event.id}}" class="btn btn-danger">Delete</a>
                        </center>
                    </td>
                </tr>
            {% endfor %}
        </tbody>
        </table>
    </div>
</main>

{% include 'footer.html' %}
{% include 'community/header.html' %}

<style>
    /* General Layout Styling */
    .container {
        max-width: 1200px;
        margin: 0 auto;
        padding: 20px;
    }

    h1, h3 {
        text-align:  center;
        font-size:  2.5rem;
        margin-top: 20px;
    }

    .opacity {
        opacity: 0.9;
    }

    /* Form Styling */
    .register {
        max-width: 700px;
        margin: 0 auto;
        padding: 20px;
        background-color: #f8f8f8;
        border-radius: 10px;
        box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);
    }
```

```
.input-field {
    width: 100%;
    padding: 12px;
    margin-bottom: 12px;
    border: 1px solid #ccc;
    border-radius: 8px;
    background-color: #fff;
    font-size: 16px;
    color: #333;
}

.btn-secondary {
    width: 100%;
    padding: 15px;
    background-color: #573b8a;
    color: white;
    border: none;
    border-radius: 8px;
    font-size: 18px;
    cursor: pointer;
    transition: background-color 0.3s ease;
}

.btn-secondary:hover {
    background-color: #6d44b8;
}

/* Table Styling */
.tables {
    margin-top: 50px;
    width: 100%;
    overflow-x: auto;
}
```

```
    .tables table {
        width: 100%;
        border-collapse: collapse;
    }

    .tables table th,
    .tables table td {
        padding: 12px;
        text-align: center;
        border: 1px solid #ddd;
    }

    .tables table th {
        background-color: #f1f1f1;
        font-weight: bold;
    }

    .tables table td img {
        width: 100px;
        height: auto;
    }

    /* Responsive Styling */
    @media (max-width: 767px) {
        .register {
            padding: 15px;
        }

        .input-field, .btn-secondary {
            font-size: 14px;
        }
    }
</style>
```

```
<main>
    <div class="container">
        <div class="row">
            <!-- Hero Text Section -->
            <div class="col-lg-12">
                <h1 class="opacity">Manage Webinars/Events</h1>
                <form action="" method="post" class="register" enctype="multipart/form-data">
                    {% csrf_token %}
                    <input type="text" class="input-field" name="title" placeholder="Webinar Title" required
/><br><br>
                    <input type="text" class="input-field" name="duration" placeholder="Duration" required
/><br><br>
                    <input type="text" class="input-field" name="link" placeholder="Link" required /><br><br>
                    <textarea class="input-field" name="description" placeholder="Description" rows="4"
required></textarea><br><br>
                    <input type="submit" class="btn-secondary" value="Add Webinar" name="submit" />
                </form>
            </div>
        </div>

        <div class="tables">
            <h1>Existing Webinars/Events</h1>
            <table class="table">
                <thead>
                    <tr>
                        <th>Title</th>
                        <th>Duration</th>

                        <th>Link</th>
                        <th>Description</th>
                        <th>Actions</th>
                    </tr>
                </thead>
                <tbody>
```

```
                    {% for course in courses %}
                        <tr>
                            <td>{{ course.title }}</td>
                            <td>{{ course.duration }}</td>
                            <td><a href="{{ course.link }}">{{ course.link }}</a></td>
                            <td>{{ course.description }}</td>
                            <td>
                                <center>
                                <a href="../edit_webinar/{{course.id}}" class="btn btn-success">Edit</a>
                                <br><br>
                                <a href="../delete_webinar/{{course.id}}" class="btn btn-danger">Delete</a>
                                </center>
                            </td>
                        </tr>
                    {% endfor %}
                </tbody>
            </table>
        </div>
    </div>
</main>

{% include 'footer.html' %}
 <!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>My Assignments</title>
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css">
</head>
<body class="container mt-5">
    <h2>My Assignments</h2>

    <table class="table table-bordered">
```

```
        <thead>
            <tr>
                <th>Title</th>
                <th>Description</th>
                <th>Due Date</th>

            </tr>
        </thead>
        <tbody>
            {% for assignment in assignments %}
            <tr>
                <td>{{ assignment.title }}</td>
                <td>{{ assignment.description }}</td>
                <td>{{ assignment.date }}</td>

            </tr>
            {% endfor %}
        </tbody>
    </table>

    <a href="/create_assignment" class="btn btn-primary">Create New Assignment</a>
</body>
</html>

{% include 'staff_pages/staff_header.html'%}
<br><br><br><br>

<style>
    form {
        width: 60%;
        margin: 50px auto;

        padding: 20px;
        border-radius: 10px;
        box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
    }

    table {
        width: 100%;
        margin-top: 20px;
    }

    th, td {
        padding: 12px;
        text-align: left;
    }

    th {
        background-color: #573b8a;
        color: #fff;
    }

    textarea {
        width: 100%;
        padding: 8px;
        border: 1px solid #ccc;
        border-radius: 4px;
        box-sizing: border-box;
        margin-top: 8px;
        margin-bottom: 16px;

    }

    input[type="submit"] {
        padding: 10px 15px;
        background-color: #573b8a;
        color: #fff;
        border: none;
```

```
        border-radius: 5px;
        cursor: pointer;
    }

    input[type="submit"]:hover {
        background-color: #452d6a;
    }
</style>

<form action="" method="post">
    {% csrf_token %}
<table align="center">
    <tr>
        <th style="width: 30%;">reply</th>
        <td><textarea name="rply" id="" cols="" rows="10"></textarea></td>
    </tr>
    <tr>
   <td align="center" colspan="2"><input type="submit" name="submit" id="" value="Reply" class="form
form-control" class="btn btn-info"></td>
    </tr>
</table>
</form>

<br><br><br><br>

{% include 'footer.html'%}
```

Chapter – 6

TESTING

Testing is the major quality measure employed during software development. After the coding phase, computer programs are available that can be executed for testing purposes. Testing not only has to recover errors introduced during coding, but also locates errors committed during the previous phases. Thus the aim of testing is to verify the requirement design or coding error in the program. System testing is an expensive but critical process that can take as much as fifty percent of the budget for program development. Consequential different level of testing are employed in fact a successful test is one that finds an error. The system performance criteria deals with turnaround time, backup, file protection and human factor. A test for the user acceptance should be carried out. The package developed was taken through different levels of testing and required modification was made.

TESTING TECHNIQUES AND STRATEGIES USED

System testing is the state of implementation which is aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is vital to the success of the system. The candidate system is subjected to a variety of tests. A series of tests are performed for the proposed system before the system is ready for user acceptance testing.

Testing is divided into several distinct operations:

- Unit testing
- Integration testing
- Validation testing
- Output testing
- Test data output
- User Acceptance testing

UNIT TESTING

Unit testing is a level of a software testing where individual units/components of software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc. In object- oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. (Some treat a module of an application as a unit. This is to be discouraged as there will probably be many individual units within that module.) Unit testing frameworks, drivers, stubs, and mock/ fake objects are used to assist in unit testing.

INTEGRATION TESTING

Integration testing is a level of software testing where individual units are combined and tested as a group. The purpose of the level of testing is to expose faults in the interaction between integrated units. The purpose of this level of testing is to expose faults in the intersection between integrated units. The drivers and test stubs are used to assist in Integration Testing.

OUTPUT TESTING

Here the output is tested to view whether that screen is what which is desired. It is also checked whether it is to the satisfaction of the user.

TEST DATA OUTPUT

After repairing test data the system under study is tested using the test data. While testing the system using the test data, errors are again uncovered and corrected by using about testing and corrections are also noted for future use.

USER ACCEPTANCE TESTING

User acceptance testing is done in presence of user. The user will have some constraints and if some constraints are absent it has to be added and then the user is satisfied.

VALIDATION CHECKS

The "CARRIER HIVE" project requires robust validation checks to ensure accuracy, security, and reliability. These include validating user registration details (email, password, phone number), service requests (location, vehicle details, emergency status), and payment transactions. Service provider profiles, availability, and feedback must also be verified. The system ensures real-time tracking and notifications are accurate, while the user assistance module validates problems and solutions. Emergency services, data security, and session management are rigorously checked to prevent vulnerabilities. Additionally, feedback, complaints, and system performance are validated to maintain user satisfaction and efficiency. These checks ensure a seamless, secure, and user-friendly experience for both vehicle owners and service providers.

IMPLEMENTATION

Implementation includes placing the system into operation and providing the users and operation personnel with the necessary documentation to use and maintain the new system. Implementation includes all those activities that take place to convert from the old system. Proper implementation is essential to provide are liable system to meet the organizational requirements. Successful implementation may not guarantee improvement in the organization using the new system, as well as improper installation will prevent. There are four methods,

- Parallel approach: The old system is operated with the new system.
- Direct cut over method: The old system is replaced with the new system.
- Pilot approach: Working version of the system is implemented in one part of the organization based on the feedback, changes are made, and the system is installed in the rest of the organization by one the other methods.
- Phase-in-method: Gradually implements the system across all users.

SECURITY

- Integrity
- Privacy
- Disaster is known as system security.
- System security can be divided into four related issues.
- Confidentiality
- Security

Data security is the protection of data from loss, disclosure, modification and destruction. System Integrity refers to the power functioning of hardware and programs, appropriate physical security, and safety against external threats such as eavesdropping and writes tapping.

Chapter -7

SYSTEM MAINTANENCE

Maintenance means restoring something to its original conditions. Enhancement means adding, modifying the code to support the changes in the user specification System maintenance conforms the system to its original requirements and enhancement adds to system capability by incorporating new requirements Thus, maintenance changes the existing system, enhancement adds features to the existing system, and development replaces the existing system. It is an important part of system development that includes the activities which corrects errors in system design and implementation, updates the documents, and tests the data

Maintenance Types:

System maintenance can be classified into four types:

- Corrective Maintenance
- Adaptive Maintenance
- Perfective Maintenance
- Preventive Maintenance

Corrective Maintenance

Corrective Maintenance deals with the repair of faults or defects found in day- today system functions. A defect can result due to errors in software design, logic and coding. Design errors occur when changes made to the software are incorrect, incomplete, wrongly communicated, or the change request is misunderstood. Logical errors result from invalid tests and conclusions, incorrect implementation of design specifications, faulty logic flow, or incomplete implementation of design specifications, faulty logic flow, or incomplete test of data. All these errors, referred to as residual errors, prevent the software from confirming to its agreed specifications. Note that the need for corrective maintenance is usually initiated by big reports drawn by the users.

Adaptive Maintenance

Adaptive Maintenance is the implementation of changes in a part of the system, which has been affected by a change that occurred in some other part of the system. Adaptive Maintenance consists of adapting software to changes in the environment such as the hardware or the operating system. The term environment in this context refers to the conditions and the influences which act (from outside) on the system. For example, business rules, work patterns and government policies have a significant impact on the software system.

Perfective Maintenance

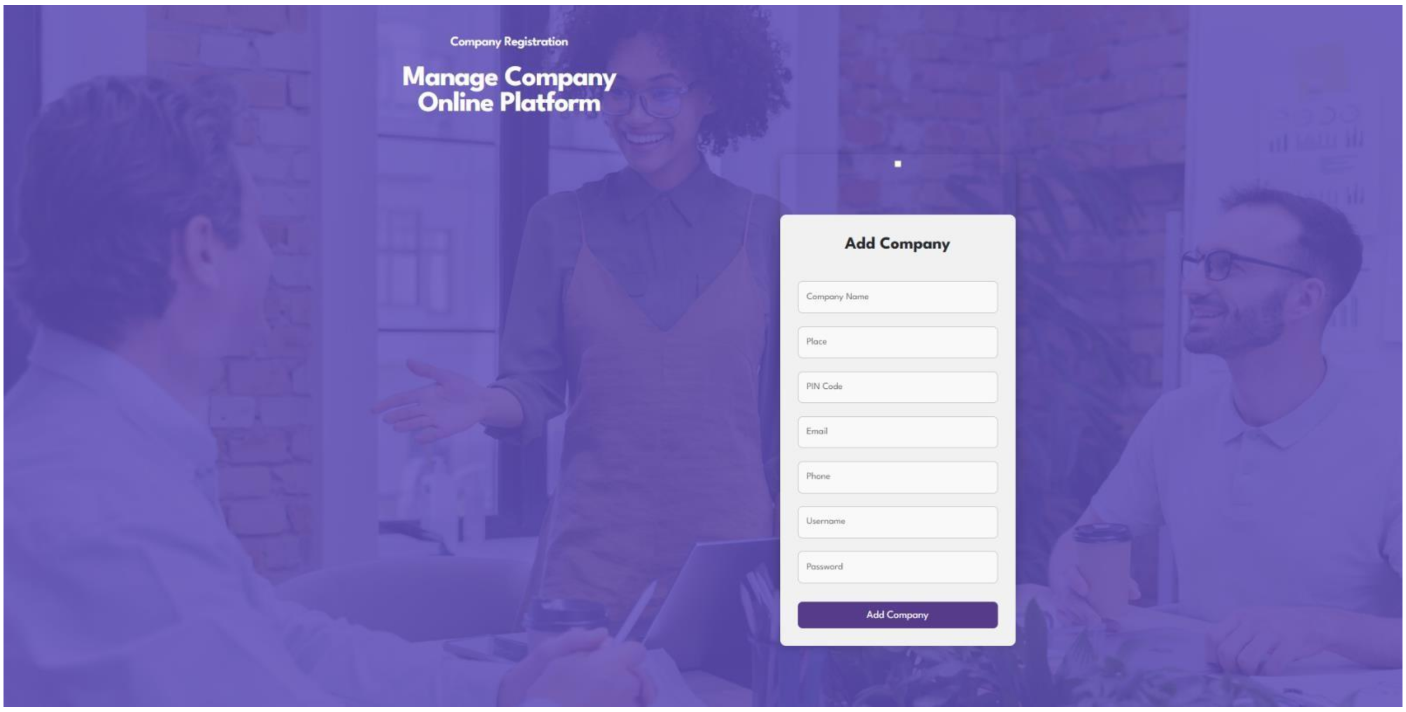
Perfective Maintenance mainly deals with implementing new or changed user requirements. Perfective Maintenance involves making functional enhancements to the system in addition to the activities to increase the system's performance even when the changes have not been suggested by faults. This includes enhancing both the function and efficiency of the code and changing the functionalities of the system as per the users' changing needs.

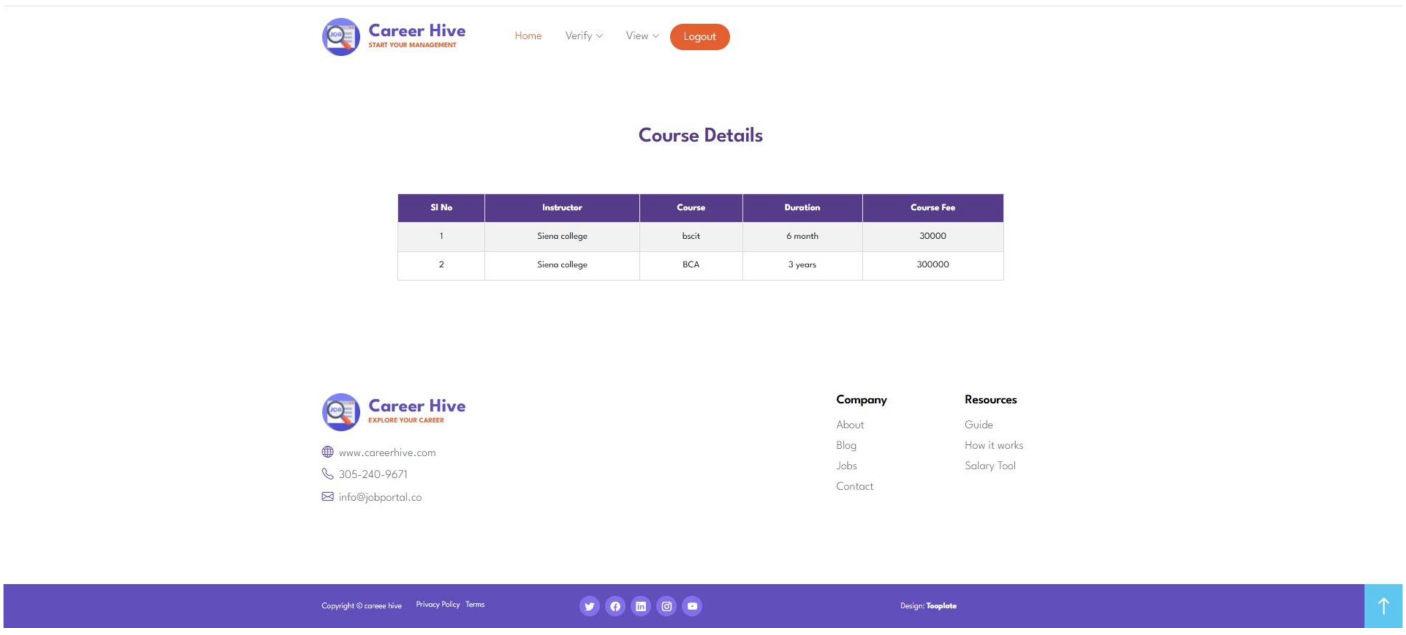
Preventive Maintenance

Preventive Maintenance involves performing activities to prevent the occurrence of errors. It tends to reduce the software complexity thereby improving program understand ability and increasing software maintainability. It comprises documentation updating, code optimization and code restructuring Documentation updating involves modifying the documents affected by the changes in order to correspond to the present state of the system. Code optimization involves modifying the programs for faster execution or efficient use of storage space. Code restructuring involves transforming the program structure for reducing the complexity in source code and making it easier to understand.

Chapter – 8

OUTPUT




Career Hive
START YOUR MANAGEMENT

Home Verify View Logout

Course Details

Sl No	Instructor	Course	Duration	Course Fee
1	Siena college	bscit	6 month	30000
2	Siena college	BCA	3 years	300000

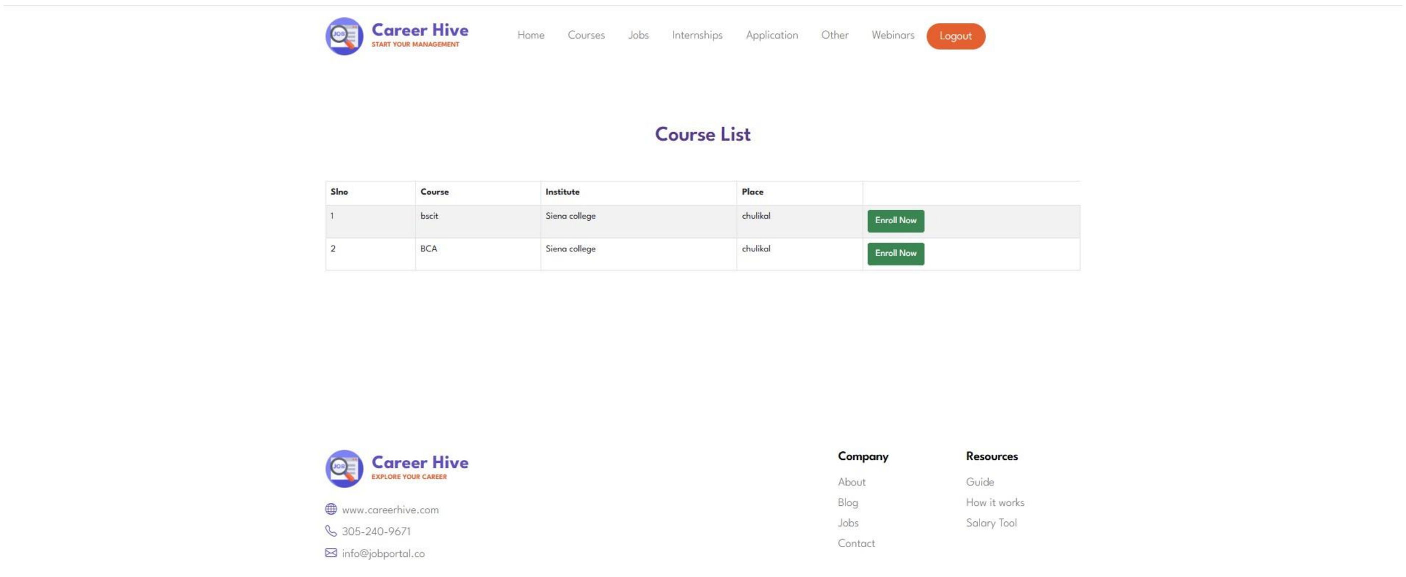
Career Hive
EXPLORE YOUR CAREER

www.careerhive.com
305-240-9671
info@jobportal.co

Company	Resources
About	Guide
Blog	How it works
Jobs	Salary Tool
Contact	

Copyright © careee hive Privacy Policy Terms

Design: Tooplate


Career Hive
START YOUR MANAGEMENT

Home Courses Jobs Internships Application Other Webinars Logout

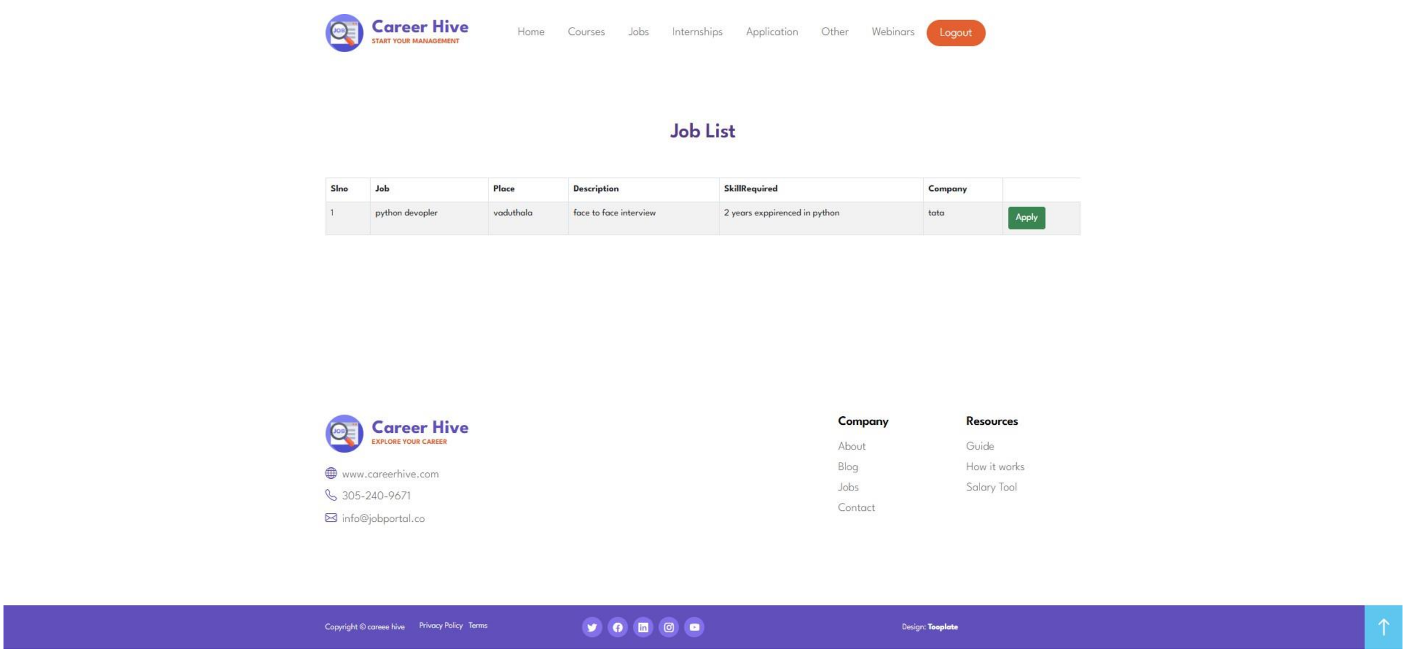
Course List

Slno	Course	Institute	Place	
1	bscit	Siena college	chulikal	Enroll Now
2	BCA	Siena college	chulikal	Enroll Now

Career Hive
EXPLORE YOUR CAREER

www.careerhive.com
305-240-9671
info@jobportal.co

Company	Resources
About	Guide
Blog	How it works
Jobs	Salary Tool
Contact	



Job List

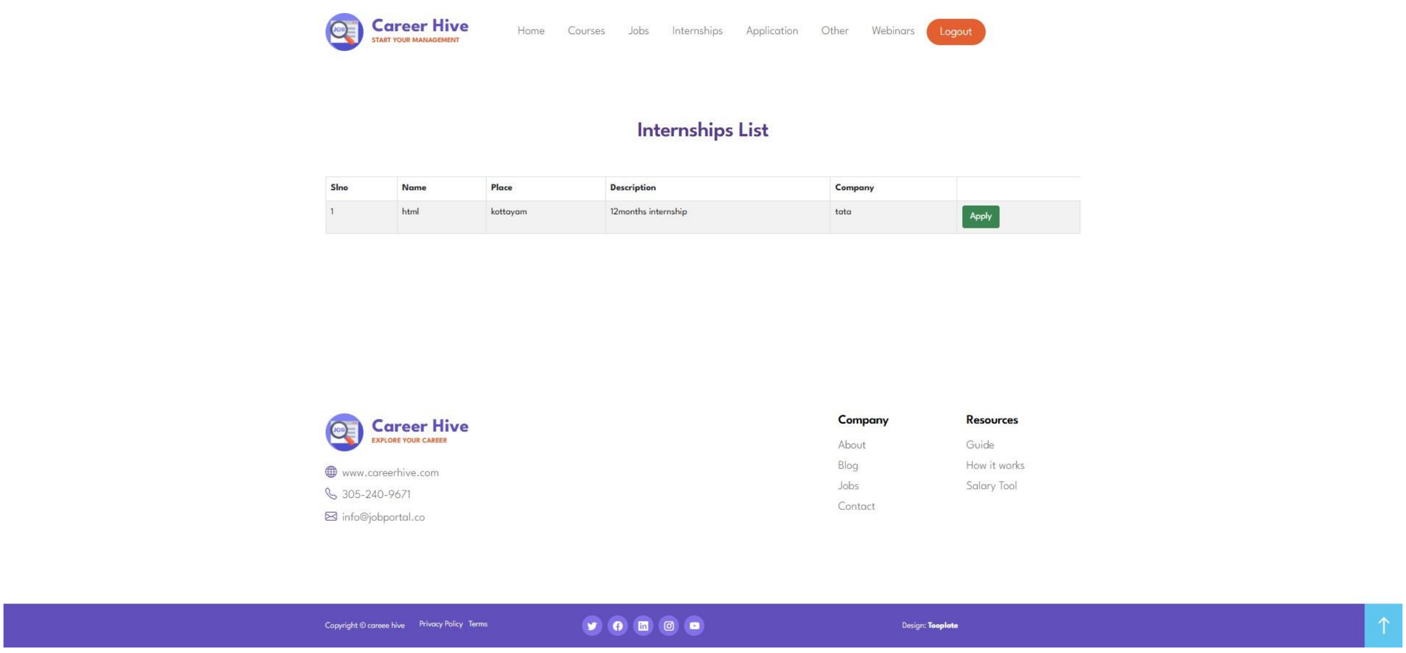
Slno	Job	Place	Description	SkillRequired	Company	
1	python devopler	vaduthala	face to face interview	2 years exppirenced in python	tata	Apply

Career Hive
EXPLORE YOUR CAREER

www.careerhive.com
305-240-9671
info@jobportal.co

Company	Resources
About	Guide
Blog	How it works
Jobs	Salary Tool
Contact	

Copyright © careee hive Privacy Policy Terms Design: **Tooplate**



Internships List

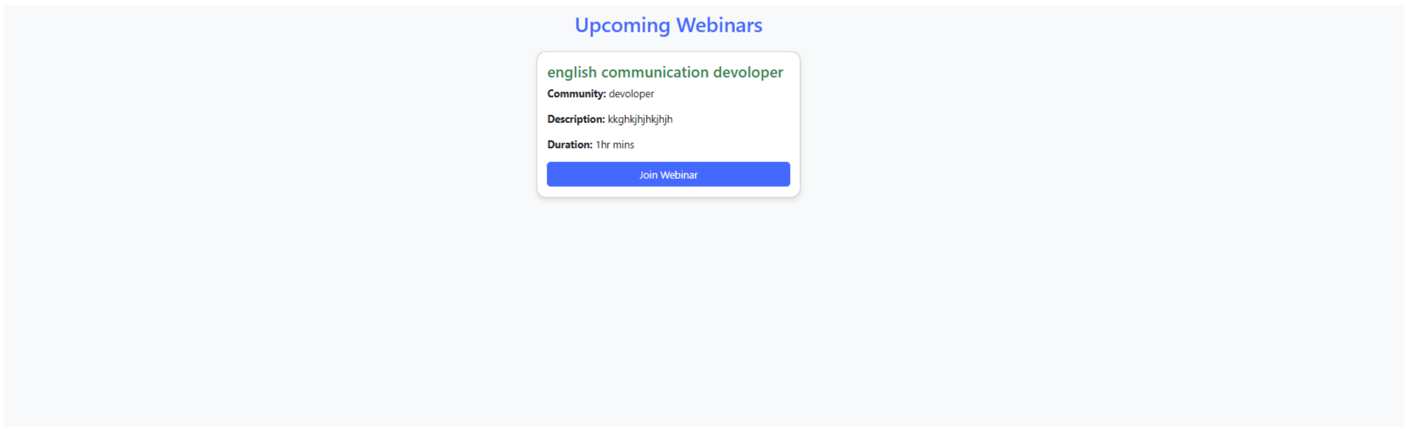
Slno	Name	Place	Description	Company	
1	html	kottayam	12months internship	tata	Apply

Career Hive
EXPLORE YOUR CAREER

www.careerhive.com
305-240-9671
info@jobportal.co

Company	Resources
About	Guide
Blog	How it works
Jobs	Salary Tool
Contact	

Copyright © careee hive Privacy Policy Terms Design: **Tooplate**



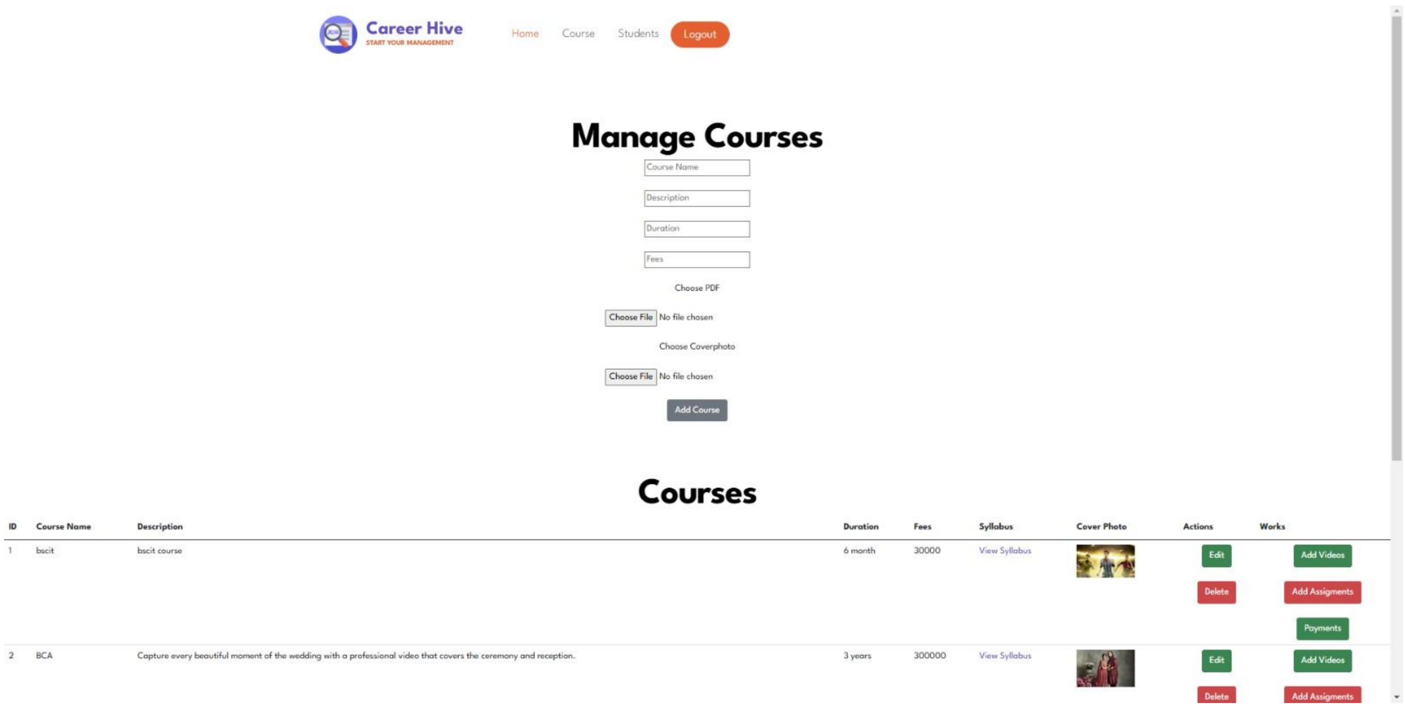
Upcoming Webinars

english communication devoloper

Community: devoloper

Description: kkghkjhjhkjhjh

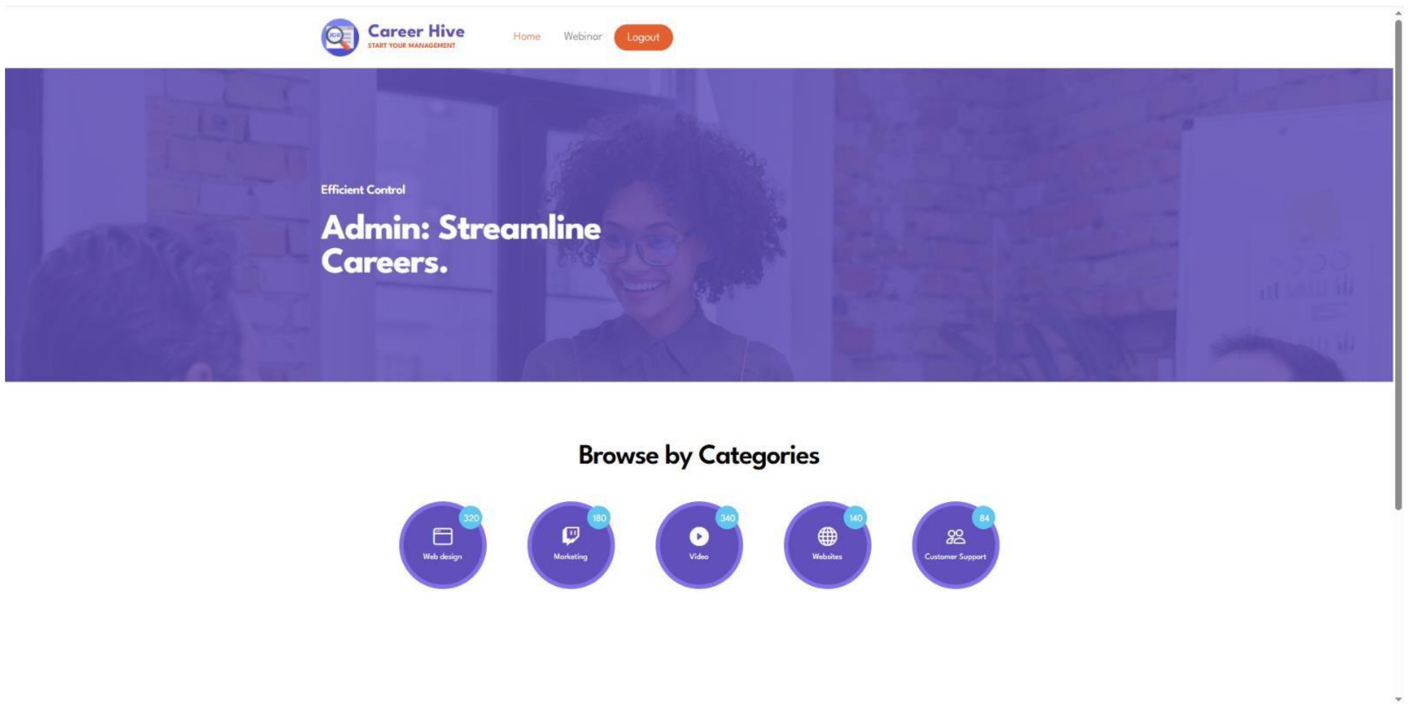
Duration: 1hr mins

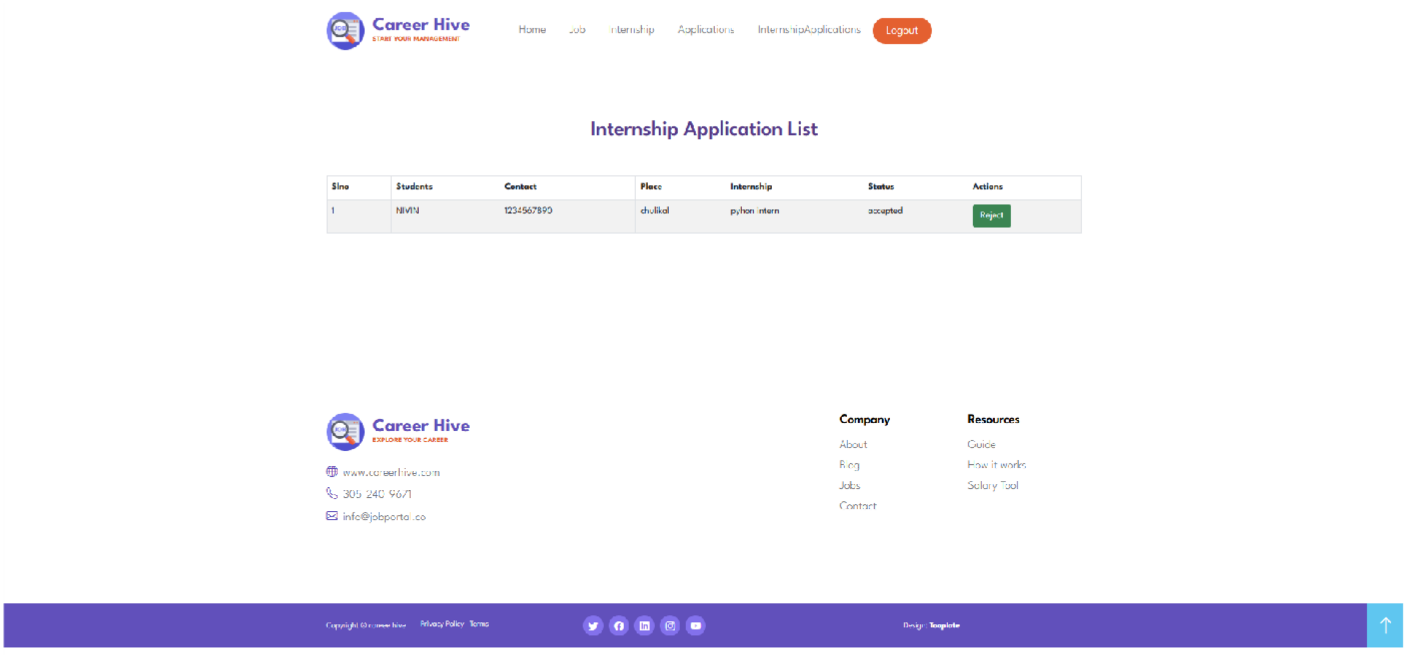


Manage Courses

Courses

ID	Course Name	Description	Duration	Fees	Syllabus	Cover Photo	Actions	Works
1	bscit	bscit course	6 month	30000	View Syllabus		Edit Delete	Add Videos Add Assigments Payments
2	BCA	Capture every beautiful moment of the wedding with a professional video that covers the ceremony and reception.	3 years	300000	View Syllabus		Edit Delete	Add Videos Add Assigments





Internship Application List

Slno	Students	Contact	Place	Internship	Status	Actions
1	NIVIN	1234567890	chulikal	pyhon intern	accepted	Reject

Chapter – 9

FUTURE ENHANCEMENTS

1. Improved Job Matching & Search

- Advanced Filtering: Enhance job search with more filters such as salary range, remote vs. onsite, and skill-based recommendations.
- Smart Job Alerts: Users receive job alerts based on their browsing history and application patterns.

2. AI-Powered Resume Screening (Optional Feature)

- Automated Resume Analysis: Employers can use AI to quickly analyse resumes and rank candidates based on job requirements.
- Skill Gap Analysis: Suggests skills users need to improve based on job market trends.

3. Internship & Apprenticeship Expansion

- Verified Internships: Introduce a system to verify and authenticate internship postings to prevent scams.
- Internship Progress Tracking: Employers can provide feedback and evaluations to interns through the platform.

4. Community & Social Features

- Webinars & Expert Sessions: Allow industry professionals to conduct live Q&A sessions and workshops.
- Networking & Mentorship: Introduce a feature for students to connect with professionals for career advice.
- Discussion Forums: Create forums where students, institutions, and companies can interact and share insights.

5. Course & Certification Enhancements

- Skill-Based Certifications: Offer certifications upon course completion to improve employability.
- Interactive Learning: Integrate quizzes, live coding environments, and discussion sections in courses.
- Course Recommendations: Suggest courses based on a user's job preferences and career goals.

6. Institution Collaboration & University Partnerships

- Verified Institutions: Ensure only recognized universities and institutions can upload courses.
- Job & Internship Tie-Ups: Partner with companies to provide direct placement opportunities for course graduates.

7. Multi-Language & Regional Support

- Language Localization: Offer Career Hive in multiple languages to make it accessible globally.
- Region-Specific Job Listings: Filter job listings based on location and region-specific requirements.

8. Blockchain for Secure Credential Verification

- Tamper-Proof Certificates: Use blockchain to verify and store course completion certificates securely.
- Employment History Records: Maintain a secure digital record of a candidate's job history for employer verification.

9. Advanced Employer & Institution Dashboards

- Data Insights: Employers and institutions get advanced analytics on job postings, applicant engagement, and hiring trends.
- AI-Based Hiring Assistance: AI suggests the most suitable candidates based on job requirements and past hiring data.

10. Gamification & Reward System

- Skill Challenges & Competitions: Encourage users to participate in career-related challenges to win rewards.
- Badges & Achievements: Users earn badges for course completions, job applications, and participation in community events.

11. Smart Interview Preparation Tools

- Mock Interviews: Offer AI-driven or expert-led mock interview sessions.
- Automated Feedback: Provide instant feedback on interview answers based on industry standards.

12. Enhanced User Experience & Mobile App Development

- Mobile-Optimized Platform: Develop a dedicated mobile app for better accessibility and user engagement.
- Intuitive UI: Simplify navigation with a personalized dashboard showing relevant jobs, courses, and community activities.

13. Employer-Student Engagement Features

- Live Hiring Events: Allow companies to host virtual job fairs and live hiring events.
- Company Profile Pages: Employers can showcase company culture, team members, and career growth opportunities.

14. AI-Powered Career Path Guidance (Optional Feature)

- Personalized Career Paths: Suggests career progression plans based on skills, education, and market trends.
- Skill Development Roadmap: Recommends step-by-step learning plans to improve employability.

15. Security & Privacy Enhancements

- Two-Factor Authentication: Enhance account security with additional authentication layers.
- Privacy Control: Users can choose what details to share with employers and institutions.

CONCLUSION

The Career Hive project is an innovative platform designed to bridge the gap between students, companies, institutions, and professional communities. By providing a centralized hub for job opportunities, internships, courses, and community engagement, it empowers users with the tools needed to build successful careers.

With features such as job and internship applications, course enrollment, community-driven webinars, and an efficient admin verification system, Career Hive streamlines the career development process. The platform ensures that students can access quality learning resources, companies can find skilled candidates, and institutions can share valuable educational content—all within a seamless digital ecosystem.

Beyond its current capabilities, Career Hive has immense potential for future enhancements, such as AI-powered job matching, blockchain-based credential verification, and gamification to encourage professional growth. Its scalable architecture allows for continuous improvement, making it a future-ready solution for career development.

In conclusion, Career Hive is more than just a job portal—it is a comprehensive career development platform that connects education, employment, and professional networking. By integrating technology with career-building resources, it empowers users to achieve their professional aspirations while fostering industry growth and collaboration

Chapter – 11

BIBLIOGRAPHY

1. **Software Engineering: A Practitioner's Approach by Roger S. Pressman**

A comprehensive guide on software development methodologies, which can be used to design and implement the platform.

2. **Database System Concepts by Abraham Silberschatz, Henry F. Korth, and S. Sudarshan**

Provides insights into database design and management, which is crucial for handling user data, service provider details, and transactions.

3. **Global Automotive Repair and Maintenance Services Market Report by Grand View Research**

Provides market trends and insights into the automotive repair industry, which can help in understanding user needs and market demands.

4. **Case Study: Predictive Maintenance in the Automotive Industry by IBM**

Explores how predictive maintenance is being used in the automotive industry, offering ideas for future enhancements.