Learn To Earn

A Student Earning Software

Nebin Reji
S6 BCA B

1. INTRODUCTION

1.1 BACKGROUND AND MOTIVATION

The project titled **"LearnToEarn"** is developed with the objective of providing a dedicated and student-centric platform that helps students find safe, flexible, and verified part-time job opportunities. In many regions, students who wish to earn alongside their studies often depend on informal sources such as word-of-mouth, social media groups, or unverified online listings. These methods frequently result in problems such as fake job postings, unsafe work environments, irregular payments, and excessive working hours that negatively affect academic performance.

With the increasing cost of education and living expenses, many students seek part-time employment to support themselves financially and gain practical experience. However, traditional job portals are primarily designed for full-time professionals and do not consider the unique needs of students, such as limited availability, academic schedules, and the need for short-duration or flexible work. This creates a gap where students struggle to find suitable opportunities that balance both earning and learning.

The motivation behind developing **LearnToEarn** arises from the need to create a technology-driven solution that bridges this gap by connecting students with verified employers and meaningful part-time opportunities. The platform focuses on simplifying the entire job-seeking process for students — from discovering suitable opportunities and applying for jobs to tracking applications and managing work schedules responsibly.

By incorporating features such as role-based access, verified job postings, skill-based profiles, and workload monitoring, LearnToEarn reduces the risks associated with informal employment and promotes responsible earning. The system enhances transparency, minimizes exploitation, and ensures that students can gain income and real-world experience without compromising their education. Ultimately, LearnToEarn aims to empower students by providing a secure and structured environment that supports financial independence, skill development, and academic balance.

1.2 THE PROPOSED SYSTEM

The proposed system, **LearnToEarn**, is a web-based student opportunity platform designed to digitalize and streamline the process of discovering, applying for, and managing part-time job opportunities for students. The system focuses on providing a

safe, flexible, and student-friendly environment that connects all key stakeholders involved in student employment, including students, employers, and the system administrator, within a single centralized platform to ensure transparency, efficiency, and academic balance.

The **administrator** acts as the primary controller of the system, responsible for managing overall platform operations. This includes verifying employer registrations, approving job postings, managing job categories and locations, and monitoring system activities. The admin ensures that all listed opportunities are legitimate, student-appropriate, and aligned with platform guidelines. The system also allows the admin to generate analytical reports, monitor user activity, and handle complaints or disputes, thereby maintaining trust and platform integrity.

Employers are provided with secure login access to post part-time job opportunities tailored for students. Employers can define job requirements such as skills needed, work duration, working hours, and location. They can view student applications, shortlist candidates, schedule interviews, and finalize hiring decisions. Employers are also responsible for updating job statuses and providing feedback after job completion, which helps in maintaining a transparent hiring process.

Students can register on the platform, create detailed profiles, and specify their skills, availability, and preferences. The system enables students to browse verified job listings, apply for suitable opportunities, track application status, and receive notifications regarding interviews or selection updates. LearnToEarn emphasizes responsible earning by allowing students to manage their workload and maintain a balance between academics and work.

In addition to employer-posted jobs, the platform supports **student-to-student services**, where students can offer skill-based services such as tutoring, mentoring, design assistance, or basic technical support to fellow students. This feature promotes peer learning and provides additional earning opportunities within the student community in a controlled and ethical manner.

Overall, the proposed system aims to bridge the gap between students seeking flexible earning opportunities and employers looking for reliable part-time assistance, while prioritizing academic wellbeing. LearnToEarn promotes financial independence, skill development, and responsible work culture, making it a sustainable and practical solution for student employment.

1.3 PROJECT SCOPE

1.3.1 Limitations of Existing System

The The existing system for finding part-time jobs for students is largely informal and unorganized, leading to several difficulties and risks for students. The absence of a centralized, student-focused digital platform results in poor coordination between students and employers. Below are the major limitations identified in the current system:

1. Lack of a centralized student job platform
2. Dependence on informal sources such as social media and word-of-mouth
3. Presence of fake or unverified job postings
4. No student-specific job filtering or flexibility
5. Poor communication between students and employers
6. No proper tracking of job applications and status
7. Absence of workload or time-limit monitoring for students
8. Absence of workload or time-limit monitoring for students
9. Limited focus on student safety and job authenticity
10. No structured record of skills, experience, or completed work
11. Low transparency and accountability in part-time hiring
12. Difficulty in balancing academics with work commitments

1.3.2 Advantages of Proposed System

The proposed system, **BioBinConnect**, overcomes the drawbacks of the traditional manual waste management process by providing an automated, organized, and transparent digital platform. It ensures real-time monitoring, effective coordination, and eco-friendly compost utilization. The key advantages are as follows:

1. Provides a centralized digital platform for managing all student part-time job activities.
2. Ensures verified and safe job opportunities through admin-controlled approval.
3. Improves communication between students, employers, and administrators.
4. Enables easy job application, tracking, and status management for students.
5. Automates job posting, application handling, and notification processes.
6. Maintains transparency and accountability in the hiring process.
7. Supports student life balance through flexible job options and workload monitoring.
8. Stores structured data for report generation and system analysis.
9. Reduces dependency on informal and unreliable job-search methods.
10. Encourages responsible earning while prioritizing academic commitments.
11. Can be scaled and enhanced with future technologies and features.

2. SYSTEM ANALYSIS

2.1 INTRODUCTION

Software Engineering is the analysis, design, construction, verification and management of technical or social entities. To engineer software accurately, a software engineering process must be defined. System analysis is a detailed study of the various operations performed by the system and their relationship within and module of the system. It is a structured method for solving the problems related to the development of a new system. The detailed investigation of the present system is the focal point of system analysis. This phase involves the study of parent system and identification of system objectives. Information has to be collected from all people who are affected by or who use the system. During analysis, data are collected on the variable files, decision point and transactions handled by the present system. The main aim of system is to provide the efficient and user-friendly automation. So, the system analysis process should be performed with extreme precision, so that an accurate picture of existing system, its disadvantages and the requirements of the new system can be obtained.

System analysis involves gathering the necessary information and using the structured tool for analysis. This includes the studying existing system and its drawback, designing a new system and conducting cost benefit analysis. System analysis is a problem-solving activity that requires intensive communication between the system users and system developers. The system is studied to the minute detail and analyzed. The system is viewed as a whole and the inputs to the system are identified. The outputs from the organization are traced through various phases of processing of inputs.

There are a number of different approaches to system analysis. When a computer-based information system is developed, systems analysis (according to the Waterfall model) would constitute the following steps:

- The development of a feasibility study, involving determining whether a project is economically, technologically and operationally feasible.
- Conducting fact-finding measures, designed to ascertain the requirements of the system's end-users. These typically span interviews, questionnaires, or visual observations of work on the existing system.

- Gauging how the end-users would operate the system (in terms of general experience in using computer hardware or software), what the system would be used for and so on

Techniques such as interviews, questionnaires etc. can be used for the detailed study of these processes. The data collected by these sources must be scrutinized to arrive at a conclusion.

The conclusion is an understanding of how the system functions. This system is called the Existing System. The Existing system is then subjected to close observation and the problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as a proposal which is the Proposed System. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is then presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is a loop that ends as soon as the user is satisfied with the proposal.

2.2 STAKEHOLDERS IN THIS PROJECT

2.2.1 ADMIN

The **Admin** in the **LearnToEarn** system is responsible for managing all platform activities and ensuring smooth coordination between students, employers, and the system itself. The admin oversees user registrations, verifies employer accounts, and controls access based on assigned roles to maintain platform security and trust. By validating employer profiles and approving job postings, the admin ensures that all opportunities listed on the platform are legitimate, student-appropriate, and compliant with platform guidelines.

In addition to user management, the admin supervises the overall workflow of job postings and applications. This includes managing job categories, locations, and opportunity types, as well as monitoring application progress and interview scheduling activities. The admin ensures that communication between students and employers is transparent and structured, reducing confusion and delays in the hiring process. Any misuse of the platform, suspicious activity, or violation of rules is monitored and addressed by the admin.

The admin also handles system-level management such as maintaining notifications,

resolving user feedback and complaints, and taking corrective actions when required. The platform allows the admin to generate detailed reports on student participation, employer activity, job trends, application statistics, and overall system performance. These reports help in evaluating platform effectiveness and identifying areas for improvement.

Through continuous monitoring, verification, and informed decision-making, the admin ensures that the entire **LearnToEarn ecosystem** operates smoothly, securely, and efficiently. By maintaining transparency, accountability, and student safety, the admin plays a crucial role in supporting responsible earning opportunities while prioritizing students' academic wellbeing.

2.2.2 EMPLOYER

The The **Employer** in the **LearnToEarn** system is responsible for creating and managing part-time job opportunities suitable for students. Employers log into the platform to post job listings by specifying job details such as role description, required skills, working hours, duration, location, and compensation. They ensure that the jobs offered are student-friendly, flexible, and aligned with the platform's guidelines. Employers can view and manage their posted opportunities through a dedicated dashboard.

After posting a job, employers review applications submitted by students and evaluate candidates based on their profiles, skills, and availability. The system allows employers to update application statuses, shortlist suitable candidates, and schedule interviews by setting interview dates directly within the platform. Employers can communicate important instructions or updates to students through system notifications, ensuring clear and structured communication.

Once a student is selected, the employer confirms the hiring decision and oversees the completion of assigned work. Employers are responsible for updating job completion status and providing feedback on student performance after the work is completed. This feedback contributes to the student's skill and helps maintain transparency and accountability within the platform.

By maintaining accurate job details, fair selection practices, and timely updates, employers play a key role in ensuring that LearnToEarn functions as a reliable and trustworthy student employment ecosystem. Their participation enables students to gain

meaningful work experience while balancing academic commitments.

2.2.3 STUDENT

The **Student** in the **LearnToEarn** system is responsible for participating in part-time job opportunities and services offered through the platform while maintaining academic balance. Students log into the system to create and manage their profiles, update personal details, specify skills, and indicate their availability for work. They browse verified job listings posted by employers and apply for suitable opportunities based ontheir interests, qualifications, and time constraints.

Once an application is submitted, students track the status of their applications in real time and receive notifications regarding shortlisting, interview schedules, or selection decisions. When shortlisted, students attend interviews or follow employer instructions as communicated through the platform. After being accepted, students carry out the assigned work according to the agreed schedule and job requirements.

Throughout their participation, students are responsible for adhering to platform guidelines, maintaining professionalism, and completing assigned tasks responsibly. They can report issues, provide feedback after job completion, and stay updated with notifications related to job status or schedule changes. By managing their applications, availability, and participation effectively, students ensure smooth interaction with employers while balancing work commitments alongside their academic responsibilities.

2.3 SOFTWARE REQUIREMENT SPECIFICATION

2.3.1 ADMIN

1. System should have the facility to login using username and password.
2. Systemshould have the facility to view the homepage of admin.
3. Admin should have the facility to approve/reject/remove employer accounts.
4. Admin should have the facility to view student and employer accounts.
5. Admin should have the facility to verify job postings by employers before publishing.
6. Admin should have the facility to view all job listings, student details, and employer details.
7. Admin should have the facility to handle complaints or disputes raised by students or employers.
8. Admin should have the facility to add/view/edit/delete districts and locations.
9. Admin should have the facility to generate systemreports (user statistics, job activity, complaints, etc.).

10. Admin should have the facility to block or deactivate fake or inactive users.
11. Admin should have the facility to send notifications or warnings to users.
12. The system should have the facility for secure Logout.

2.3.2 EMPLOYER

1. Systemshould have the facility to login using username and password.
2. Systemshould have the facility to view the homepage of employer
3. Employer should have the facility to create and update their company profile.
4. Employer should have the facility to add/edit/delete new part-time job postings
5. Employer should have the facility to view the status of job posts
6. Employer should have the facility to view and manage student applications.
7. Employer should have the facility to shortlist, reject, or select students
8. Employer should have the facility to schedule interviews or contact selected students
9. Employer should have the facility to mark a job as completed or position filled
10. Employer should have the facility to give feedback/complaints
11. The system should have the facility for secure Logout.

2.3.3 STUDENT

1. Student should have the facility to login using user name and password.
2. Student should have the facility to view the homepage of student
3. Student should have the facility to update or edit their profile details
4. Student should have the facility to browse verified/Unverified job listings posted by employers
5. Student should have the facility to search or filter jobs based on category, type duration, and location.
6. Student should have the facility to view full job details and apply.
7. Student should have the facility to track the status of applications
8. Student should have the facility to withdraw applications
9. Student should have the facility to rate and give feedback/complaint
10. Student should have the facility to view job history
11. The system should have the facility for secure Logout.

Table 2.1 Sign off table

Sl. No.	Name & Designation	Date	Accepted (Yes / No)
1	Ashwin TJ Assistant Professor Santhigiri College of Computer Sciences		
3	Nebin Reji Developer		

2.4 FEASIBILITY STUDY

Feasibility is defined as the practical extent to which a project can be performed successfully. To evaluate feasibility, a feasibility study is performed, which determines whether the solution considered to accomplish the requirements is practical and workable in the software. Information such as resource availability, cost estimation for software development, benefits of the software to the organization after it is developed and cost to be incurred on its maintenance are considered during the feasibility study. The objective of the feasibility study is to establish the reasons for developing the software that is acceptable to users, adaptable to change and conformable to established standards. Various other objectives of feasibility study are listed below.

- To analyze whether the software will meet organizational requirements.
- To determine whether the software can be implemented using the current technology and within the specified budget and schedule.
- To determine whether the software can be integrated with other existing software. When our project guide as well as our client Ms. Dalbina Dalan told us regarding the project and about Word to the Wise for getting the desired product developed, it comes up with rough idea about what all functions the software must perform and which all features are expected from the software.

Referencing to this information, we have done studies and discussions about whether the desired system and its functionality are feasible to develop and the output of this phase is a feasibility studyreport that should contained adequate comments and recommendations.

Various types of feasibility that we checked include technical feasibility, operational

feasibility, and economic feasibility.

Technical Feasibility

Technical feasibility assesses the current resources (such as hardware and software) and technology, which are required to accomplish user requirements in the software within the allocated time and budget. For this, the software development team ascertains whether the current resources and technology can be upgraded or added in the software to accomplish specified user requirements. Technical feasibility also performs the following tasks.

- Analyses the technical skills and capabilities of the software development team members.
- Determines whether the relevant technology is stable and established.
- Ascertains that the technology chosen for software development has a large number of users so that they can be consulted when problems arise or improvements are required.

From our perspective there are two languages PYTHON, HTML and database MySQL which are used to develop this web-based applications. HTML and CSS are used in the front end and PYTHON is used in the back end. The Word to the Wise is web based and thus can be accessed through any browsers. As I am using these latest technologies which are currently trending and used by a number of developers across the globe, we can say that our project is technically feasible.

Operational Feasibility

Operational feasibility assesses the extent to which the required software performs a series of steps to solve business problems and user requirements. This feasibility is dependent on human resources (software development team) and involves visualizing whether the software will operate after it is developed and be operative once it is installed. Operational feasibility also performs the following tasks.

- Determines whether the problems anticipated in user requirements are of high priority.
- Determines whether the solution suggested bythe software development team is acceptable.
- Analyses whether users will adapt to a new software.
- Determines whether the organization is satisfied by the alternative solutions proposed by the software development team.

We found that our project will be satisfied for the client since we were discussing every detail about the software with the client at every step. The most important part of operational feasibility study is the input from client. So, the software is built completely according to the requirements of the client. We have used the current industry standards for the software. Hence, we can say that this software is operationally feasible.

Economic Feasibility

Economic feasibility determines whether the required software is capable of generating financial gains for an organization. It involves the cost incurred on the software development team, estimated cost of hardware and software, cost of performing feasibility study, and so on. For this, it is essential to consider expenses made on purchases (such as hardware purchase) and activities required to carry out software development. In addition, it is necessary to consider the benefits that can be achieved by developing the software. Software is said to be economically feasible if it focuses on the issues listed below.

- Cost incurred on software development to produce long-term gains for an organization.
- Cost required to conduct full software investigation (such as requirements elicitation and requirements analysis).
- Cost of hardware, software, development team, and training.

It is estimated that our project is economically feasible as development cost is very minimal since the tools and technologies used are available online. It's a group student project so there are no personal costs. Development time is well planned and will not affect other operations and activities of the individuals. Once the system has been developed, the companies purchasing the system will be providing with a manual for training purposes. There is no need to purchase new hardware since the existing computers can still be used to implement the new system.

2.5 SOFTWARE DEVELOPMENT LIFE CYCLE MODEL

One of the basic notions of the software development process is SDLC models which stand for Software Development Life Cycle models. SDLC – is a continuous process, which starts from the moment, when it's made a decision to launch the project, and it ends at the moment of its full remove from the exploitation. Software development lifecycle (SDLC) is a framework that defines the steps involved in the development of software. It covers the detailed plan for building, deploying and maintaining the software. SDLC

defines the complete cycle of development i.e. all the tasks involved in gathering a requirement for the maintenance of a product.

Some of the common SDLC models are Waterfall Model, V-Shaped Model, Prototype Model, Spiral Model, Iterative Incremental Model, Big Bang Model, Agile Model. We used Agile Model for our Project.

Agile Model

Agile Model is a combination of Iterative and incremental model. This model focuses more on flexibility while developing a product rather than on the requirement. In the agile methodology after every development iteration, the client is able to see the result and understand if he is satisfied with it or he is not. Extreme programming is one of the practical uses of this model. The basis of this model consists of short meetings where we can review our project. In Agile, a product is broken into small incremental builds. It is not developed as a complete product in one go. At the end of each sprint, the project guide verifies the product and after his approval, it is finalized. Client feedback is taken for improvement and his suggestions and enhancement are worked on in the next sprint. Testing is done in each sprint to minimize the risk of any failures.

Advantages of Agile Model:

- It allows more flexibility to adapt to the changes.
- The new feature can be added easily.
- Customer satisfaction as the feedback and suggestions are taken at every stage.
- Risks are minimized thanks to the flexible change process.

Disadvantages of Agile Model:

- Lack of documentation.
- If a customer is not clear about how exactly they want the product to be, then the project would fail.
- With all the corrections and changes there is possibility that the project willexceed expected time.

2.6. HARDWARE AND SOFTWARE REQUIREMENTS

2.6.1 SOFTWARE SPECIFICATION

This project is built upon the latest technology software.

Front end : HTML, CSS, JavaScript

Development tool	:	Python Django
Database	:	MySQL
Frame work	:	Django, Bootstrap
Web server	:	WAMP server
Operating System	:	Windows 11

2.6.1.1 PYTHON DJANGO

Django, an open-source web framework written in Python, facilitates rapid and scalable web development with a clean, pragmatic design. At its core is the Model-View-Template (MVT) architectural pattern, comprising models for data structures, views for handling user requests and rendering responses, and templates for defining HTML structure. Django's Object-Relational Mapping (ORM) simplifies database interaction by translating Python models into corresponding database tables. The framework boasts a dynamic admin interface for effortless database management and customization. URL routing is declarative, providing a clear structure for web applications, while the template system allows the separation of HTML and Python code. Middleware components enhancerequest/response processing, enabling tasks like authentication and security checks. Django includes a form handling system for easy validation and processing of HTML forms, and its security features guard against common vulnerabilities such as Cross-Site Scripting (XSS) and Cross-Site Request Forgery (CSRF). The built-in authentication system manages user-related functionalities, while the testing framework ensures code reliability. Optionally, the Django REST framework extends capabilities for building robust Web APIs with features like serializers, authentication, and view sets. With a command-line interface (CLI) for various tasks, Django empowers developers to create scalable, secure, and maintainable web applications efficiently, making it a preferred choice for web development projects globally.

2.6.1.2 MySQL

MySQL is the world's most popular open-source database software, with over 100 million copies of its software downloaded or distributed throughout its history. With its superior speed, reliability, and ease of use, MySQL has become the preferred choice for Web, Web 2.0, SaaS, ISV, Telecom companies and forward-thinking corporate IT Managers because it eliminates the major problems associated with downtime, maintenance and administration for modern, online applications.

Many of the world's largest and fastest-growing organizations use MySQL to save time and money powering their high-volume Web sites, critical business systems, and packaged software — including industry leaders such as Yahoo!, Alcatel-Lucent, Google, Nokia, YouTube, Wikipedia, and Booking.com.

The flagship MySQL offering is MySQL Enterprise, a comprehensive set of production tested software, proactive monitoring tools, and premium support services available in an affordable annual subscription.

MySQL is a key part of LAMP (Linux, Apache, MySQL, PHP / Perl / Python), the fast-growing open-source enterprise software stack. More and more companies are using LAMP as an alternative to expensive proprietary software stacks because of its lower cost and freedom from platform lock-in.

MySQL was originally founded and developed in Sweden by two Swedes and a Finn: David Axmark, Allan Larsson and Michael "Monty" Widenius, who had worked together since the 1980's. MySQL, the most popular Open-Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.

MySQL is a database management system. A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

MySQL databases are relational. A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to one, one- to-many, unique, required or optional, and --pointers between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data.

The SQL part of --MySQL‖ stands for --Structured Query Language. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate

reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, ―SQL-92‖ refers to the standard released in 1992, ―SQL:1999 refers to the standard released in 1999, and ―SQL:2003 refers to the current version of the standard.

We use the phrase ―the SQL standard‖ to mean the current version of the SQL Standard at any time. MySQL software is Open Source. Open-Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs.

The MySQL software use the GPL (GNU General Public License), http://www.fsf.org/licenses/, to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us.

The MySQL Database Server is very fast, reliable, scalable, and easy to use. If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available. MySQL can also scale up to clusters of machines, networked together.

MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet. MySQL Server works in client/server or embedded systems.

The MySQL Database Software is a client/server system that consists of a multi- threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs). We also provide MySQL Server as an embedded multi-threaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

A large amount of contributed MySQL software is available. MySQL Server has a practical set of features developed in close cooperation with our users. It is very likely that your favorite application or language supports the MySQL Database Server.

2.6.1.3 VISUAL STUDIO

Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including C, C#, C++, FORTRAN, Go, Java, JavaScript, Node.js, Python, Rust, and Julia. It is based on the Electron framework, which is used to develop Node.js web applications that run on the Blink layout engine. Visual Studio Code employs the same editor component (code named "Monaco") used in Azure DevOps.

Out of the box, Visual Studio Code includes basic support for most common programming languages. This basic support includes syntax highlighting, bracket matching, code folding, and configurable snippets. Visual Studio Code also ships with Intelligent for JavaScript, Type Script, JSON, CSS, and HTML. Support for additional languages can be provided by freely available extensions on the VS Code Marketplace. Instead of a project system, it allows users to open one or more directories, which can then be saved in work

spaces for future reuse. This allows it to operate as a language agnostic code editor for any language. It supports many programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via the settings. Many Visual Studio Code features are not exposed through menus or the user interface but can be accessed via the command palette.

2.6.1.4 WINDOWS 11

Operating System is defined as a program that manages the computer hardware. An operating system can be viewed as a scheduler, where it has resources for which it has charge. Resources include CPU, memory, I/O device and disk space. In another view, the operating system is a new machine. The third view is that operating system is a multiplexer which allows sharing of resources provides protection from interference and provides a level of cooperation between users. This project is developed using Windows 11 as the operating system and supports its latest versions. Windows 11 is the latest installment in Microsoft's Windows NT family of operating systems, succeeding Windows 10. It was officially announced on June 24, 2021, and released to manufacturing on October 5, 2021, with general availability starting on October 5, 2021. Windows 11 introduces a refreshed user interface with centered Start Menu and taskbar icons, enhanced snap layouts, and

virtual desktops. It also integrates Microsoft Teams directly into the operating system, providing seamless communication and collaboration capabilities. Other new features include support for running Android apps through the Amazon Appstore, a revamped Microsoft Store with a broader range of applications, improved gaming performance through Auto HDR and Direct Storage, and enhanced security features such as Windows Hello for faster, more secure login options. Initial reviews of Windows 11 have been generally positive, praising its design changes, performance improvements, and new features. Critics have noted concerns over hardware requirements and compatibility issues, particularly with older hardware, as well as some controversial decisions such as the requirement for a Microsoft account and internet connection during setup. Despite these criticisms, Windows 11 represents Microsoft's continued evolution of its operating system, aiming to enhance productivity, security, and user experience across a range of devices.

2.6.1.2 MICROSOFT WORD

Microsoft Word (or simply Word) is a word processor developed by Microsoft. It was first released on October 25, 1983 under the name Multi-Tool Word for Xenix systems. Subsequent versions were later written for several other platforms including IBM PCs running DOS (1983), Apple Macintosh running the Classic Mac OS (1985), AT&T Unix PC (1985), Atari ST (1988), OS/2 (1989), Microsoft Windows (1989), SCO Unix (1994), and mac OS (formerly OS X; 2001).Commercial versions of Word are licensed as a standalone product or as a component of Microsoft Office, Windows RT or the discontinued Microsoft Works suite. Unlike most MS-DOS programs at the time, Microsoft Word was designed to be used with a mouse. Advertisements depicted the Microsoft Mouse, and described Word as a WYSIWYG, windowed word processor with the ability to undo and display bold, italic, and underlined text, although it could not render fonts. It was not initially popular, since its user interface was different from the leading word processor at the time, WordStar. However, Microsoft steadily improved the product, releasing versions 2.0 through 5.0 over the next six years. In 1985, Microsoft ported Word to the classic Mac OS (known as Macintosh System Software at the time). This was made easier by Word for DOS having been designed for use with high-resolution displays and laser printers, even though none were yet available to the general public. Following the precedents of Lisa Write and MacWrite, Word for Mac OS added true WYSIWYG features. It fulfilled a need for a word processor that was more capable than MacWrite.

After its release, Word for Mac OS's sales were higher than its MS-DOS counterpart for at least four years.

2.6.2 HARDWARE REQUIREMENTS

The selection of hardware configuring is a very task related to the software development, particularly inefficient RAM may affect adversely on the speed and corresponding on the efficiency of the entire system. The processor should be powerful to handle all the operations. The hard disk should have the sufficient to solve the database and the application.

Hardware used for development:

CPU :Intel i5 Processer

Memory : 4 GB

Cache : 6 MB

Hard Disk: 1 TB

Monitor : 15.6” Monitor

Keyboard : Standard108 keys Enhanced Keyboard

Mouse : Optical Mouse

Minimum Hardware Required for Implementation:

CPU : Pentium IV Processor

Memory : 256MB Above

Cache : 512 KB Above

Hard Disk: 20 GB Above

Monitor : Any

Keyboard : Any

Mouse : Any

3. SYSTEM DESIGN

3.1 SYSTEM ARCHITECTURE

A system architecture or system's architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures of the system.

System architecture can comprise system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture; collectively these are called architecture description languages (ADLs).

The system architecture can best be thought of as a set of representations of an existing (or to be created) system. It is used to convey the informational content of the elements comprising a system, the relationships among those elements, and the rules governing those relationships. The architectural components and set of relationships between these components that architecture describes may consist of hardware, software, documentation, facilities, manual procedures, or roles played by organizations or people. System architecture is primarily concerned with the internal interfaces among the system's components or subsystems, and the interface between the system and its external environment, especially the user.

The structural design reduces complexity, facilitates change and result in easier implementation by encouraging parallel development of different parts of the system. The procedural design transforms structural elements of program architecture into a procedural description of software components. The architectural design considers architecture as the most important functional requirement. The system is based on the three-tier architecture.

The first level is the user interface (presentation logic), which displays controls, receives and validates user input. The second level is the business layer (business logic) where the application specific logic takes place. The third level is the data layer where the application information is stored in files or database. It contains logic about to retrieve and update data. The important feature about the three-tier design is that information only travels from one level to an adjacent level.

3.2 MODULE DESIGN

Modular programming is a software design technique that emphasizes separating the functionality of a program into independent, interchangeable modules, such that each contains everything necessary to execute only one aspect of the desired functionality. Conceptually, modules represent a separation of concerns, and improve maintainability by enforcing logical boundaries between components.

Different modules of this project include:

1. **Authentication**

   This module ensures a secure and efficient login process for all stakeholders in the **LearnToEarn** system, including students, employers, and the administrator. The admin logs in using authorized credentials and gains access to manage user verification, job approvals, category management, system monitoring, and report generation. Employers log in to post part-time job opportunities, view student applications, shortlist candidates, schedule interviews, and manage job statuses. Students log in to browse verified job listings, apply for suitable opportunities, track application status, and receive interview or selection updates.After successful authentication, each user is redirected to their respective dashboard based on their assigned role, ensuring role-based access control and data security. This module also includes a "Forgot Password" feature that allows users to securely reset their passwords through verification mechanisms, ensuring uninterrupted access to the platform. By enforcing secure authentication and role-based navigation, this module forms the foundation for system security and reliable user interaction within LearnToEarn.

2. **Registeration**

   This module manages the registration process for all stakeholders in the **LearnToEarn** system, including students and employers. Students register by providing personal details, academic information, contact details, and basic skill preferences. Employers register by submitting organization details, contact information, and required verification data needed to post part-time job opportunities on the platform.All employer registrations are subject to verification by the admin before account activation to ensure the authenticity and safety of job postings. The admin reviews submitted details and either approves or rejects registrations based on platform guidelines. Once verified, users are granted login access and assigned roles that determine their systemprivileges. This module ensures controlled

participation, prevents fake accounts, and maintains the overall reliability of the LearnToEarn platform.

3. **Activity**

This module handles the core operational activities of the **LearnToEarn** system related to part-time job management. Employers use this module to create, update, and manage job postings by specifying job roles, required skills, working hours, duration, and location. Students access this module to browse verified job opportunities, apply for suitable positions, and manage their applications.The module also allows employers to review applications, shortlist candidates, schedule interviews, and update application statuses such as accepted or rejected. Students receive real-time updates regarding application progress and interview schedules through system notifications. By coordinating all job-related interactions between students and employers, this module ensures a smooth, transparent, and structured employment workflow.

4. **Payment**

This module manages all payment-related activities within the **LearnToEarn** system. It records payments made by employers for posted job opportunities and tracks student earnings after the successful completion of assigned work. Payment details such as transaction records, payment status, and timestamps are securely stored for reference and auditing purposes.The admin monitors all financial activities to ensure transparency and accountability across the platform. This module helps maintain trust among users by providing a clear record of earnings and payments while supporting a reliable financial workflow. Although advanced payment gateways may be integrated in the future, the current module focuses on secure payment tracking and record management.

5. **Report**

This module generates detailed reports to support system monitoring and administrative decision-making in the **LearnToEarn** platform. The admin can generate reports related to student registrations, employer participation, job postings, application statistics, interview outcomes, and payment summaries.These reports help the admin analyze system performance, identify trends, and evaluate user engagement across the platform. By maintaining structured records and

providing analytical insights, this module supports transparency, operational improvement, and long-term planning for the LearnToEarn system.

3.3 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 the 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
- Performance

A database is an integrated collection of data and provides centralized access to the data. Usually, the centralized data managing the software is called RDBMS. The main significant difference between RDBMS and other DBMS is the separation of data as seen by the program and data has in direct access to stores device. This is the difference between logical and physical data.

3.3.3.1 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 in new ways. Tuples will be added and deleted. Information stored may undergo updating also. New association may also be added. In such situations 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 form, 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 increases, 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.

Normalization theory is built around the concept of normal forms. A relation is said to be in normal form if it satisfies a certain specified set of constraints. For example, a relation is said to be in first normal form (1NF) if it satisfies the constraint that it contains atomic values only. Thus, every normalized relation is in 1NF.Numerous normal forms have been defined. Codd defined the first three normal forms.

All normalized relations are in 1NF, some 1NF relations are also in 2NF and some 2NFrelations are also in 3NF.2NF relations are more desirable than 1Nf and 3NF are more desirable than 2NF. That is, the database designer should prefer 3NF than 1NF or 2NF.Normalization procedure states that a relation that is in some given normal form can be converted into a set of relations in a more desirable form. We can define this procedure as the successive reduction of a given collection of relations to some more desirable form. 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 loss decomposition".

First Normal Form

A relation is in first normal form (1NF) if and all its attributes are based on single domain. The objective of normalizing 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 1NF and every attribute in 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.

3.3.2 Table Structure

Table is a collection of complete details about a particular subject. These data are saved in rows and Columns. The data of each Row are different units. Hence, rows are called RECORDS and Columns of each row are called FIELDS.

Data is stored in tables, which is available in the backend the items and data, which are entered in the input, form id directly stored in this table using linking of database. We can link more than one table to input forms. We can collect the details from the different tables to display on the output.

There are mainly 13 tables in the project. They are,

1. tbl_login
2. tbl_district
3. tbl_student
4. tbl_employee
5. tbl_subscription
6. tbl_jobcategory
7. tbl_jobposting
8. tbl_application
9. tbl_jobhistory
10. tbl_feedback
11. tbl_notification
12. tbl_report

1. Table: **tbl_login**

   Description: This table is used to store the login information.

Table 3.1 tbl_login

Field Name	Data Type	Constraint	Description of field
id	Int	Primary Key	This field uniquely identifies the user, with values being automatically generated.
username	Varchar(50)	Unique, Not Null	This field stores the username used by the user to log in.
password	Varchar(255)	Not Null	This field stores the hashed password of the user.
role	Varchar(20)	Not Null	This field stores the type of user such as Admin, Employer, or Student.
is_active	Boolean	Not Null	This field indicates whether the user account is active or blocked.

2. Table: **tbl_district**

   Description: This table is used to store the district information.

Table 3.2 tbl_district

Field Name	Data Type	Constraints	Description of the field
district_id	Int	Primary key	This field uniquely identifies the district, with values being automatically generated.
district_name	Varchar(30)	Not null	This field stores the district name.

3. Table: **tbl_student**

Description: This table is used to store the student information.

Table 3.4 tbl_student

Field name	Data type	Constraint	Description of the field
id	Int	Primary Key	This field uniquely identifies the student profile, with values being automatically generated.
user_id	Int	One-to-One, Foreign Key	This field is used to refer the User table.
name	Varchar(100)	Not Null	This field stores the full name of the student.
phone	Varchar(15)	Not Null	This field stores the contact number of the student.
email	Varchar(100)	Unique, Not Null	This field stores the email address of the student.
academic_status	Varchar(200)	Nullable	This field stores the current academic details of the student.
Id_card	Varchar(200)	Not Null	This field stores the current id card details of the student.
skills	Text	Nullable	This field stores the list of skills of the student.

4. Table: **tbl_employee**

Description: This table is used to store the employee information.

Table 3.5 tbl_employee

Field name	Data type	Constraint	Description of the field
id	Int	Primary Key	This field uniquely identifies the employer profile, with values being automatically generated.
user_id	Int	One-to-One, Foreign Key	This field is used to refer the User table.
company_name	Varchar(100)	Not Null	This field stores the name of the employer's company.
contact_person	Varchar(100)	Not Null	This field stores the name of the main contact person (HR).
phone	Varchar(15)	Not Null	This field stores the contact number of the company.
address	Varchar(255)	Not Null	This field stores the physical address of the company.
Verification status	Varchar(255)	Not Null	This field stores the verification statusof the company.

5. Table: **tbl_subscription**

Description: This table is used to store the subscription information.

Table 3.6 tbl_subscription

Field name	Data type	Constraint	Description of the field
id	Int	Primary Key	This field uniquely identifies the subscription, with values being automatically generated.
employer_id	Int	Foreign Key	This field is used to refer the EmployerProfile table.
plan_name	Varchar(50)	Not Null	This field stores the type of subscription plan selected by the employer.
start_date	Date	Not Null	This field stores the date on which the subscription started.
end_date	Date	Not Null	This field stores the date on which the subscription ends.
is_active	Boolean	Not Null	This field indicates whether the subscription is currently active.

6. Table: **tbl_jobcategory**

Description: This table is used to store the job category information.

Table 3.7 tbl_jobcategory

Field name	Data type	Constraint	Description of the field
category_id	Int	Primary Key	This field uniquely identifies the job category, with values being automatically generated.
name	Varchar(100)	Unique, Not Null	This field stores the name of the job category.

7. Table: **tbl_jobposting**

Description: This table is used to store the job posting information.

Table 3.8 tbl_jobposting

Field name	Data type	Constraint	Description of the field
job_id	Int	Primary Key	This field uniquely identifies the job posting, with values being automatically generated.
employer_id	Int	Foreign Key	This field is used to refer the EmployerProfile table.
title	Varchar(100)	Not Null	This field stores the title of the job.
description	Text	Not Null	This field stores the detailed description of the job.
category_id	Int	Foreign Key	This field is used to refer the JobCategory table.
work_mode	Varchar(20)	Not Null	This field stores whether the job is Online or Offline.
location_id	Int	Foreign Key	This field is used to refer the Location table.
duration	Varchar(50)	Not Null	This field stores the duration or period of the job.
posted_date	Date	Not Null	This field stores the date on which the job was posted.
status	Varchar(20)	Not Null	This field stores the verification status of the job such as Pending, Verified, or Rejected.
is_filled	Boolean	Not Null	This field indicates whether the job position has been filled.

8. Table: **tbl_application**

Description: This table is used to store the job application details.

Table 3.9 tbl_application

Field name	Data type	Constraint	Description of the field
application_id	Int	Primary Key	This field uniquely identifies the application, with values being automatically generated.
job_id	Int	Foreign Key	This field is used to refer the JobPosting table.
student_id	Int	Foreign Key	This field is used to refer the StudentProfile table.
applied_date	Date	Not Null	This field stores the date on which the student applied for the job.
status	Varchar(20)	Not Null	This field stores the current application status such as Applied, Shortlisted, Selected, or Rejected.
interview_scheduled	Datetime	Nullable	This field stores the date and time of the scheduled interview.

9. Table: **tbl_jobhistory**

Description: This table is used to store the job history details.

Table 3.10 tbl_jobhistory

Field name	Data type	Constraint	Description of the field
history_id	Int	Primary Key	This field uniquely identifies the completed job record, with values being automatically generated.
student_id	Int	Foreign Key	This field is used to refer the StudentProfile table.
job_id	Int	Foreign Key	This field is used to refer the JobPosting table.
completion_date	Date	Not Null	This field stores the date on which the job was completed.

10. Table: **tbl_feedback**

Description: This table is used to store the feedback details.

Table 3.10 tbl_feedback

Field name	Data type	Constraint	Description of the field
id	Int	Primary Key	This field uniquely identifies the feedback or complaint, with values being automatically generated.
sender_id	Int	Foreign Key	This field is used to refer the User table and identifies who submitted the complaint.
recipient_id	Int	Foreign Key	This field is used to refer the User table and identifies whom the complaint is about.
subject	Varchar(100)	Not Null	This field stores the subject of the complaint or feedback.
message	Text	Not Null	This field stores the detailed complaint or feedback message.
submitted_at	Datetime	Not Null	This field stores the date and time when the complaint was submitted.
status	Varchar(20)	Not Null	This field stores the status of the complaint such as New, In Progress, or Resolved.

11. Table: **tbl_notification**

Description: This table is used to store the notification details.

Table 3.12 tbl_notification

Field name	Data type	Constraint	Description of the field
id	Int	Primary Key	This field uniquely identifies the notification, with values being automatically generated.
user_id	Int	Foreign Key	This field is used to refer the User table.
message	Varchar(255)	Not Null	This field stores the content of the notification.
timestamp	Datetime	Not Null	This field stores the date and time when the notification was sent.
is_read	Boolean	Not Null	This field indicates whether the notification has been read by the user.

12. Table: **tbl_report**

Description: This table is used to store the report details.

Table 3.13 tbl_report

Field name	Data type	Constraint	Description of the field
id	Int	Primary Key	This field uniquely identifies the report, with values being automatically generated.
report_type	Varchar(100)	Not Null	This field stores the type of report such as User Statistics or Job Activity.
generation_date	Date	Not Null	This field stores the date on which the report was generated.
file_path	Varchar(255)	Not Null	This field stores the file path of the generated report such as PDF or Excel.

3.3.3 Data Flow Diagrams

3.3.3.1 Introduction to Data Flow Diagrams

Data Flow Diagram is a network that describes the flow of data and processes that change, or transform, data throughout the system. This network is constructed by use a set of symbols that do not imply a physical implementation. It is a graphical tool for structured analysis of the system requirements. DFD models a system by using external entities from which data flows to a process, which transforms the data and creates, output-data-flows which go to other processes or external entities or files. Data in files may also flow to processes as inputs.

There are various symbols used in a DFD. Bubbles represent the processes. Named arrows indicate the data flow. External entities are represented by rectangles. Entities supplying data are known as sources and those that consume data are called sinks. Data are stored in a data store by a process in the system. Each component in a DFD is labelled with a descriptive name. Process names are further identified with a number.

The Data Flow Diagram shows the logical flow of a system and defines the boundaries of the system. For a candidate system, it describes the input (source), outputs (destination), database (files) and procedures (data flow), all in a format that meet the user's requirements.

The main merit of DFD is that it can provide an overview of system requirements, what data a system would process, what transformations of data are done, what files are used, and where the results flow.

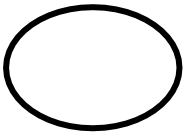
This network is constructed by use a set of symbols that do not imply a physical implementation. It is a graphical tool for structured analysis of the system requirements. DFD models a system by using external entities from which data flows to a process, which transforms the data and creates, output-data-flows which go to other processes or external entities or files. External entities are represented by rectangles. Entities supplying data are known as sources and those that consume data are called sinks. Data are stored in a data store by a process in the system. It is a graphical tool for structured analysis of the system requirements. DFD models a system by using external entities from which data flows to a process, which transforms the data and creates, output-dataflows which go to other processes or external entities or files. Data in files may also flow to processes as inputs.

Rules for constructing a Data Flow Diagram

- Arrows should not cross each other
- Squares, circles and files must bear names
- Decomposed data flow squares and circles can have same time
- Choose meaningful names for data flow
- Draw all data flows around the outside of the diagram

Basic Data Flow Diagram Symbols

Table 3.14 Data Flow Diagram Symbols

	A data flow is a route, which enables packets of data to travel from one point to another. Data may flow from a source to a process and from data store or process. An arrow line depicts the flow, with arrow head pointing in the direction of the flow.
	Circles stands for process that converts data in to information. A process represents transformation where incoming data flows are changed into outgoing data flows.
	A data store is a repository of data that is to be stored for use by a one or more process may be as simple as buffer or queue or sophisticated as relational database. They should have clear names. If a process merely uses the content of store and does not alter it, the arrowhead goes only from the store to the process. If a process alters the details in the store, then a double-headed arrow is used.
	A source or sink is a person or part of an organization, which enters or receives information from the system, but is considered to be outside the contest of data flow model.

3.3.3.2 Data Flow Diagram

Each component in a DFD is labelled with a descriptive name. Process names are further identified with number. Context level DFD is draw first. Then the process is decomposed into several elementary levels and is represented in the order of importance. A DFD describes what data flow (logical) rather than how they are processed, so it does not depend on hardware, software, and data structure or file organization.

A DFD methodology is quite effective; especially when the required design.

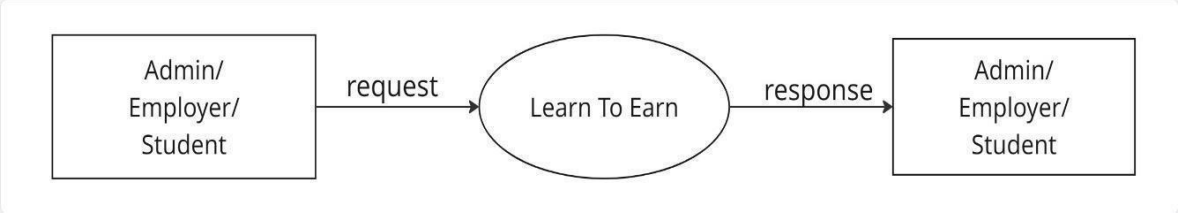


Fig 3.1 Context Level Data Flow Diagram for Learn To Earn

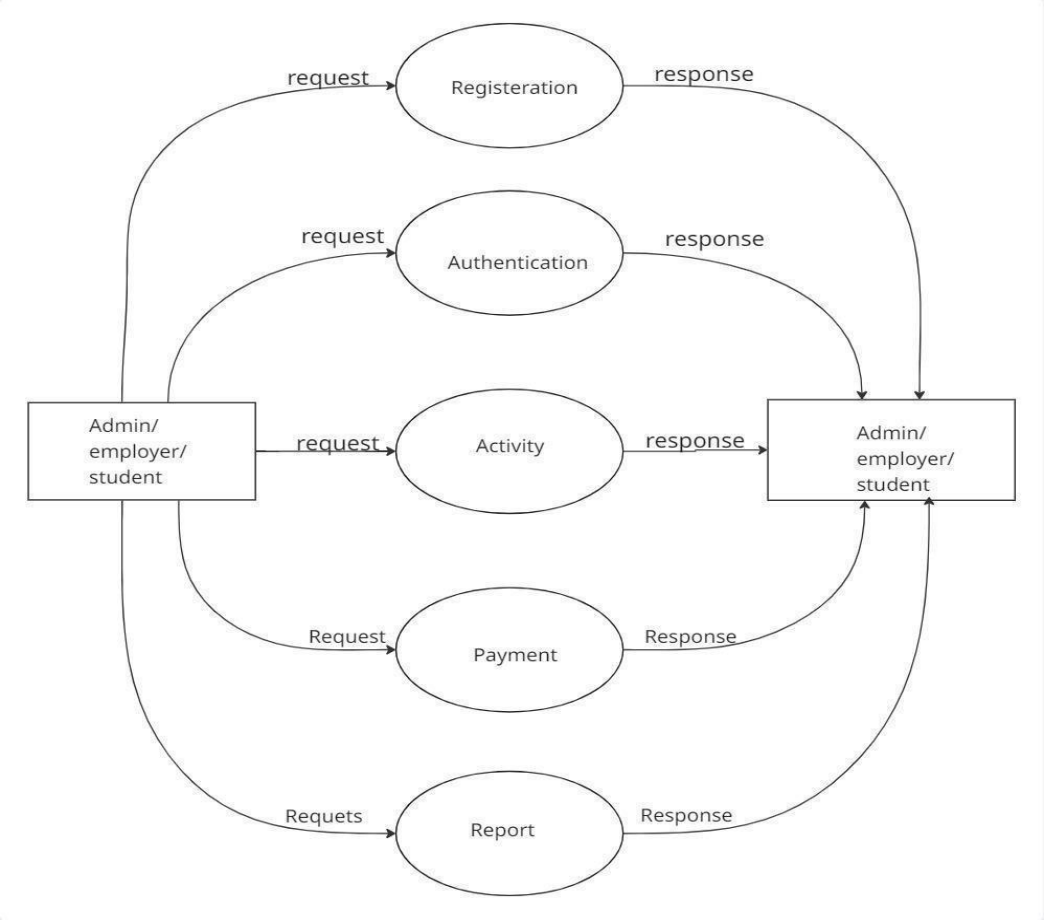


Fig 3.2 First Level Data Flow Diagram for Job Sphere

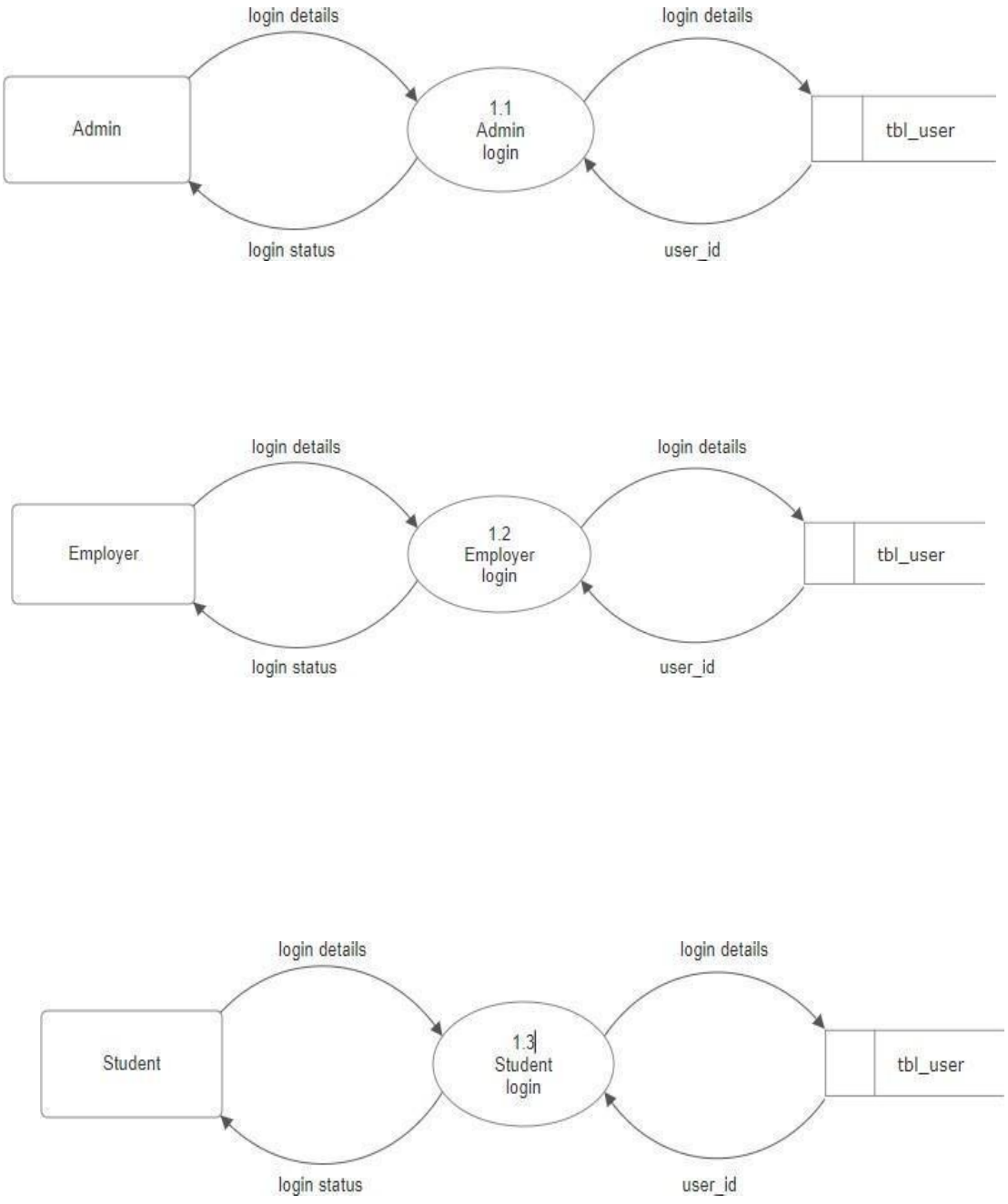
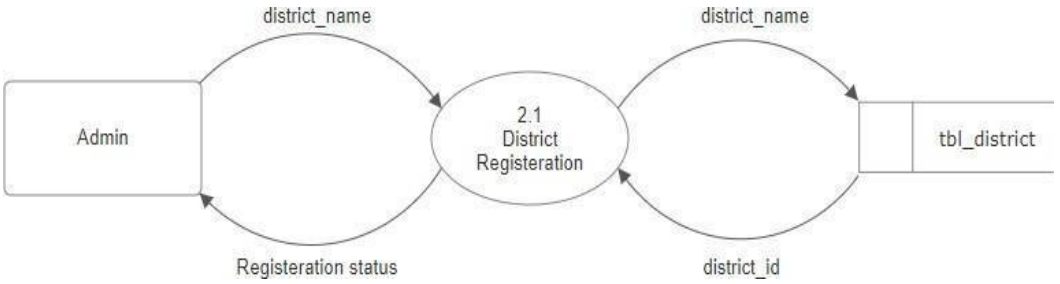
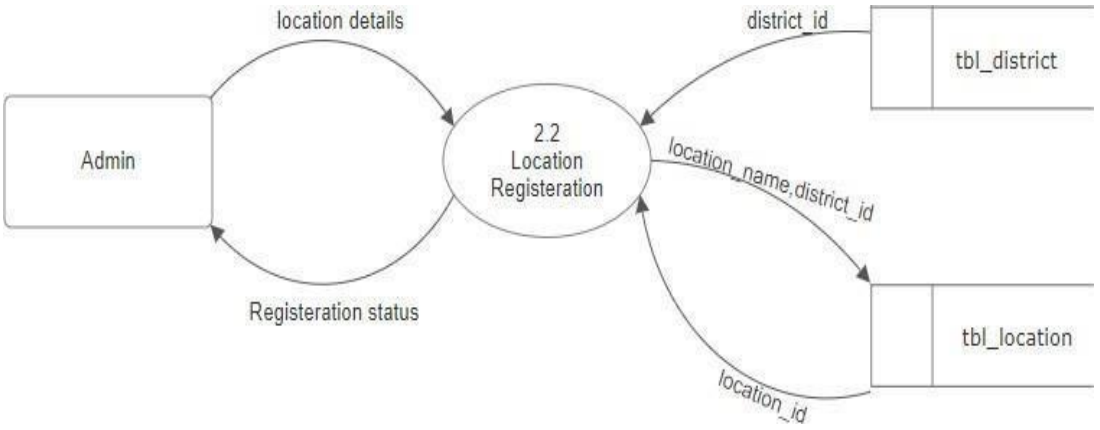
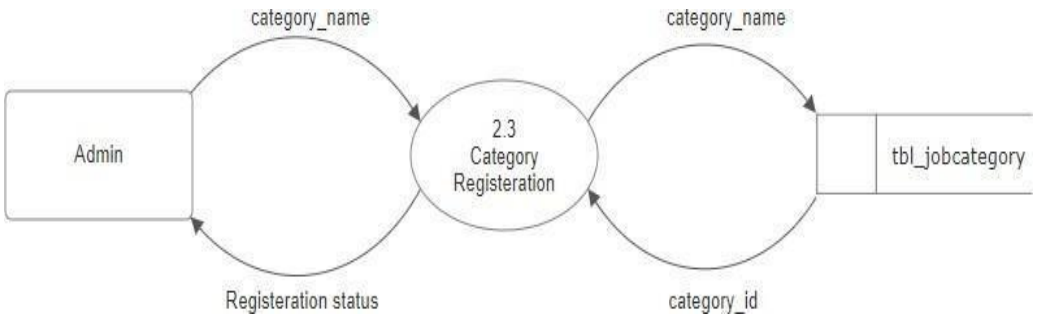
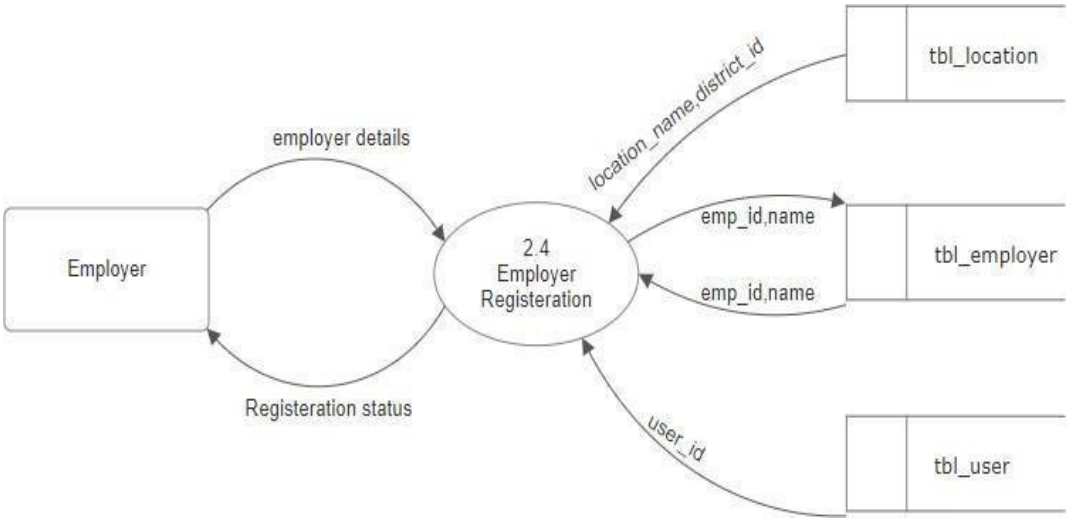


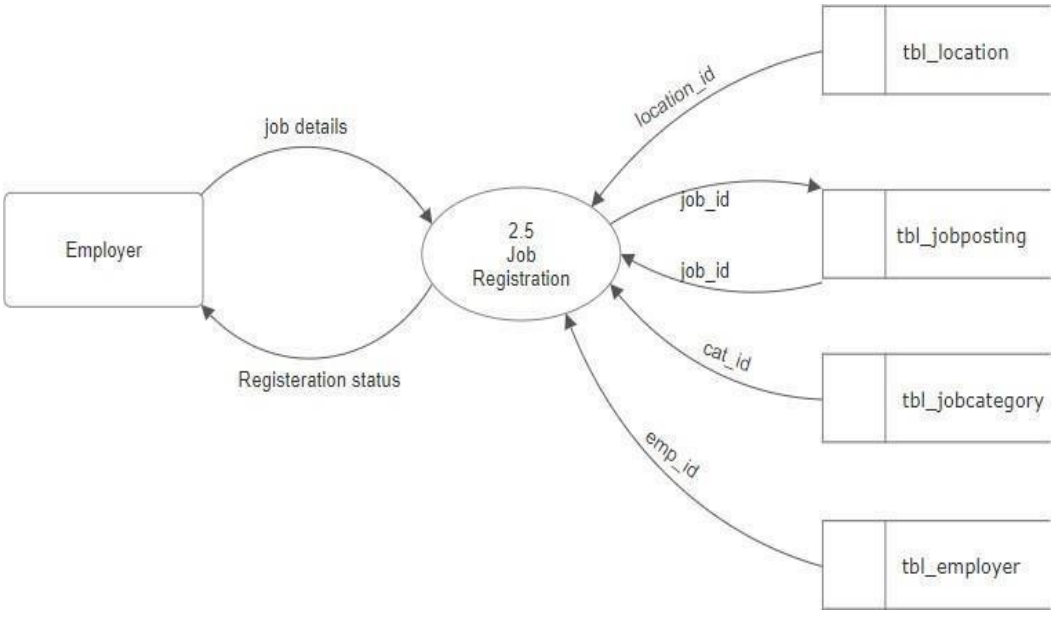
Fig 3.3 Second Level DFD for Learn to earn (Authentication)


district_name
district_name
Admin
2.1
District
Registeration
tbl_district
Registeration status
district_id


location details
district_id
tbl_district
Admin
2.2
Location
Registeration
location_name,district_id
tbl_location
Registeration status
location_id


category_name
category_name
Admin
2.3
Category
Registeration
tbl_jobcategory
Registeration status
category_id


tbl_location
location_name,district_id
employer details
Employer
2.4
Employer
Registeration
emp_id,name
emp_id,name
tbl_employer
Registeration status
user_id
tbl_user


tbl_location
location_id
job details
Employer
2.5
Job
Registration
job_id
job_id
tbl_jobposting
cat_id
tbl_jobcategory
Registeration status
emp_id
tbl_employer

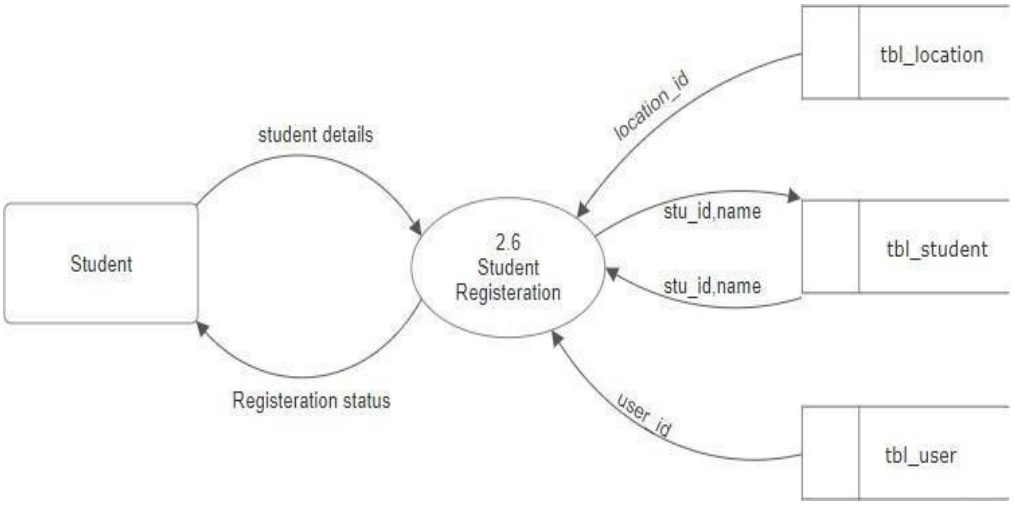
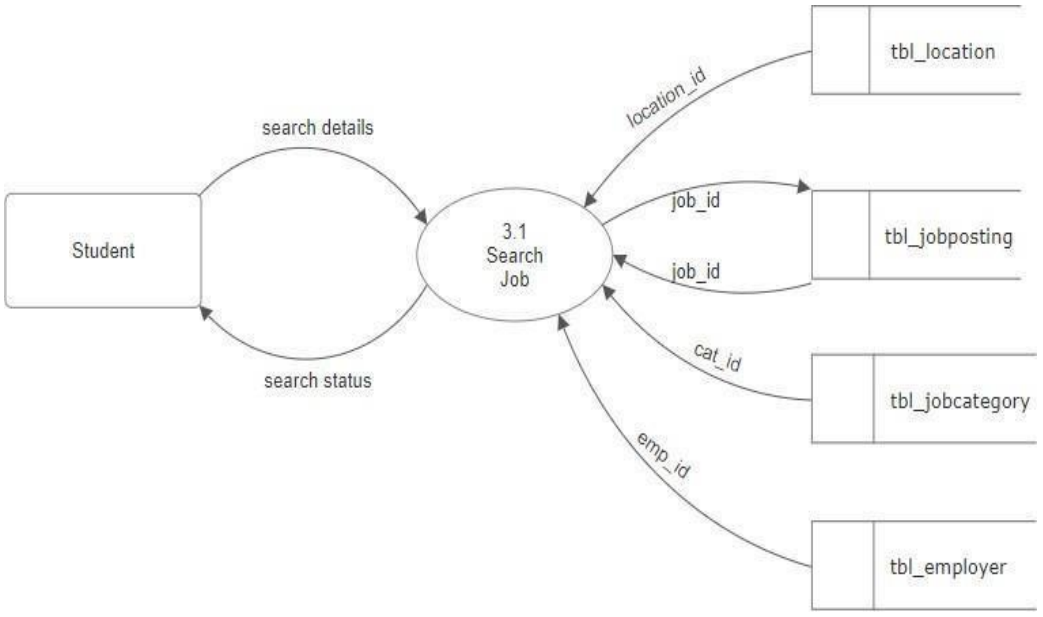


Fig 3.4 Second Level DFD for Learn To Earn (Registration)



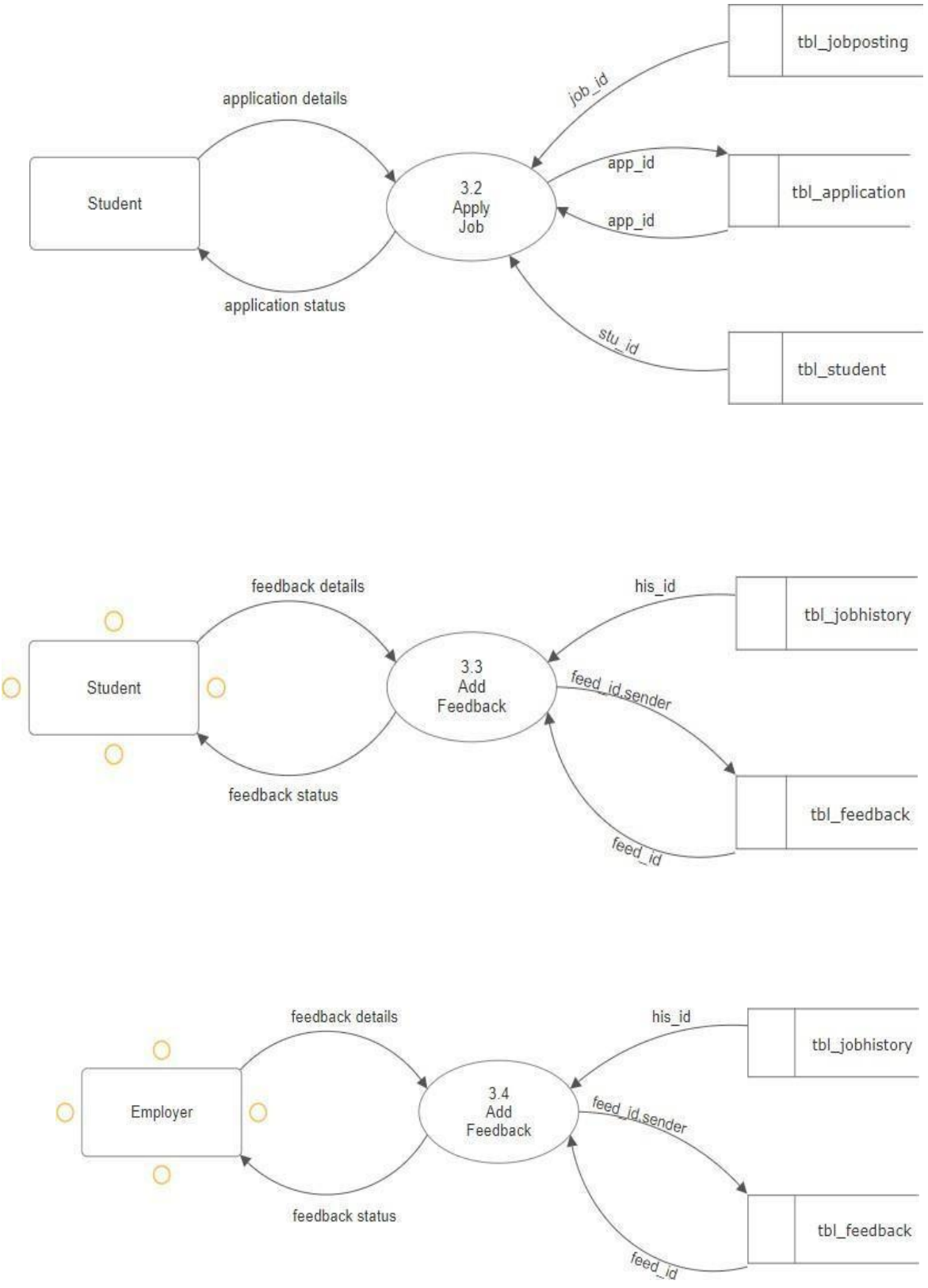


Fig 3.5 Second Level DFD for Learn To Earn (Activity)

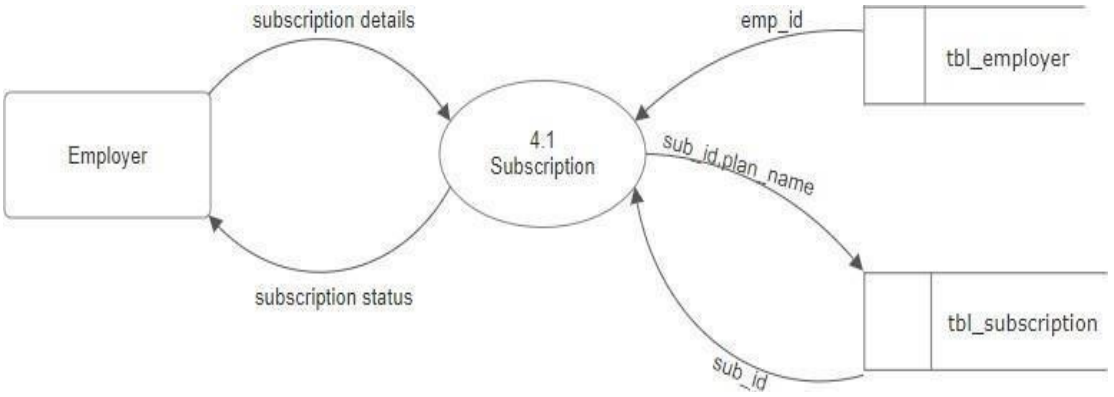
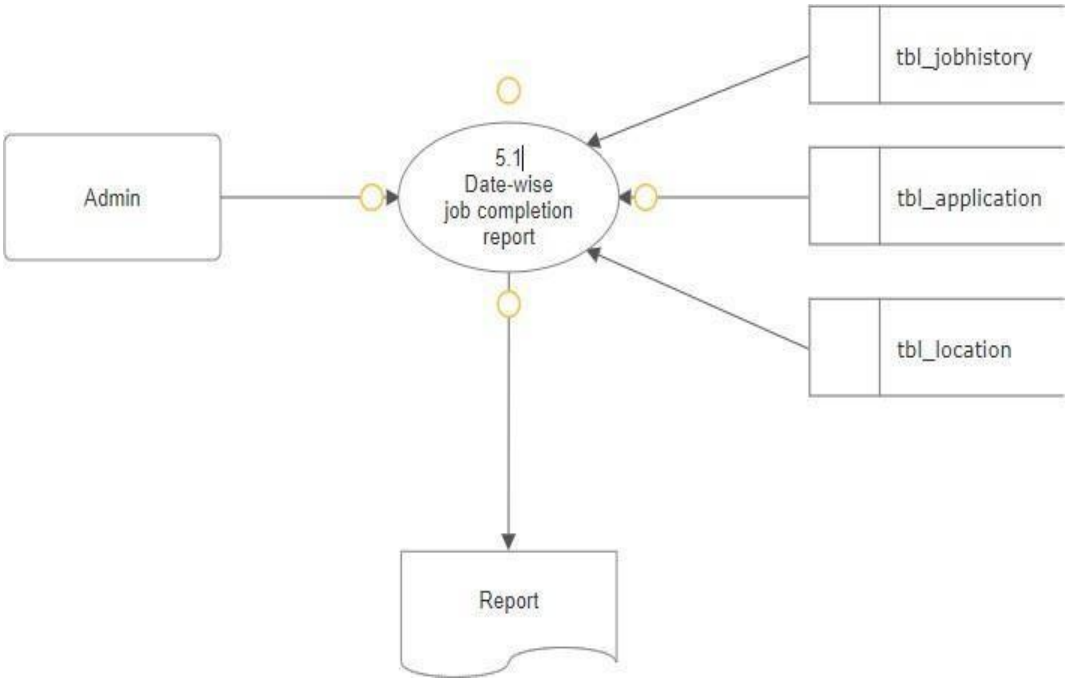


Fig 3.6 Second Level DFD for Learn To Earn (Payment)



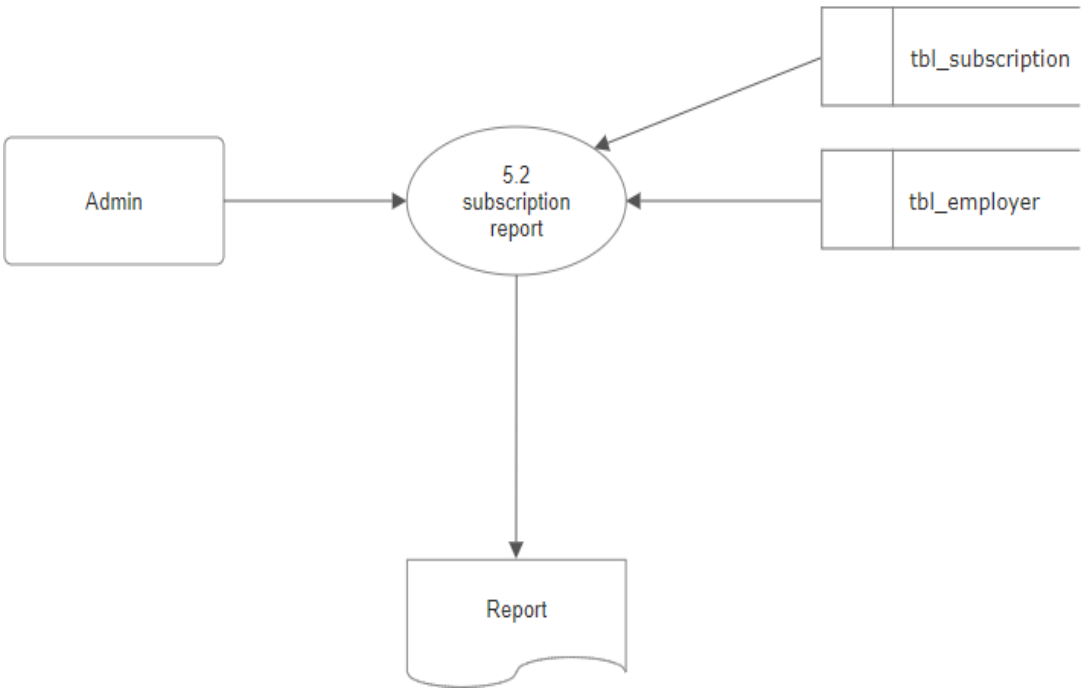


Fig 3.7 Second Level DFD for Learn To Earn (Report)