

A PROJECT REPORT ON
STUDENT MANAGEMENT AND MONITORING
SYSTEM USING DJANGO

**A Mini Project Submitted to Jawaharlal Nehru Technological University, Kakinada in Partial
fulfillments of Requirements for the Award of the Degree of**

BACHELOR OF TECHNOLOGY IN
COMPUTER SCIENCE AND ENGINEERING



Submitted By

M.NITHIN	(19KT1A05A2)
M.ANUDEEP	(19KT1A05A6)
P.AMULYA	(19KT1A05A9)
S.RAMPRASAD	(19KT1A05C1)

Under the Esteemed Guidance of

Mr. A.CHANDRAMOULI ,M.Tech ,(Ph.D)

Associate Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
POTTI SRIRAMULU CHALAVADI MALLIKARJUNA RAO
COLLEGE OF ENGINEERING & TECHNOLOGY
(Approved by AICTE New Delhi, Affiliated to JNTU-Kakinada)
KOTHAPET, VIJAYAWADA-520001, A.P
2019-2023

POTTI SRIRAMULU CHALAVADI MALLIKHARJUNARAO

COLLEGE OF ENGINEERING & TECHNOLOGY

KOTHAPET, VIJAYAWADA-520001.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the project work entitled “**STUDENT MANAGEMENT AND MONITORING SYSTEM USING DJANGO**” is a bonafide work carried out by **Mr.M.NITHIN (19KT1A05A2)**, **Mr.M.ANUDEEP (19KT1A05A6)**, **Ms. P.AMULYA (19KT1A05A9)**, **Mr.S.RAMPRASAD (19KT1A05C1)**. Fulfillment for the for the completion of mini project in **COMPUTER SCIENCE AND ENGINEERING** of Jawaharlal Nehru Technological University, Kakinada during the year **2021-2022**. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the above degree

Project Guide

Head of the Department

External Examiner

ACKNOWLEDGEMENT

We owe a great many thanks to a great many people who helped and supported and suggested us in every step.

We are glad for having the support of our principal **Dr. K. Sri Rama Krishna** who inspired us with his words filled with dedication and discipline towards work.

We express our gratitude towards **Dr. A. Pathanjali Sastri, Professor & HoD of CSE** for extending his support through training classes which had been the major source to carry out our project.

We are very much thankful to **Mr.A.Chandramouli, M.Tech, (Ph.D)** Associate Professor, Guide of our project for guiding and correcting various documents of ours with attention and care. She has taken pain to go through the project and make necessary corrections as and when needed.

Finally we thank one and all who directly and indirectly helped us to complete our project successfully.

Project Associates

M.NITHIN (19KT1A05A2)

M.ANUDEEP (19KT1A05A6)

P.AMULYA (19KT1A05A9)

S.RAMPRASAD (19KT1A05C1)

DECLARATION

This is to declare that the project entitled “STUDENT MANAGEMENT AND MONITORING SYSTEM USING DJANGO” submitted by us in the partial fulfillment of requirements for the award of the degree of **Bachelor of Technology in Computer Science & Engineering** in **Potti Sriramulu Chalavadi Mallikarjuna Rao College of Engineering and Technology**, is bonafide record of project work carried out by us under the supervision and guidance of **Mr.A.Chandramouli, M.Tech, (Ph.D) Associate Professor of CSE**. As per our knowledge, the work has not been submitted to any other institute or universities for any other degree.

Project Associates

M.NITHIN (19KT1A05A2)

M.ANUDEEP (19KT1A05A6)

P.AMULYA (19KT1A05A9)

S.RAMPRASAD (19KT1A05C1)

ABSTRACT

Students from the main part of any institution's concerns with data storing. But the institutions find it difficult to keep details of so many students of the organization just in one stretch. It will involve a lot of pwn paperwork. Sometimes there will be some huge heap of files bundled up and kept together in some corner of the office. If you want any information regarding the particular student then it can be obtained by just entering the roll number of the name of the student to be searched. This Student Management system will make the work of storing the data in an organized way. Class Monitoring System is about monitoring the students' performance, marks, grades, and personal details by collecting student data. Individual students' data are collected and stored in Database for future use. Class Teacher is provided with the admin details to log in to the database for the checking of the data or to change the data. Once the student enters the data into the registration form it is stored in the Database. Access to the database is held by only the Class teacher. He is the only one who can access and can manipulate the data in the database. From this project, we can create a Student Details collection web page, and using Django the collected data from the students is stored in the database and the credentials of the database or access to the data in the database are provided for Specific class teachers of the class.

CONTENTS

SL.NO	TOPIC	PAGE NO
1.	INTRODUCTION	1
	1.1 Brief Overview of the project	1
	1.1.1 Scope	1
	1.1.2 Purpose	1
	1.1.3 Objective of the study	1
	1.1.4 Literature Survey	2
	1.1.5 Existing System	4
	1.1.6 Proposed System	2
2.	SYSTEM ANALYSIS	6
	2.1 Feasibility Analysis	6
	2.1.1 Technical Feasibility	6
	2.1.2 Economic Feasibility	6
3.	SYSTEM DESIGN	8
	3.1.1 About System Design	8
	3.1.2 Initialize Design Definition	8
	3.1.3 Establish Design Characteristics	9
	3.1.4 Assess alternatives for system elements	9
	3.1.5 Manage the design	9
	3.2. System Architecture	9
	3.3. Data Flow Diagram	11
	3.3.1 Importance of data flow diagrams	11
	3.3.2 Different Types of DFD's	14
	3.3.3 Data flow Diagrams	14
	3.4. Types of Data flow diagram	16
	3.4.1. Context level DFD	16

	3.4.2. Level 1 DFD	17
	3.4.3. Level 2 DFD	18
	3.4.4. Making Levels	20
	3.5. DFD Heirarchy	21
	3.6. Student Data Entry	21
4.	DJANGO	29
	4.1 About Django	29
	4.2 Components	29
	4.3 Extensibility	30
	4.4 Server Arrangements	30
	4.5 Features of Django	31
	4.5.1 Complete	31
	4.5.2 Versatile	31
	4.5.3 Secure	31
	4.5.4 Scalable	32
	4.5.5 Maintainable	32
	4.5.6 Portable	32
5.	DJANGO IMPLEMENTATION	35
	5.1 Installing Django	35
6.	SYSTEM IMPLEMENTATION	44
	6.1 Source code of Project	44
	6.1.1 wsgi.py	44
	6.1.2 urls.py	44
	6.1.3 asgi.py	44
	6.1.4 settings.py	44
	6.2 Sourec code for User app	46
	6.2.1 views.py	46
	6.2.2 models.py	47
	6.2.3 apps.py	48

	6.2.4 admins.py	48
	6.2.5 Template.py	48
	6.3 Results	52
7.	TESTING	58
	7.1 About Testing	58
	7.2 Testing Methods	58
	7.2.1 Static vs Dynamic Testing	58
	7.2.2 The Box Approach	59
	7.2.2.1 White Box Testing	59
	7.2.2.2 Black-Box Testing	59
	7.2.3 Testing Levels	60
	7.2.3.1 Unit Testing	60
	7.2.3.2 Integration Testing	61
	7.2.3.3 Component Interface Testing	61
	7.2.3.4 System Testing	61
	7.3 Validation and Verification	61
	7.3.1 Verification in Software Testing	62
	7.3.2 Validation in Software Testing	62
8.	CONCLUSION	64
9.	REFERENCES	66
10.	APPENDIX	69

LIST OF FIGURES

S.NO	Fig. NO	NAME OF THE FIGURE	PAGE NO
1	3.2.1	System Architecture	10
2	3.3.1.1	Data Flow Diagrams	15
3	3.4.1.1	Level 0 DFD	16
4	3.4.2.1	Level 1 DFD	17
5	3.4.3.1	Level 2 DFD	19
6	3.6.1	Level 1 of Student Data Entry	21
7	3.6.2	Marks Entry into Database	22
8	3.6.3	Marks with Student Database	22
9	3.6.3	Student Registration Form(a)	23
10	3.6.5	Student Registartion Form(b)	24
11	3.6.6	Student Registartion Form(c)	25
12	3.6.7	Student Registartion Form(d)	26
11	3.6.8	Edit Student Details	27
12	5.1.1	Opening Powershell	34
13	5.1.2	Verifying Python	36
14	5.1.3	Creating a Virtual Environment	37
15	5.1.4	Activating the Virtual Environment	38
16	5.1.5	Installing Django	39
17	5.1.6	Running the Development Server	41
18	5.1.7	Opening page in browser	42
19	6.3.1	Entering Student Details(a)	52
20	6.3.2	Entering Student Details(b)	53
21	6.3.3	Running Server	54
22	6.3.4	Admin Login	54

23	6.3.5	Admin Homepage	55
24	6.3.6	Viewing of Entered Student Details(a)	55
25	6.3.7	Viewing of Entered Student Details(b)	56
26	6.3.8	Logging out from Admin Page	56

STUDENT MANAGEMENT AND MONITORING SYSTEM

INTRODUCTION

1. INTRODUCTION

1.1 BREIF OVERVIEW OF THE PROJECT

Schools and Universities are the foundation of knowledge and an educational body on which students rely upon. Therefore, they need to maintain a proper database of its students to keep all the updated records and easily share information with students. Most schools and Universities count on an advanced software tool known as '**Student Information System (SIS)**' to keep all their student records and administrative operations including, examinations, attendance, and other activities. Over the recent years, the performance and efficiency of the education industry have been enhanced by using the **Student Management System**. This tool has productively taken over the workload of the admin department with its well-**organized, easy, and reliable online school management software**. From this project we implemented a webpage which collects the student information and stores it in the database for the easy access and monitoring of the student data.

1.1.1 Scope

The main aim of this project is to easy access of the student details and monitoring about their performance towards Education and also self development

1.1.2 Purpose

Project is mainly useful for the following details

- To access the student information easily and quickly.
- Monitoring the student regularly and motivating the student.

1.1.3 Objective of the study

Student Management System is software which is helpful for students as well as the school authorities. In the current system all the activities are done manually. It is very time consuming and costly. Our Student Management System deals with the various activities related to the students.

There are mainly 3 modules in this software

User module

Student Module

Mark management

In the Software we can register as a user and user has of two types, student and administrator.

Administrator has the power to add new user and can edit and delete a user. A student can register a user and can add edit and delete his profile. The administrator can add edit and delete marks for the student. All the users can see the marks.

1.1.4 LITERATURE SURVEY

Toward a Student Information System for Sebha University This paper basically focuses on providing a simple interface for the easy collation and maintenance of all manner of student information. The creation and management of accurate, up-to- date information regarding students' academic careers is critical students and for the faculties and administration of Sebha University in Libya and for any other educational institution. A student information system deals with all kinds of data from enrollment to graduation, including program of study, attendance record, payment of fees and examination results to name but a few. All these data need to be made available through an onlineinterface.

A Study of Student Information Management Software This paper focuses on providing information to support the operation, management and decision-making functions of enterprises or organizations. In the face of huge amount of information, it is required to possess the student information management system to improve the efficiency of student management. Through this system, the standardized management, scientific statistics and fast query of student information can be realized, and thus the workload of management can be reduced. In this paper, a typical student information management system will be established to realize the systematization, standardization and automation of student information relationship.

Web Based Student Information System This paper focuses on simple interface for maintenance of student information. The creation and management of accurate, up-to- date information regarding a student's academic career is critically important in the university as well as colleges. Student information system deals with all kind of student details, academic related reports, college details, course details, curriculum, batch details, placement details and other resource related details too. It tracks all the details of a student which can be used for all reporting purpose, tracking of attendance, progress in the course, completed semesters, years. Different reports and Queries can be generated based on vast options related to students, batch, course, faculty, exams, semesters, certification and even for the entire college.

In India there are many academic institutions. But very few institutions are modernized and use software to manage their day to day work. In a city like Bengaluru there are around 1000 schools,

STUDENT MANAGEMENT AND MONITORING SYSTEM

more than 300 pre-university colleges and degree colleges. Most of these academic institutions still relay on traditional way of management which mainly involves paper-work, much of human effort.

The students, who are admitted to those institutions which are dependent on traditional way of managing things, have to struggle a lot just to get a certificate or any other documents. Also the administrations face difficulty in maintaining all the records, tracking the records and fetching the record of their interest in time. The administrations of those institutions also have to employ a number of employees just to maintain the records required to manage and support their daily work.

Some of the universities like PESIT and Christ University in Bengaluru have their own web application to address the previously mentioned issues.

The web application that is being used by these and many other institutions have the following features and functionalities such as, Login/Sign Up, Dashboard, Viewing of results, attendance, courses, time table, assignments and students progress, upload/download documents and notification.

1.1.5 Existing system

System Analysis is a detailed study of the various operations performed by a system and their relationships within and outside of the system. Here the key question is- what all problems exist in the present system? What must be done to solve the problem? Analysis begins when a user or manager begins a study of the program using existing system. During analysis, data collected on the various files, decision points and transactions handled by the present system. The commonly used tools in the system are Data Flow Diagram, interviews, etc. Training, experience and common sense are required for collection of relevant information needed to develop the system. A good analysis model should provide not only the mechanisms of problem understanding but also the frame work of the solution. Thus, it should be studied thoroughly by collecting data about the system. Then the proposed system should be analyzed thoroughly in accordance with the needs

1.1.6 PROPOSED SYSTEM

In our proposed system we have the provision for adding the details of the students by themselves. So, the overhead of the school authorities and the teachers is become less. Another advantage of the system is that it is very easy to edit the details of the student and delete a student when it found unnecessary. The marks of the student are added in the database and so students can also view the marks whenever they want.

Our proposed system has several advantages

- User friendly interface
- Fast access to database
- Less error
- More Storage Capacity
- Search facility
- Look and Feel Environment
- Quick transaction

All the manual difficulties in managing the student details in a school or college have been rectified by implementing computerization.

SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

2.1 FEASIBILITY ANALYSIS

Whatever we think need not be feasible. It is wise to think about the feasibility of any problem we undertake. Feasibility is the study of impact, which happens in the organization by the development of a system. The impact can be either positive or negative. When the positives nominate the negatives, then the system is considered feasible. Here the feasibility study can be performed in two ways such as technical feasibility and Economical Feasibility.

2.1.1 TECHNICAL FEASIBILITY:

We can strongly say's that it is technically feasible, since there will not be much difficulty in getting required resources for the development and maintaining the system as well. All the resources needed for the development of the software as well as the maintenance of the same is available in the organization here we are utilizing the resources which are available already.

2.1.2 ECONOMIC FEASIBILITY:

Development of this application is highly economically feasible. The organization needed not spend much money for the development of the system already available. The only thing is to be done is making an environment for the development with an effective supervision. I f we are doing so, we can attain the maximum usability of the corresponding resources. Even after the development, the organization will not be in condition to invest more in the organization. Therefore, the system is economically feasible

SYSTEM DESIGN

3. SYSTEM DESIGN

3.1.1.ABOUT SYSTEM DESIGN

System design is the process of designing the elements of a system such as the architecture, modules, components, the different interfaces of those components and the data that goes through that system.

The purpose of the System Design process is to provide sufficient detailed data and information about the system and its system elements to enable the implementation consistent with architectural entities as defined in models and views of the system architecture.

Elements of a System:

Architecture - This is the conceptual model that defines the structure, behavior and more views of a system. We can use flowcharts to represent to illustrate the architecture.

Modules - This are components that handle one specific task in a system. A combination of the modules makes up the system.

Components - This provides a particular function or group of related functions. They are made up of modules.

Interfaces - This is the shared boundary across which the components of the system exchange information and relate.

Data - This management of the information and data flow.

3.1.2.Initialize design definition

- Plan for and Identify the technologies that will compose and implement the systems elements and their physical interfaces.
- Determine which technologies and system elements have a risk to become obsolete, or evolve during the operation stage of the system. Plan for their potential replacement.
- Document the design definition strategy, including the need for and requirements of any enabling systems, products, or services to perform the design.

3.1.3. Establish design characteristics

- Define the design characteristics relating to the architectural characteristics and check that they are implementable.
- Define the interfaces that were not defined by the System Architecture process or that need to be refined as the design details evolve.
- Define and document the design characteristics of each system element

3.1.4. Assess alternatives for obtaining system elements

- Assess the design options
- Select the most appropriate alternatives.
- If the decision is made to develop the system element, rest of the design definition process and the implementation process are used. If the decision is to buy or reuse a system element, the acquisition process may be used to obtain the system element.

3.1.5. Manage the design

- Capture and maintain the rationale for all selections among alternatives and decisions for the design, architecture characteristics.
- Assess and control the evolution of the design characteristics.

3.2 SYSTEM ARCHITECTURE

A system architecture or systems 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 and behaviors of the system.

An architecture description also indicates how non functional requirements will be satisfied. For example:

- Specification of system/component performance. For example, data throughput and response times as a function of concurrent users.
- **Consideration of scalability.** For example, can an air traffic control system designed to manage 100 aircraft be extended to manage 1000 aircraft?
- **System availability.** For example, elements of the design that enable a system to operate 24/7.

- **Safety integrity.** Elements of the design that reduce the risk that the system will cause (or allow causation of) harm to property and human beings.
- **Fault tolerance.** Elements of the design that allow the system to continue to operate if some components fail (e.g. no single point of failure).
- **Consideration of product evolution.** The facility for individual components to be modified or dynamically reconfigured without the need for major modification of the system as a whole. Further, the ability to add functionality with new components in a cost effective manner.
- **Consideration of the emergent qualities of the system as a whole** when components are assembled and operated by human beings. For example, can the missile launch system be effectively operated in a high stress combat situation

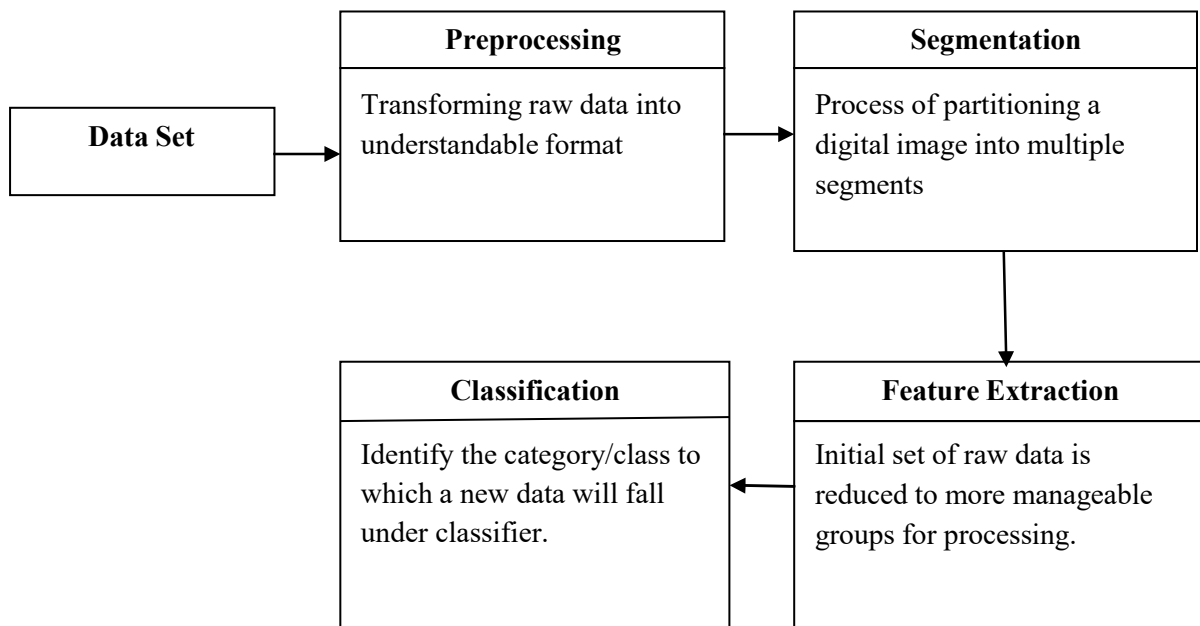


Fig 3.2.1: System Architecture

3.3 DATA-FLOW DIAGRAM

A data-flow diagram (DFD) is a way of representing a flow of a data of a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented by flowchart systems. For each data flow, at least one of the endpoints (source and / or destination) must exist in a process. The refined representation of a process can be done in another data-flow diagram, which subdivides this process into sub-processes.

The data-flow diagram is part of the structured-analysis modeling tools. When using UML, the activity diagram typically takes over the role of the data-flow diagram. A special form of data-flow plan is a site-oriented data-flow plan.

Data-flow diagrams can be regarded as inverted Petri nets, because places in such networks correspond to the semantics of data memories. Analogously, the semantics of transitions from Petri nets and data flows and functions from data-flow diagrams should be considered equivalent.

3.3.1 Importance of Data Flow Diagrams

DFDs are mostly used by systems designers to simplify designing management and decision support systems. Software design processes, such as DFDs simplify and reduce time consuming complex coding routines; not to mention business management documentation processes. Research on DFDs focuses on new measures and criteria for their manifestations and implementations, in general, not their importance for specifically designing data structures, and other related systems.

There is no literature specifically emphasizing the importance of DFDs to the design of data structure. Most related literature stressed the value of Unified Modeling Languages UML, and Systems Engineering Modeling

Processes to designing decision and artificial support systems and other management applications, as a whole. Data structure is part of the process that leads to designing prototypes of all types of software applications.

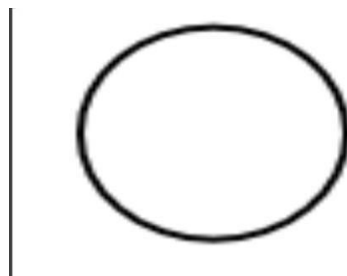
The process of identifying primary data keys and their relationship to other elements within the data structure as a whole. This segment of the design procedure acts as a guide that relat

the elements of a data-structure to the function of that data in handling the specific coding of the related routine.

DFD consists of processes, flows, warehouses, and terminators. There are several ways to view these DFD components.

Process

The process (function, transformation) is part of a system that transforms inputs to outputs. The symbol of a process is a circle, an oval, a rectangle or a rectangle with rounded corners (according to the type of notation). The process is named in one word, a short sentence, or a phrase that is clearly to express its essence.



Process

Data Flow

Data flow (flow, dataflow) shows the transfer of information (sometimes also material) from one part of the system to another. The symbol of the flow is the arrow. The flow should have a name that determines what information (or what material) is being moved. Exceptions are flows where it is clear what information is transferred through the entities that are linked to these flows. Material shifts are modeled in systems that are not merely informative. Flow should only transmit one type of information (material). The arrow shows the flow direction (it can also be bi-directional if the information to/from the entity is logically dependent - e.g. question and answer). Flows link processes, warehouses and terminators.



Warehouse

The warehouse (datastore, data store, file, database) is used to store data for later use. The symbol of the store is two horizontal lines, the other way of view is shown in the DFD Notation. The name of the warehouse is a plural noun (e.g. orders) - it derives from the input and output streams of the warehouse. The warehouse does not have to be just a data file, for example, a folder with documents, a filing cabinet, and optical discs. Therefore, viewing the warehouse in DFD is independent of implementation. The flow from the warehouse usually represents the reading of the data stored in the warehouse, and the flow to the warehouse usually expresses data entry or updating (sometimes also deleting data). Warehouse is represented by two parallel lines between which the memory name is located (it can be modelled as a UML buffer node).



Warehouse

Terminator

The Terminator is an external entity that communicates with the system and stands outside of the system. It can be, for example, various organizations (eg a bank), groups of people (e.g. customers), authorities (e.g. a tax office) or a department (e.g. a human-resources department) of the same organization, which does not belong to the model system. The terminator may be another system with which the modeled system communicates.^[2]



Input (or) output

3.3.2.Different Types of DFD's:

There are three different types of DFD's; Context, Diagram O and detailed dataflow diagrams.

➤ Context Dataflow Diagrams:

The emphasis is on the relationship between the system and the environment. The system itself as a whole is represented by a bubble and the external entities are shown by squares which the input flows and output is directed.

➤ Diagram O:

A Diagram O is a diagram which only shows the system itself. It pictures the major process; external entities, data stores and data flow. IT is a single, top-level diagram of the system, and doesn't go into too much detail for each process.

➤ Detailed DFD's:

They are detailed pictures of major processes or subsystems. They are more likely to be used by designers in their own work rather than for presentations to users or managers. A more detailed diagram can be used if needed.

3.3.3.Data Flow Diagrams

Data flow diagrams provide a graphical representation of how information moves between processes in a system. Data flow diagrams follow a hierarchy; that is, a diagram may consist of several layers, each unique to a specific process or data function. The Data Flow Diagram (DFD) depicts the logic models and expresses data transformation in a system. It includes a mechanism to model the data flow and supports decomposition to illustrate details of the data flows and functions. A Data Flow Diagram cannot present information on operation sequence. Therefore, it is not a process or procedure modeling method. In brief, the purposes of data flow diagrams can be said to be as follows –

1. Supporting the analysis and requirement stage of system design.
2. A diagramming technique with annotation.
3. Describing a network of activities/processes of the target system.
4. Allowing for behaviors of parallel and asynchronous.

5. Stepwise refinement through hierarchical decomposition of processes.

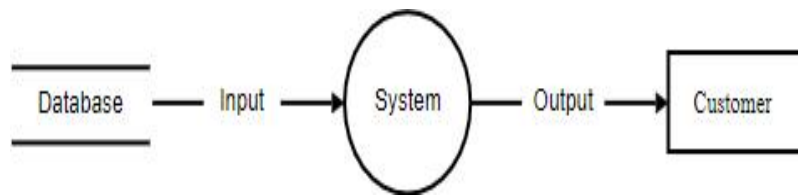


Fig3.3.3.: Data flow diagrams

How to Draw a Data Flow Diagram?

DFD is usually beginning with a context diagram as level 0 , a simple representation of the whole system. To elaborate further from that, we drill down to a level 1 diagram with lower- level functions decomposed from the major functions of the system. This could continue to evolve to become a level 2 diagram when further analysis is required. Progression to levels 3,4 and so on is possible but anything beyond level 3 is not very common.

3.4 TYPES OF DATA FLOW DIAGRAM

3.4.1 Context Level DFD

Context level DFD, also known as level 0 DFD, sees the whole system as a single process and emphasis the interaction between the system and external entities. Let's draw a context DFD.

1. To create a DFD, select Diagram > New from the toolbar.
2. In the New Diagram window, select Data Flow Diagram and click Next.
3. Enter Context Diagram as diagram name and click OK to confirm.
4. Name the diagram Context Diagram.
5. We need to create the main process. Drag Process from diagram toolbar to diagram. Name it Online Bookstore. We will use online bookstore as an example to show you how to create multiple levels DFD.

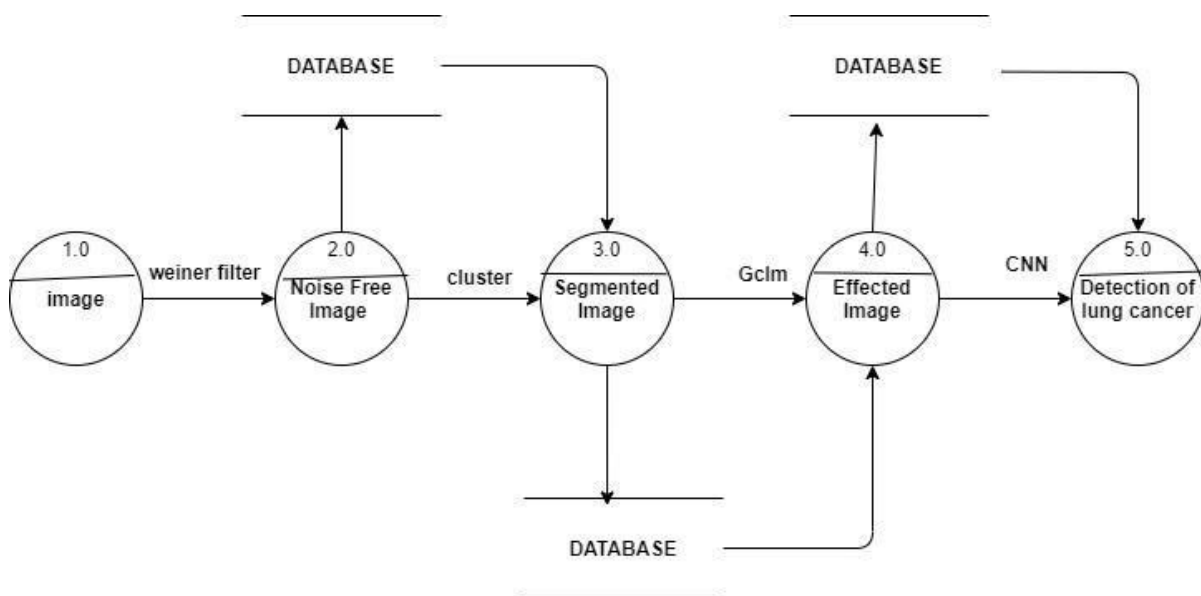


Fig 3.4.1.1: Level 0 DFD

3.4.2. Level 1 DFD

If no context diagram exists, first create one before attempting to construct the level 1 DFD (or construct the context diagram and level 1 DFD simultaneously).

The following steps are suggested to aid the construction of Level 1 DFD:

1. Identify processes. Each data-flow into the system must be received by a process. For each data-flow into the system examine the documentation about the system and talk to the users to establish a plausible process of the system that receives the data-flow. Each process must have at least one output data-flow. Each output data-flow of the system must have been sent by a process; identify the processes that sends each system output.
2. Draw the data-flows between the external entities and processes.
3. Identify data stores by establishing where documents / data needs to be held within the system. Add the data stores to the diagram, labelling them with their local name or description.
4. Add data-flows flowing between processes and data stores within the system. Each data store must have at least one input data-flow and one output data-flow (otherwise data may be stored, and never used, or a store of data must have come from nowhere). Ensure every data store has input and output data-flows to system processes. Most processes are normally associated with at least one data store.

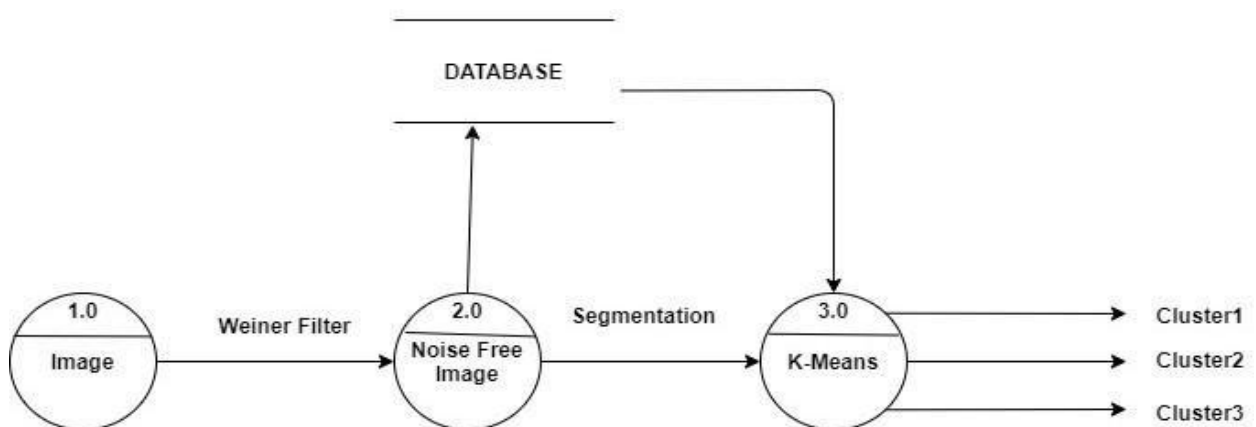


Fig 3.4.2.1: Level 1 DFD

3.4.3 Level 2 DFD Diagram

A level 2 data flow diagram (DFD) offers a more detailed look at the processes that make up an information system than a level 1 DFD does. It can be used to plan or record the specific makeup of a system. You can then input the particulars of your own system.

The level 1 data-flow diagram provides an overview of the system. As the software engineers' understanding of the system increases it becomes necessary to expand most of the level 1 processes to a second or even third level in order to depict the detail within it. Decomposition is applied to each process on the level 1 diagram for which there is enough detail hidden within the process. Each process on the level 2 diagrams also needs to be checked for possible decomposition, and so on.

A process box that cannot be decomposed further is marked with an asterisk in the bottom right hand corner. A brief narrative description of each bottom-level process should be provided with the data-flow diagrams to complete the documentation of the data-flow model. These make up part of the process definitions which should be supplied with the DFD.

Each process on the level 1 diagram is investigated in more detail, to give a greater understanding of the activities and data-flows. Normally processes are decomposed where: There are more than six data-flows around the process.

The process name is complex or very general which indicates that it incorporates a number of activities.

1. Make the process box on the level 1 diagram the system boundary on the level 2 diagram that decomposes it. This level 2 diagram must balance with its "parent" process box. The data-flows to and from the process on the level 1 diagram will all become data-flows across the system boundary on the level 2 diagram. The sources and recipients of data-flows across the level 2 system boundary are drawn outside the boundary and labelled exactly as they are on the level 1 diagram. Note that these sources and recipients may be data stores as well external entities or other processes. This is because a data store in a level 1 diagram will be outside the boundary of a level 2 process that sends or receives data-flows to/from the data store.

STUDENT MANAGEMENT AND MONITORING SYSTEM

1. Identify the processes inside the level 2 system boundary and draw these processes and their data-flows. Remember, each data-flow into and out of the level 2 system boundary should be to/from a process. Using the results of the more detailed investigation, filter out and draw the processes at the lower level that send and receive information both across and within the level 2 system boundary. Use the level numbering system to number sub-processes so that, for example, process 4 on the level 1 diagram is decomposed to sub-processes 4.1, 4.2, 4.3 ... on the level 2 diagram.
2. Identify any data stores that exist entirely within the level 2 boundary, and draw these data stores
3. Identify data-flows between the processes and data stores that are entirely within the level 2 system boundary. Remember, every data store inside this boundary should have at least one input and one output data flow.
4. Check the diagram. Ensure that the level 2 data-flow diagram does not violate the rules for data-flow diagram constructs.

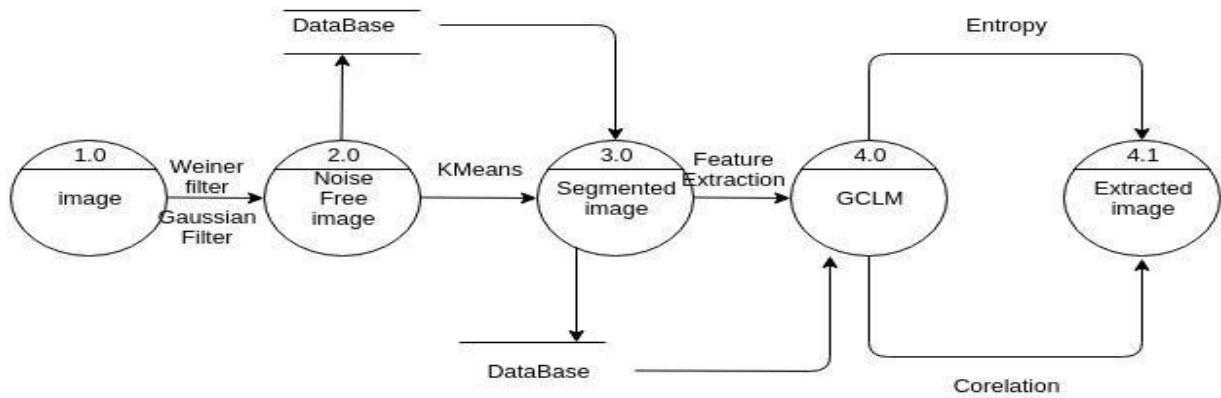


Fig 3.4.3.1: Level 2 DFD

3.4.4 Making levels

For all systems it is useful to make at least two levels — the context diagram and the level one diagram. In fact, when in the earlier description of how to create DFDs you were told to start by identifying the external entities and then to identify the inputs and outputs of the system, you were learning how to

produce the context diagram. The rest of the description was how to produce the level one diagram. Whenever you perform data-flow modeling, start in exactly this way, producing a context diagram and then a level one diagram. Of course, in producing the level one diagram you may realize you need more inputs and outputs and possibly even more external entities. In this case, simply add the new data-flows and the new entities to the level one diagram and then go back and add them to the context diagram so that both diagrams still balance. Conversely, you may realise that some of the inputs and outputs you originally identified are not relevant to the system. Remove them from the level one diagram and then go back to the context diagram and make it balance by removing the same inputs and outputs. This constant balancing between diagrams is very common when doing levelling.

What about making more levels? There are two reasons for making more levels. The first is the obvious one: you, as the software engineer, have not fully described a process to your satisfaction, so you expand that process into a next level diagram.

The new diagram is built in just the same way that a level one is built from a context diagram only the new inputs and outputs are precisely to the data flows to and from the process you are expanding.

The second reason is that you realise the diagram you are working on is becoming cluttered and unclear. To simplify the diagram, collect together a few of the processes. Ideally, these processes should be related in some way. Replace them with a single process and treat the original collection of processes as a lower level, expanding the new process. The inputs and outputs to the new process are whatever inputs and outputs that are needed to make the diagrams balance. Remember to re-number the old processes to show that they have been moved down a level.

3.5 DFD HEIRARCHY

To make the DFD more transparent (i.e. not too many processes), multi-level DFDs can be created. DFDs that are at a higher level are less detailed (aggregate more detailed DFD at lower levels). The contextual DFD is the highest in the hierarchy (see DFD Creation Rules). The so-called zero level is followed by DFD 0, starting with process numbering (e.g., process1, process 2). In the next, the so-called first level - DFD 1 - the numbering continues. E.g. process 1 is divided into the first three levels of the DFD, which are numbered 1.1, 1.2 and 1.3.

Similarly, processes in the second level (DFD 2) are numbered eg 1.1.1, 1.1.2, 1.1.3 and 1.1.4. The number of levels depends on the size of the model system. DFD 0 processes may not have the same number of decomposition levels. DFD 0 contains the most important (aggregated) system functions. The lowest level should include processes that make it possible to create a process specification (Process Specification) for roughly one A4 page. If the mini-specification should be longer, it is appropriate to create an additional level for the process where it will be decomposed into multiple processes. For a clear overview of the entire DFD hierarchy, a vertical (cross-sectional) diagram can be created. The warehouse is displayed at the highest level where it is first used and at every lower level as well.

3.6 STUDENT DATA ENTERY

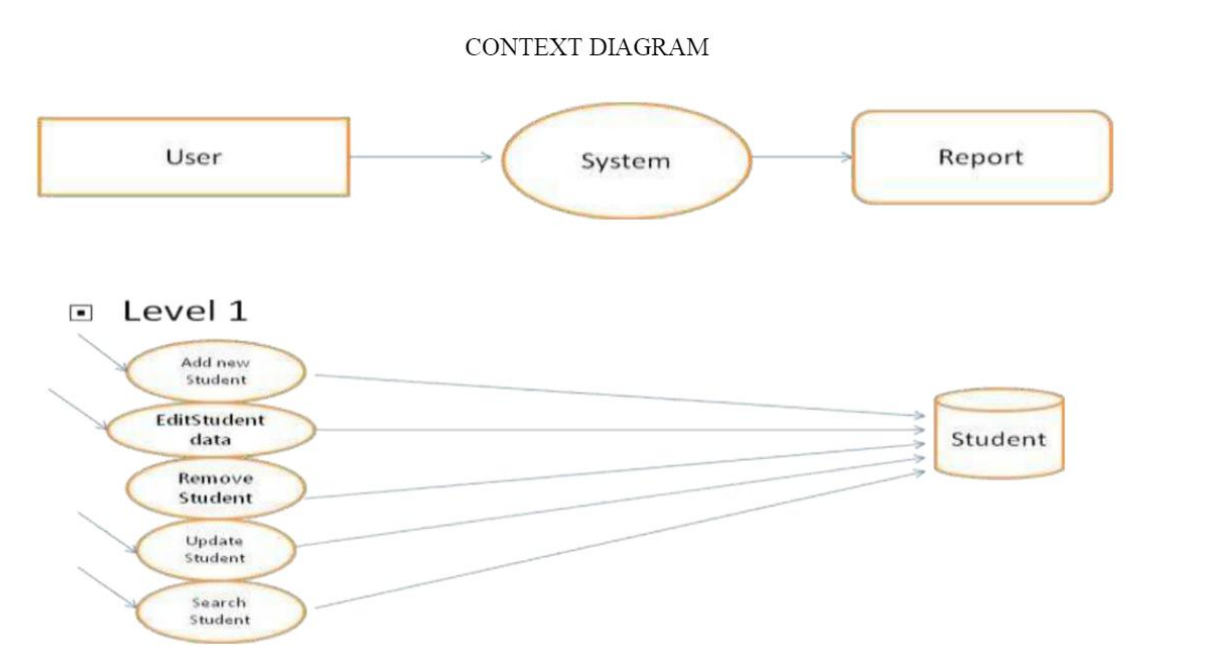


Fig 3.6.1 Level 1 of Student Data Entry

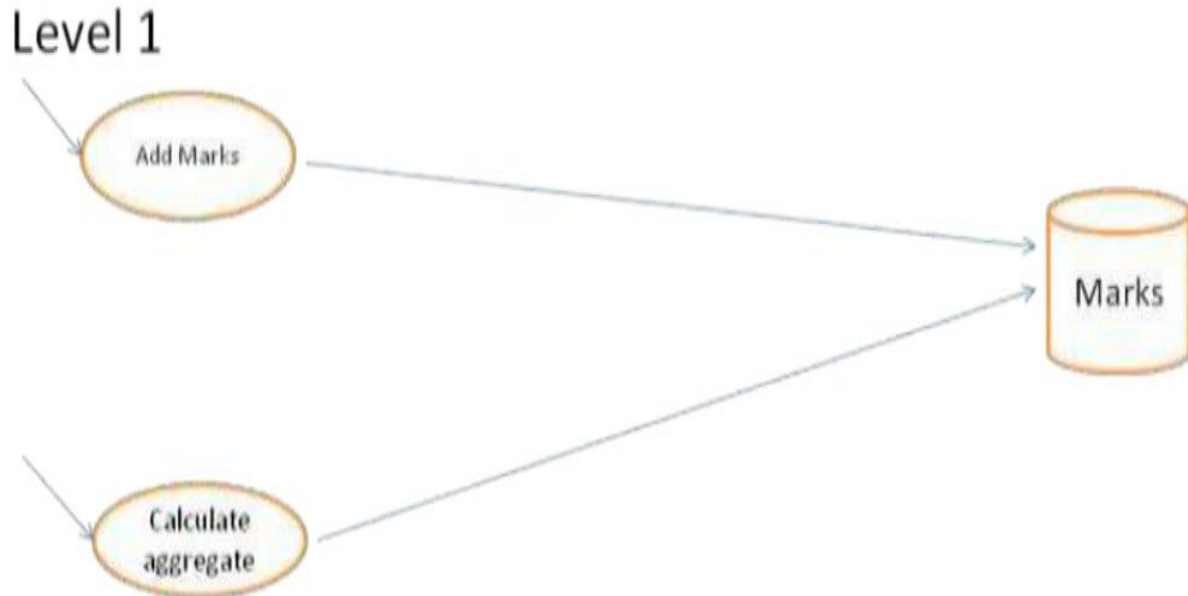


Fig 3.6.2 Marks entry into Database

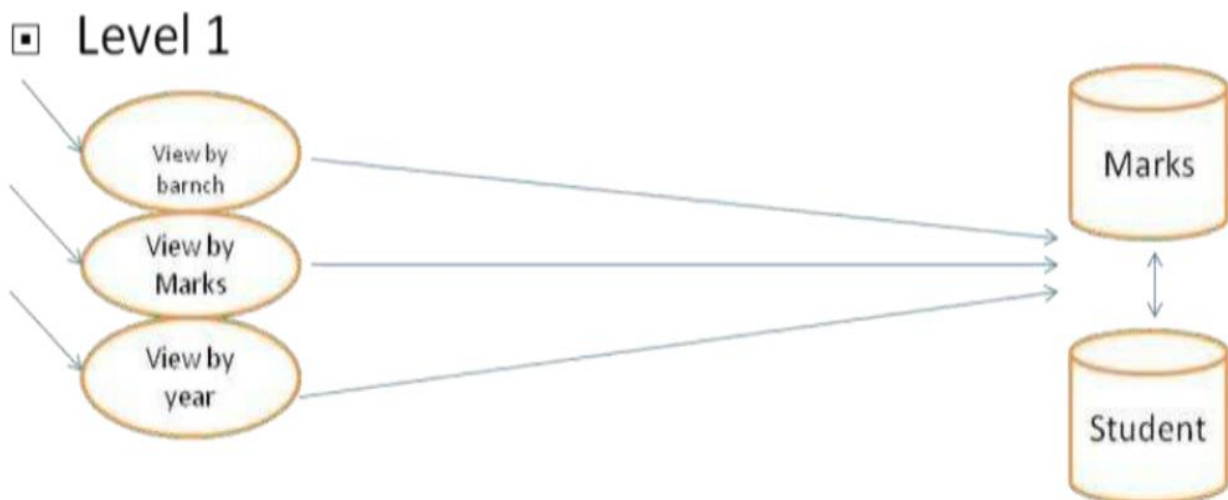


Fig 3.6.3 Marks with Student Database

STUDENT MANAGEMENT AND MONITORING SYSTEM

A screenshot of a web-based student registration form. The form is centered on a light blue background with abstract geometric patterns. It contains several input fields and a dropdown menu, each preceded by a label in a bold, serif font. The labels are: 'ROLL NUMBER', 'First Name', 'Last Name', 'Branch', 'Age', 'Gender', and 'Father / Guardian Name'. The input fields are rectangular with thin borders. The 'Gender' field is a dropdown menu with a downward arrow. The form is presented within a browser window frame.

ROLL NUMBER

First Name

Last Name

Branch

Age

Gender

Father / Guardian Name

Fig 3.6.4 Student Registration Form(a)

STUDENT MANAGEMENT AND MONITORING SYSTEM

A screenshot of a web-based student registration form. The form is centered on a light blue background with a subtle geometric pattern. It contains seven input fields, each preceded by a label. The labels are: 'Father / Guardian Name', 'Mother / Guardian Name', 'Student Mobile Number', 'Email', 'Parent / Guardian Mobile number', 'Address', and 'Percentage or CGPA scored in Class 10th'. The input fields for 'Student Mobile Number', 'Email', and 'Parent / Guardian Mobile number' contain placeholder text: 'Enter your Mobile numbe', 'Your email', and 'Enter your Parent / Guan' respectively. The form is enclosed in a light blue border with a small upward arrow in the top right corner and a small downward arrow in the bottom right corner.

Father / Guardian Name

Mother / Guardian Name

Student Mobile Number

Email

Parent / Guardian Mobile number

Address

Percentage or CGPA scored in Class 10th

Fig 3.6.5 Student Registrartion Form(b)

STUDENT MANAGEMENT AND MONITORING SYSTEM

Percentage or CGPA scored in Class 10th

Percentage or CGPA scored in Class 12th / Inter

Current Percentage in BTECH

Number of Backlogs

Hobbies

City

ZIP Code

Fig 3.6.6 Student Registration Form(c)

A web form titled "Student Registration Form(d)" with a light blue and white geometric background. The form contains several input fields and two buttons at the bottom.

Number of Backlogs

Hobbies

City

ZIP Code

State

Country

Submit Reset

Fig 3.6.7 Student Registration Form(d)

ROLL NUMBER

5143

First Name

ROCKY

Last Name

KRISHNAMACHARI

Branch

Private

Age

50

Gender

Male ▼

Father / Guardian Name

RAHUL KRISHNAMACHARI

Fig 3.6.8 Editing Student Details

DJANGO

4. DJANGO

4.1 ABOUT DJANGO

Django is a Python-based web framework, free and open-source, that follows the model–template–views (MTV) architectural pattern. It is maintained by the Django Software Foundation (DSF), an independent organization established in the US as a non-profit.

Django's primary goal is to ease the creation of complex, database-driven websites. The framework emphasizes reusability and "pluggability" of components, less code, low coupling, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings, files, and data models. Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models.

Django was created in the fall of 2003, when the web programmers at the *Lawrence Journal-World* newspaper, Adrian Holovaty and Simon Willison, began using Python to build applications. Jacob Kaplan-Moss was hired early in Django's development shortly before Simon Willison's internship ended. It was released publicly under a BSD license in July 2005. The framework was named after guitarist Django Reinhardt. Adrian Holovaty is a Romani jazz guitar player and a big fan of Django Reinhardt.[*citation needed*]

In June 2008, it was announced that a newly formed Django Software Foundation (DSF) would maintain Django in the future.

4.2 COMPONENTS

Django was created in the fall of 2003, when the web programmers at the *Lawrence Journal-World* newspaper, Adrian Holovaty and Simon Willison, began using Python to build applications. Jacob Kaplan-Moss was hired early in Django's development shortly before Simon Willison's internship ended.[*]* It was released publicly under a BSD license in July 2005. The framework was named after guitarist Django Reinhardt. Adrian Holovaty is a Romani jazz guitar player and a big fan of Django Reinhardt.[*citation needed*]

In June 2008, it was announced that a newly formed Django Software Foundation (DSF) would maintain Django in the future.

Despite having its own nomenclature, such as naming the callable objects generating the HTTP responses "views",[7] the core Django framework can be seen as an MVC architecture.[8] It consists of an object-relational mapper (ORM) that mediates between data models (defined as Python classes) and a relational database ("Model"), a system for processing HTTP requests with a web templating system ("View"), and a regular-expression-based URL dispatcher ("Controller").

Also included in the core framework are:

- a lightweight and standalone web server for development and testing
- a form serialization and validation system that can translate between HTML forms and values suitable for storage in the database

STUDENT MANAGEMENT AND MONITORING SYSTEM

- a template system that utilizes the concept of inheritance borrowed from object-oriented programming
- a caching framework that can use any of several cache methods
- support for middleware classes that can intervene at various stages of request processing and carry out custom functions
- an internal dispatcher system that allows components of an application to communicate events to each other via pre-defined signals
- an internationalization system, including translations of Django's own components into a variety of languages
- a serialization system that can produce and read XML and/or JSON representations of Django model instances
- a system for extending the capabilities of the template engine
- an interface to Python's built-in unit test framework
-

4.3 EXTENSIBILITY

Django's configuration system allows third party code to be plugged into a regular project, provided that it follows the reusable app conventions. More than 2500 packages are available to extend the framework's original behavior, providing solutions to issues the original tool didn't tackle: registration, search, API provision and consumption, CMS, etc.

This extensibility is, however, mitigated by internal components' dependencies. While the Django philosophy implies loose coupling, the template filters and tags assume one engine implementation, and both the auth and admin bundled applications require the use of the internal ORM. None of these filters or bundled apps are mandatory to run a Django project, but reusable apps tend to depend on them, encouraging developers to keep using the official stack in order to benefit fully from the apps ecosystem.

4.4 SERVER ARRANGEMENTS

Django can be run in conjunction with Apache, Nginx using WSGI, Gunicorn, or Cherokee using flup (a Python module). Django also includes the ability to launch a FastCGI server, enabling use behind any web server which supports FastCGI, such as Lighttpd or Hiawatha. It is also possible to use other WSGI-compliant web servers. Django officially supports five database backends: PostgreSQL, MySQL, MariaDB, SQLite, and Oracle. Microsoft SQL Server can be used with `django-mssql` while similarly external backends exist for IBM Db2, SQL Anywhere and Firebird. There is a fork named `django-nonrel`, which supports NoSQL databases, such as MongoDB and Google App Engine's Datastore.

Django may also be run in conjunction with Jython on any Java EE application server such as GlassFish or JBoss. In this case `django-jython` must be installed in order to provide JDBC drivers for database connectivity, which also can provide functionality to compile Django in to a .war suitable for deployment.

Google App Engine includes support for Django version 1.x.x as one of the bundled frameworks.

4.5 FEATURES OF DJANGO

4.5.1 Complete:

Django follows the "Batteries included" philosophy and provides almost everything developers might want to do "out of the box". Because everything you need is part of the one "product", it all works seamlessly together, follows consistent design principles, and has extensive and up-to-date documentation.

4.5.2 Versatile:

Django can be (and has been) used to build almost any type of website — from content management systems and wikis, through to social networks and news sites. It can work with any client-side framework, and can deliver content in almost any format (including HTML, RSS feeds, JSON, XML, etc).

Internally, while it provides choices for almost any functionality you might want (e.g. several popular databases, templating engines, etc.), it can also be extended to use other components if needed.

4.5.3 Secure:

Django helps developers avoid many common security mistakes by providing a framework that has been engineered to "do the right things" to protect the website automatically. For example, Django provides a secure way to manage user accounts and passwords, avoiding common mistakes like putting session information in cookies where it is vulnerable (instead cookies just contain a key, and the actual data is stored in the database) or directly storing passwords rather than a password hash.

A password hash is a fixed-length value created by sending the password through a cryptographic hash function. Django can check if an entered password is correct by running it through the hash function and comparing the output to the stored hash value. However due to the "one-way" nature of the function, even if a stored hash value is compromised it is hard for an attacker to work out the original password.

Django enables protection against many vulnerabilities by default, including SQL injection, cross-site scripting, cross-site request forgery and clickjacking (see Website security for more details of such attacks).

4.5.4 Scalable:

Django uses a component-based "shared-nothing" architecture (each part of the architecture is independent of the others, and can hence be replaced or changed if needed). Having a clear separation between the different parts means that it can scale for increased traffic by adding hardware at any level: caching servers, database servers, or application servers. Some of the busiest sites have successfully scaled Django to meet their demands (e.g. Instagram and Disqus, to name just two).

4.5.5 Maintainable:

Django code is written using design principles and patterns that encourage the creation of maintainable and reusable code. In particular, it makes use of the Don't Repeat Yourself (DRY) principle so there is no unnecessary duplication, reducing the amount of code. Django also promotes the grouping of related functionality into reusable "applications" and, at a lower level, groups related code into modules (along the lines of the Model View Controller (MVC) pattern).

4.5.6 Portable:

Django is written in Python, which runs on many platforms. That means that you are not tied to any particular server platform, and can run your applications on many flavors of Linux, Windows, and macOS. Furthermore, Django is well-supported by many web hosting providers, who often provide specific infrastructure and documentation for hosting Django sites

DJANGO IMPLEMENTATION

5. DJANGO IMPLEMENTATION

5.1 INSTALLING DJANGO

Step 1 — Opening PowerShell

First, you need to open PowerShell on your computer. You can do that by searching for PowerShell in the Windows search box or you can open the **Run** dialog box by holding the **Windows logo** key and **R**(WIN+R). Once the dialog is open, type powershell, and then click **OK**.

You should now have the PowerShell window opened.

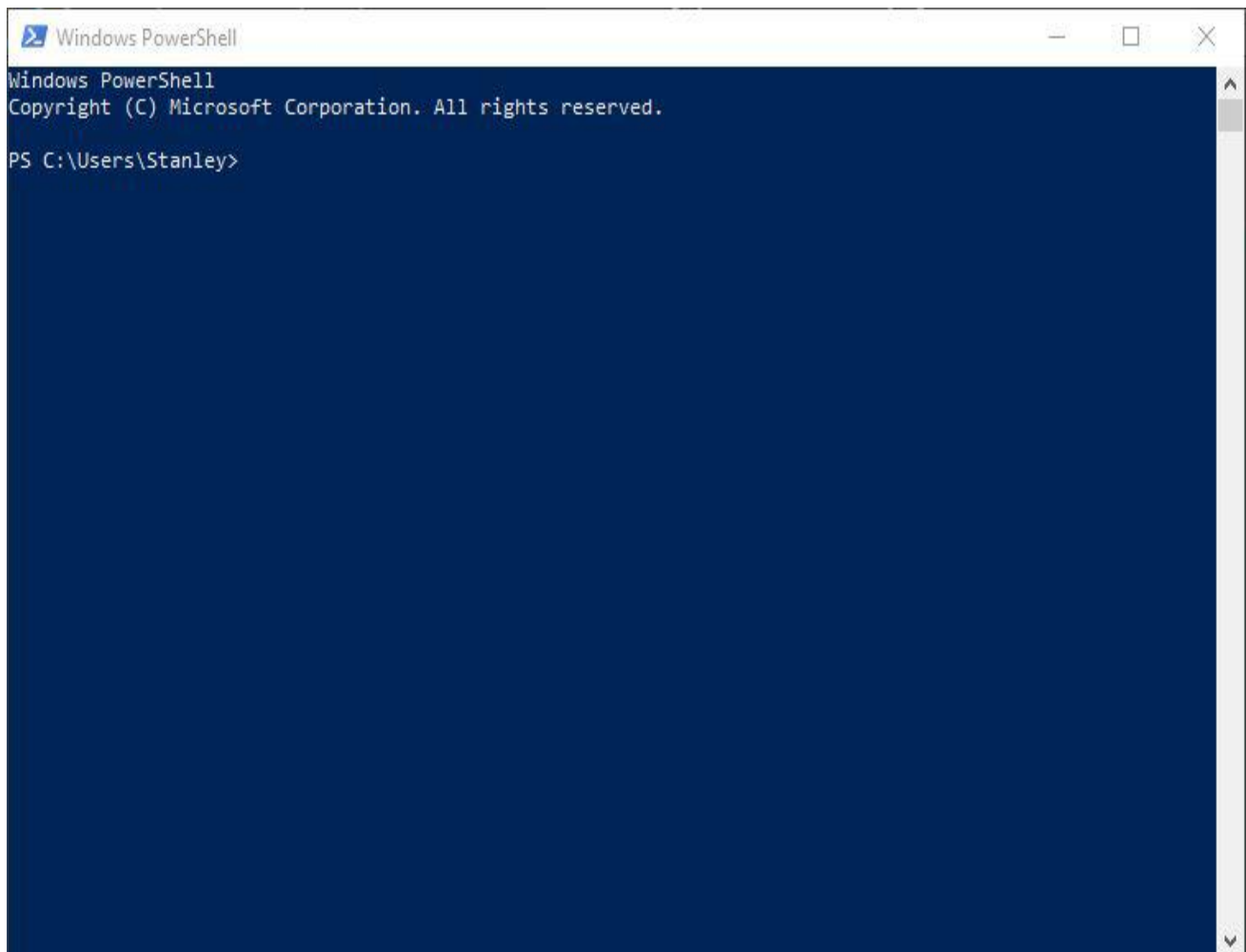


Fig 5.1.1 Opening Powershell

Now that you have opened PowerShell on your computer, you'll verify the installation of Python in the next section.

Step 2 - Verifying Python Installation

Before you install Django, first, you need to make sure that you installed Python on your system.

To do that, type the following command in PowerShell prompt to verify the installation:

```
> python -V
```

-V option logs the Python version installed on your system.

After running the command, you should see output like this:

```
PS C:\Users\Username> python -VPython 3.9.7
```

At the time of writing, it is Python 3.9.7. You might have a different version from mine, and that's fine. As long as you see the Python version logged, Python is installed on your system.

Now that you've confirmed Python is installed on your system, you will upgrade pip.

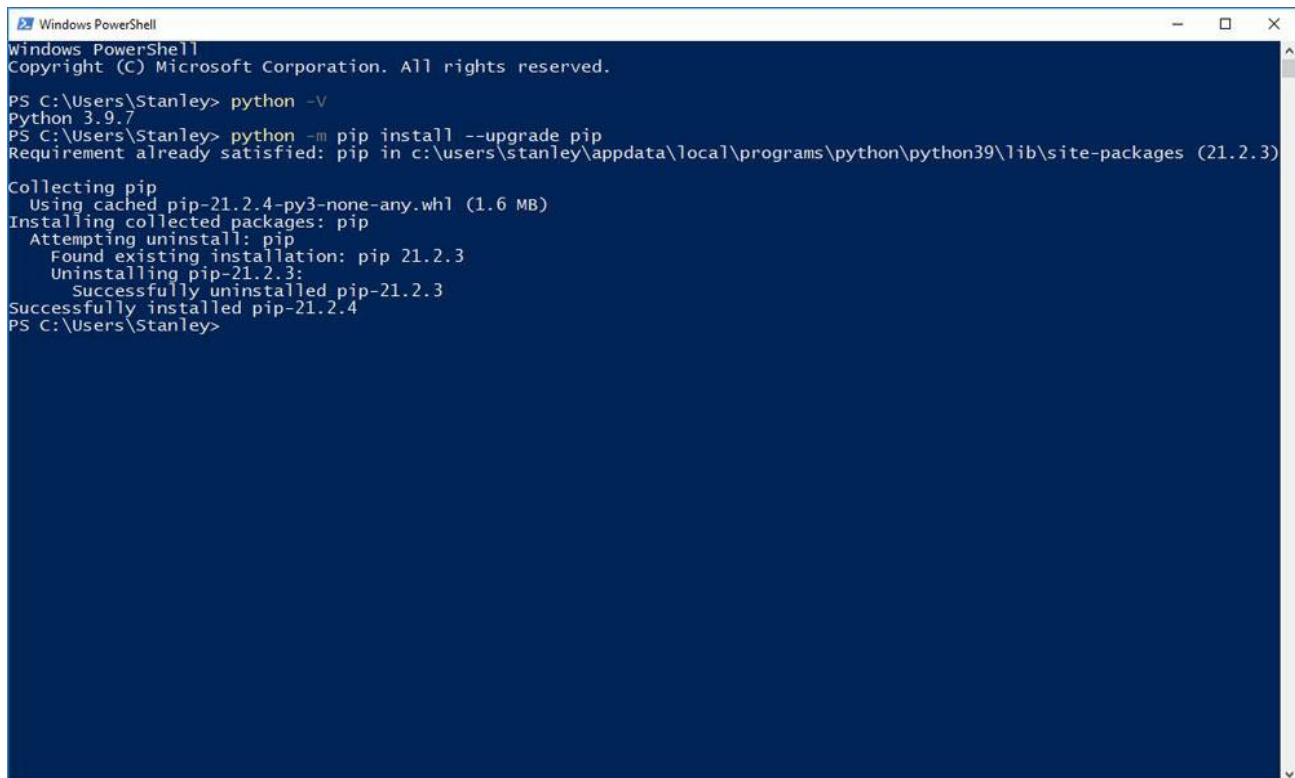
Step 3 - Upgrading Pip

Python comes with pip by default. But most of the time, it comes with an old version. it's always a good practice to upgrade pip to the latest version.

Enter the following command to upgrade pip on your system:

```
> python -m pip install --upgrade pip
```

You'll get output identical to the following screenshot showing you that the upgrade was a success:



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Stanley> python -V
Python 3.9.7
PS C:\Users\Stanley> python -m pip install --upgrade pip
Requirement already satisfied: pip in c:\users\stanley\appdata\local\programs\python\python39\lib\site-packages (21.2.3)
Collecting pip
  Using cached pip-21.2.4-py3-none-any.whl (1.6 MB)
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 21.2.3
    Uninstalling pip-21.2.3:
      Successfully uninstalled pip-21.2.3
Successfully installed pip-21.2.4
PS C:\Users\Stanley>
```

Fig 5.1.2 Verifying Python

Now you've upgraded pip, you'll create the project directory where you'll install Django.

Step 4 - Creating a Project Directory

In this section, you will create a directory that will contain your Django application. We will name it `django_project` since this tutorial is a demo. But in a real project, you can give the directory a suitable name, such as `forum`, `blog`, etc.

Change into your Desktop directory with the `cd` command:

```
> cd Desktop
```

Create the directory using the `mkdir` command:

```
> mkdir django_project
```

Move into the `django_project` directory using the `cd` command:

```
> cd django_project
```

Your prompt should now show you that you're in the `django_project` directory as shown in the following output:

```
PS C:\Users\Stanley\Desktop\django_project>
```

Now that you've created the working directory for your project, you'll create a virtual environment where you'll install Django.

Step 5 - Creating the Virtual Environment

In this step, you'll create a virtual environment for your project. A virtual environment is an isolated environment in Python where you can install the project dependencies without affecting other Python projects. This lets you create different projects that use different versions of Django.

If you don't use a virtual environment, your projects in your system will use the same Django version installed globally. This might look like a good thing until the latest version of Django comes out with breaking changes causing your projects to fail altogether.

You can learn more about the virtual environment by following [Python Virtual Environments: A Primer](#).

To create a virtual environment, type the following command and wait for a few seconds:

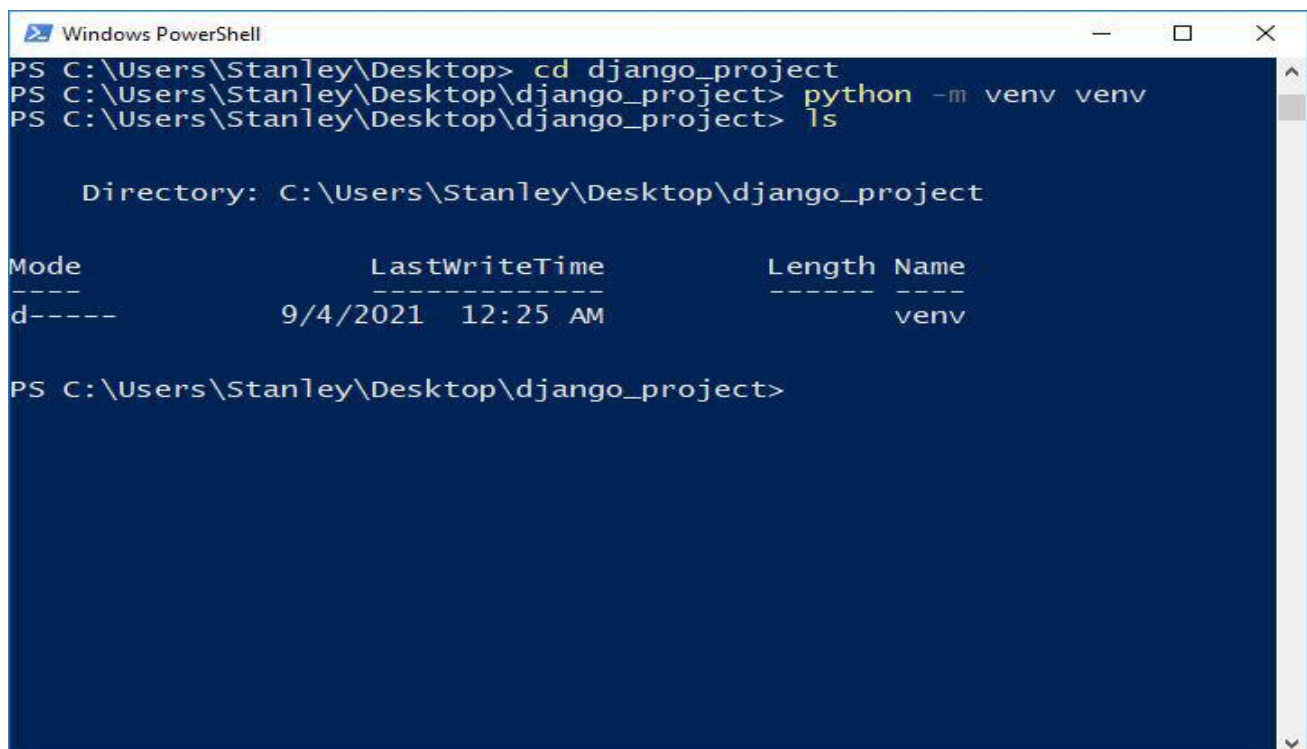
```
> python -m venv venv
```

The command will create a directory called `venv` inside your project directory.

Next, confirm the `venv` directory has been created by listing the directory contents using the `ls` command:

```
> ls
```

You should see the directory `venv` in the output as shown in the following screenshot:



```
Windows PowerShell
PS C:\Users\Stanley\Desktop> cd django_project
PS C:\Users\Stanley\Desktop\django_project> python -m venv venv
PS C:\Users\Stanley\Desktop\django_project> ls

        Directory: C:\Users\Stanley\Desktop\django_project

Mode                LastWriteTime         Length Name
----                -
d-----          9/4/2021 12:25 AM                venv

PS C:\Users\Stanley\Desktop\django_project>
```

Fig 5.1.3 Creating a Virtual Environment

Now you've created the virtual environment directory, you'll activate the environment.

Step 6 - Activating the Virtual Environment

In this section, you'll activate the virtual environment in your directory.

Run the following command to activate the virtual environment:

```
> venv\Scripts\activate
```

After you run the command, you will see

a (venv) at the beginning of the prompt. This shows that the virtual environment is activated:

```
(venv) PS C:\Users\Stanley\Desktop\django_project>
```

If you run into the error shown in the following screenshot on PowerShell when activating the virtual environment, for the sake of brevity, I described the reason and the solution in another post. Follow [Solution to "Running Scripts Is Disabled On This System" Error on PowerShell](#) and don't close your PowerShell:

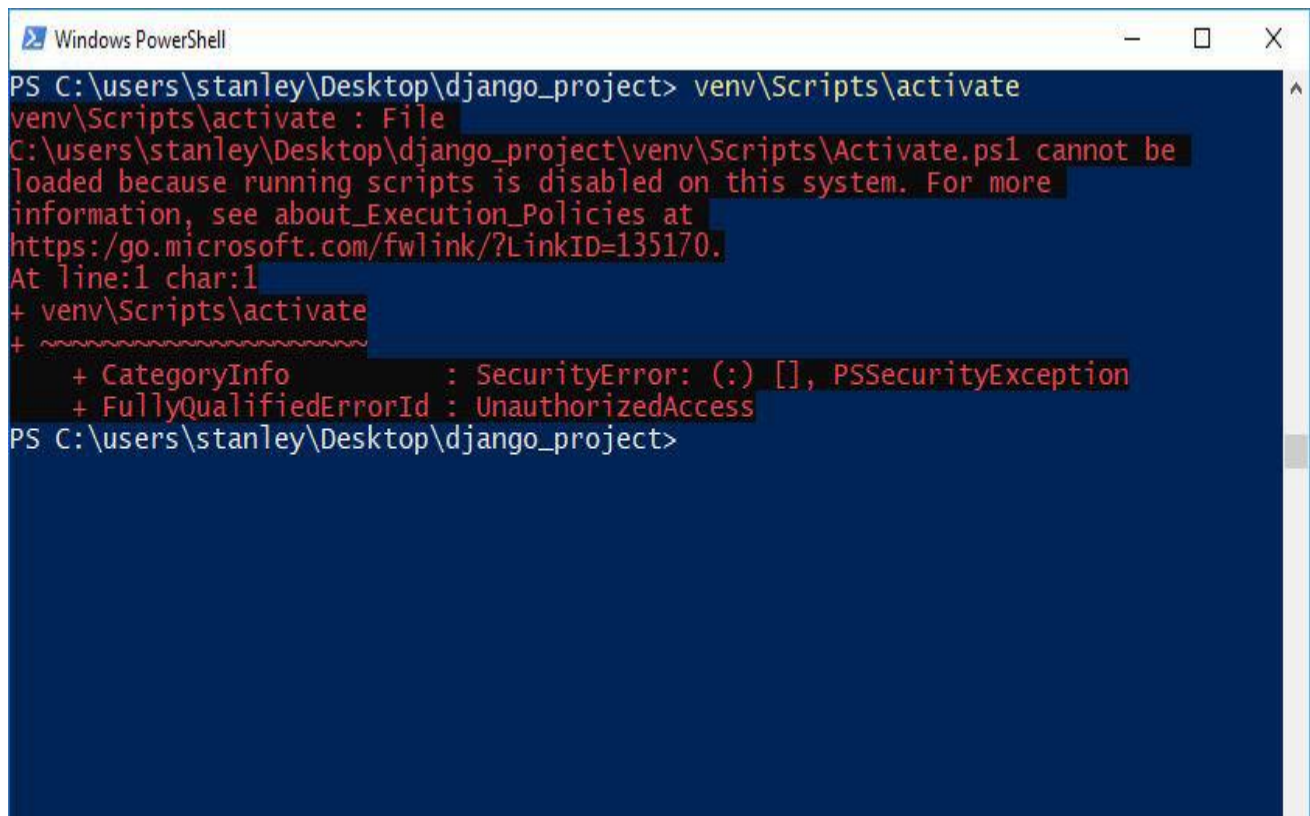


Fig 5.1.4 Activating the Virtual Environment

Now that you've activated the virtual environment for your project, the moment you've been waiting for is here. It's time to install Django!

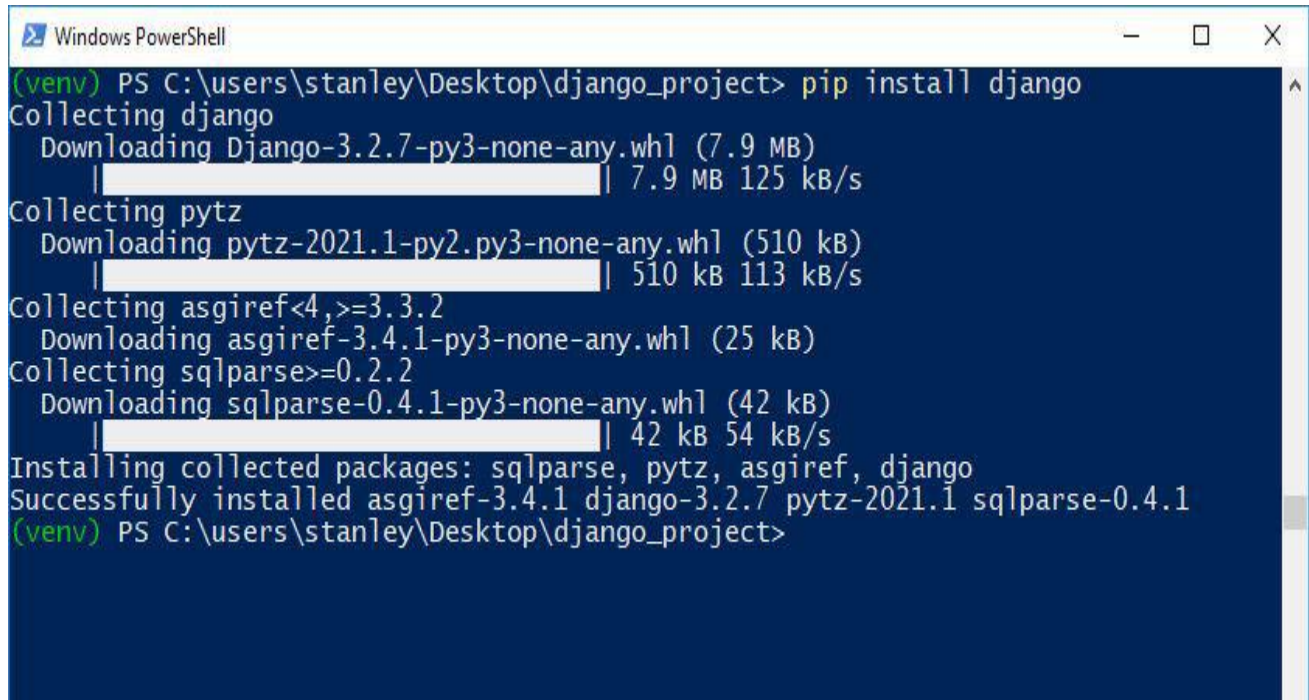
Step 7 - Installing Django

In this section, you will install Django on your system using pip.

Run the following command to install Django using pip install:

```
(venv)> pip install django
```

The command will install the latest version of Django. You should see Django being downloaded as shown in the following screenshot:



```
Windows PowerShell
(venv) PS C:\users\stanley\Desktop\django_project> pip install django
Collecting django
  Downloading Django-3.2.7-py3-none-any.whl (7.9 MB)
    | 7.9 MB 125 kB/s
Collecting pytz
  Downloading pytz-2021.1-py2.py3-none-any.whl (510 kB)
    | 510 kB 113 kB/s
Collecting asgiref<4,>=3.3.2
  Downloading asgiref-3.4.1-py3-none-any.whl (25 kB)
Collecting sqlparse>=0.2.2
  Downloading sqlparse-0.4.1-py3-none-any.whl (42 kB)
    | 42 kB 54 kB/s
Installing collected packages: sqlparse, pytz, asgiref, django
Successfully installed asgiref-3.4.1 django-3.2.7 pytz-2021.1 sqlparse-0.4.1
(venv) PS C:\users\stanley\Desktop\django_project>
```

Fig 5.1.5 Installing Django

If you want to install a different Django version, you can specify the version as follows:

```
(venv)> pip install django==3.1
```

Once the installation finishes, you need to verify that Django has been installed. To do that, type the following command:

```
(venv)> django-admin --version
```

You will get output showing you the Django version installed on your system:

```
(venv) PS C:\users\stanley\Desktop\django_project> django-admin --version3.2.7
```

At the time of writing, the latest Django version is 3.2.7, and that's why my output shows that.

You've now installed Django on your system, great job! You'll begin to create a Django project.

Step 8 - Creating the Django Project

Now it's time to create a project. A project has a different meaning from what you may be used to. The [Django documentation](#) defines it as:

A Python package – i.e. a directory of code – that contains all the settings for an instance of Django. This would include database configuration, Django-specific options and application-specific settings.

You create a project by using the command-line utility `django-admin` that comes with Django. The command generates files where you can configure the settings for your database, add third-party packages for your project to mention a few.

Create the project using the `django-admin startproject` command:

```
(venv)> django-admin startproject test project
```

Change into the `test_project` directory:

```
(venv)> cd test project
```

Type the following command to see the contents in the project directory:

```
(venv)> ls test project
```

You will get output similar to this:

```
(venv) PS C:\users\stanley\Desktop\django_project\test_project> ls
```

```
Directory: C:\users\stanley\Desktop\django_project\test_project
```

Mode	LastWriteTime	Length	Name
----	-----	-----	----
d----	9/4/2021 1:25 AM		test project
-a----	9/4/2021 1:25 AM	690	manage.py

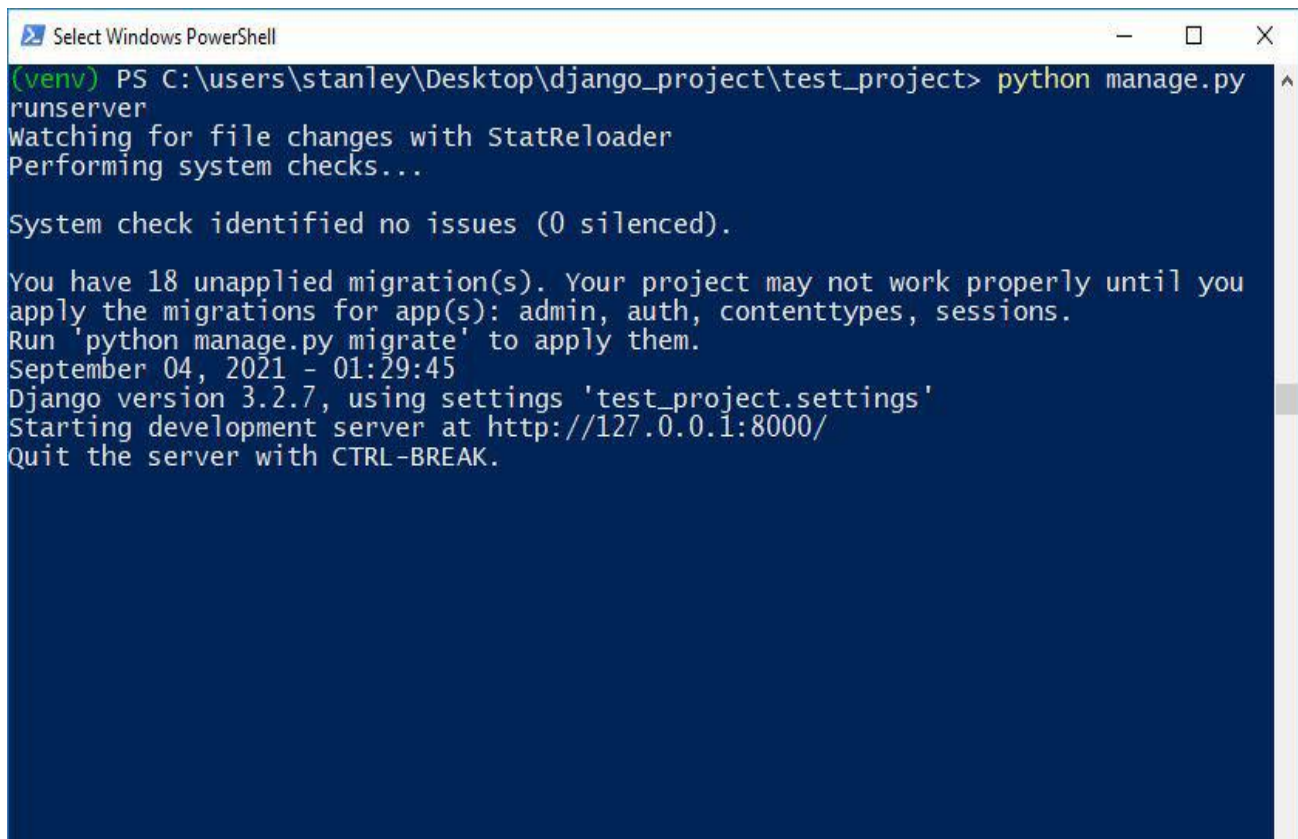
The directory `test_project` contains Django configuration files. The `manage.py` file comes in handy when starting a development server, and that's what you will do in the next step.

Step 9 - Running the Development Server

Now that the project has been created, we will start the Django development server.

Start the development server using the `manage.py runserver` command:

```
(venv)> python manage.py runserver
```



```
Select Windows PowerShell
(venv) PS C:\users\stanley\Desktop\django_project\test_project> python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you
apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
September 04, 2021 - 01:29:45
Django version 3.2.7, using settings 'test_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Fig 5.1.6 Running the Development Server

Next, visit <http://127.0.0.1:8000/> in your web browser. You should see a page similar to the following screenshot:

STUDENT MANAGEMENT AND MONITORING SYSTEM

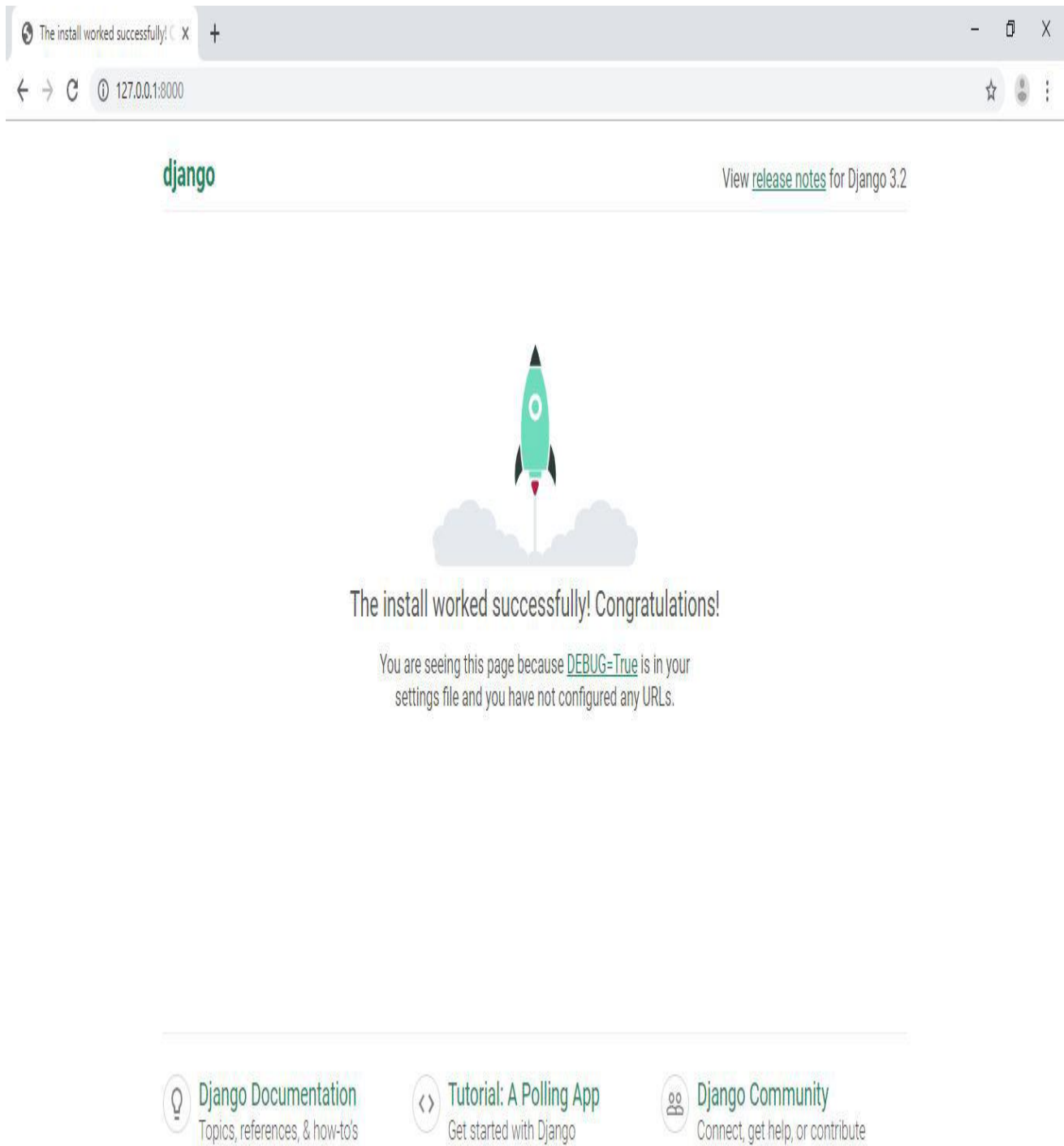


Fig 5.1.7 Opening page in browser

Now, you are ready to start developing your project.

SYSTEM IMPLEMENTATION

6. SYSTEM IMPLEMENTATION

6.1 Source Code of Project

6.1.1 wsgi.py

```
import os

from django.core.wsgi import get_wsgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'classroom_monitoring_system.settings')

application = get_wsgi_application()
```

6.1.2 urls.py

```
from django.contrib import admin
from django.urls import path, include
from cms_app import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('class/', views.classes, name='class'),
]
```

6.1.3 asgi.py

```
import os

from django.core.asgi import get_asgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'classroom_monitoring_system.settings')

application = get_asgi_application()
```

6.1.4 settings.py

```
from pathlib import Path

# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.1/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = '1ybr$vu5p^x0oh4bv5&=m5l#*xhf34r$u8io03%$k$e1_8h2y'

# SECURITY WARNING: don't run with debug turned on in production!
```


STUDENT MANAGEMENT AND MONITORING SYSTEM

```
DEBUG = True
```

```
ALLOWED_HOSTS = []
```

```
# Application definition
```

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'cms_app',  
]
```

```
MIDDLEWARE = [  
    'django.middleware.security.SecurityMiddleware',  
    'django.contrib.sessions.middleware.SessionMiddleware',  
    'django.middleware.common.CommonMiddleware',  
    'django.middleware.csrf.CsrfViewMiddleware',  
    'django.contrib.auth.middleware.AuthenticationMiddleware',  
    'django.contrib.messages.middleware.MessageMiddleware',  
    'django.middleware.clickjacking.XFrameOptionsMiddleware',  
]
```

```
ROOT_URLCONF = 'classroom_monitoring_system.urls'
```

```
TEMPLATES = [  
    {  
        'BACKEND': 'django.template.backends.django.DjangoTemplates',  
        'DIRS': [],  
        'APP_DIRS': True,  
        'OPTIONS': {  
            'context_processors': [  
                'django.template.context_processors.debug',  
                'django.template.context_processors.request',  
                'django.contrib.auth.context_processors.auth',  
                'django.contrib.messages.context_processors.messages',  
            ],  
        },  
    ],  
]
```

```
WSGI_APPLICATION = 'classroom_monitoring_system.wsgi.application'
```

```
# Database
```

```
# https://docs.djangoproject.com/en/3.1/ref/settings/#databases
```

```
DATABASES = {  
    'default': {  
        'ENGINE': 'django.db.backends.sqlite3',  
        'NAME': BASE_DIR / 'db.sqlite3',  
    }  
}
```

```
# Password validation
# https://docs.djangoproject.com/en/3.1/ref/settings/#auth-password-validators
```

```
AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]
```

```
# Internationalization
# https://docs.djangoproject.com/en/3.1/topics/i18n/
```

```
LANGUAGE_CODE = 'en-us'
```

```
TIME_ZONE = 'UTC'
```

```
USE_I18N = True
```

```
USE_L10N = True
```

```
USE_TZ = True
```

```
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/3.1/howto/static-files/
```

```
STATIC_URL = '/static/'
```

6.2 Source codes for User app

6.2.1 views.py

```
from django.http import HttpRequest, HttpResponse
from django.shortcuts import render, redirect

from cms_app.models import student
```

```
def classes(request):
    if request.method == "POST":
        A = request.POST['Rollnumber']
        B = request.POST['FName']
        C = request.POST['LName']
        D = request.POST['br']
        E = request.POST['age']
        F = request.POST['gender']
```

```

G = request.POST['fgname']
H = request.POST['mgname']
I = request.POST['sno']
J = request.POST['Email']
K = request.POST['pno']
L = request.POST['add']
M = request.POST['ed']
N = request.POST['ind']
O = request.POST['bd']
P = request.POST['blog']
Q = request.POST['hobbies']
R = request.POST['city']
S = request.POST['pcode']
T = request.POST['state']
U = request.POST['cntry']

student.objects.create(Rollnumber=A,FName=B,LName=C,br=D,age=E,gender=F,father_name=G,mother_name=H,student_no=I,Email=J,parent_no=K,address=L,tenth=M,inter=N,btech=O,backlog=P,hobbies=Q,city=R,pcode=S,state=T,cntry=U)

return redirect('class')
# return HttpResponse("<center><h1>Data from registration page <br> {} <br> {} <br> {} <br> {} <br> {} <br> {} <br> {}".format(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U))
return render(request,"reg2.html")
# Create your views here.

```

6.2.2 models.py

```

from django.db import models

# Create your models here.
class student(models.Model):
    Rollnumber = models.CharField(max_length=10)
    FName = models.CharField(max_length=20)
    LName = models.CharField(max_length=20)
    br = models.CharField(max_length=20)
    age = models.IntegerField(null=True)
    gender = models.CharField(max_length=20)
    father_name = models.CharField(max_length=20)
    mother_name = models.CharField(max_length=20)
    student_no = models.CharField(max_length=10)
    Email = models.EmailField(null=True)
    parent_no = models.CharField(max_length=20)
    address = models.CharField(max_length=20)
    tenth = models.CharField(max_length=10)
    inter = models.CharField(max_length=10)
    btech = models.CharField(max_length=10)
    backlog = models.IntegerField(null=True)
    hobbies = models.CharField(max_length=20)
    city = models.CharField(max_length=20)
    pcode = models.IntegerField(null=True)
    state = models.CharField(max_length=20)
    cntry = models.CharField(max_length=20)

```

```

def __str__(self):
    return self.FName

```

6.2.3 apps.py

```
from django.apps import AppConfig

class CmsAppConfig(AppConfig):
    name = 'cms_app'
```

6.2.4 admins.py

```
from django.contrib import admin

from cms_app.models import student
```

```
# Register your models here.
admin.site.register(student)
```

6.2.5 Template(HTML Code)

```
<!DOCTYPE html>
{% load static %}
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<style>
    body {
        background-image: url("{% static 'image/2.jpg' %}" );
        background-repeat: no-repeat;
        background-size: cover;
        background-attachment: fixed;
        font-size: 25px;
    }
</style>
<body><center>
    <div class="container">

        <form class="mt32" method="POST" name="classform" id="form1">
            {% csrf_token %}
            <div class="form-group row">
                <label for="rno" class="control-label col-sm-2"><b>ROLL NUMBER</b></label>
                <div class="col-sm-10">
                    <input type="text" minlength="10" maxlength="10" class="rno" name="Rollnumber" placeholder="Enter
your ROLL NUMBER" required>
                </div>
            </div>
            <br><br>
```

STUDENT MANAGEMENT AND MONITORING SYSTEM

```
<div class="form-group row">
  <label for="fname" class="control-label col-sm-2"><b>First Name</b></label>
  <div class="col-sm-10">
    <input type="text" class="form-control" name="FName" placeholder="First name" required>
  </div>
</div>
<br><br>
```

```
<div class="form-group row">
  <label for="lname" class="control-label col-sm-2"><b>Last Name</b></label>
  <div class="col-sm-10">
    <input type="text" class="form-control" name="LName" placeholder="Last name" required>
  </div>
</div>
<br><br>
```

```
<div class="form-group row">
  <label for="br" class="control-label col-sm-2"><b>Branch</b></label>
  <div class="col-sm-10">
    <input type="text" class="form-control" name="br" placeholder="" required>
  </div>
</div>
<br><br>
```

```
<div class="form-group row">
  <label for="roll" class="control-label col-sm-2"><b>Age</b></label>
  <div class="col-sm-10">
    <input type="number" class="form-control" name="age" placeholder="Your Age" required>
  </div>
</div>
<br><br>
```

```
<div class="form-group row">
  <label class="control-label col-sm-2"><b>Gender</b></label>
  <div class="col-sm-10">
    <select id="gender" name="gender" class="form-control" required>
      <option disabled selected value> -- Select your Gender -- </option>
      <option value="Male">Male</option>
      <option value="Female">Female</option>
      <option value="Other">Other</option>
    </select>
  </div>
</div>
<br><br>
```

```
<div class="form-group row">
  <label for="fgname" class="control-label col-sm-2"><b>Father / Guardian Name</b></label>
  <div class="col-sm-10">
    <input type="text" class="form-control" name="fgname" placeholder="" required>
  </div>
</div>
<br><br>
```

```
<div class="form-group row">
  <label for="mgame" class="control-label col-sm-2"><b>Mother / Guardian Name</b></label>
  <div class="col-sm-10">
    <input type="text" class="form-control" name="mgame" placeholder="" required>
  </div>
</div>
```

STUDENT MANAGEMENT AND MONITORING SYSTEM

```
</div>  
<br><br>
```

```
<div class="form-group row">  
  <label for="sno" class="control-label col-sm-2"><b>Student Mobile Number</b></label>  
  <div class="col-sm-10">  
    <input type="tel" class="form-control" name="sno" placeholder="Enter your Mobile number"  
required>  
  </div>  
</div>  
<br><br>
```

```
<div class="form-group row">  
  <label for="email" class="control-label col-sm-2"><b>Email</b></label>  
  <div class="col-sm-10">  
    <input type="email" class="form-control" name="Email" placeholder="Your email" required>  
  </div>  
</div>  
<br><br>
```

```
<div class="form-group row">  
  <label for="pno" class="control-label col-sm-2"><b>Parent / Guardian Mobile number</b></label>  
  <div class="col-sm-10">  
    <input type="tel" class="form-control" name="pno" placeholder="Enter your Parent / Guardian  
Mobile number" required>  
  </div>  
</div>  
<br><br>
```

```
<div class="form-group row">  
  <label for="add" class="control-label col-sm-2"><b>Address</b></label>  
  <div class="col-sm-10">  
    <input type="text" class="form-control" name="add" placeholder="" required>  
  </div>  
</div>  
<br><br>
```

```
<div class="form-group row">  
  <label for="ed" class="control-label col-sm-2"><b>Percentage or CGPA scored in Class  
10th</b></label>  
  <div class="col-sm-10">  
    <input type="text" class="form-control" name="ed" placeholder="" required>  
  </div>  
</div>  
<br><br>
```

```
<div class="form-group row">  
  <label for="ind" class="control-label col-sm-2"><b>Percentage or CGPA scored in Class 12th /  
Inter</b></label>  
  <div class="col-sm-10">  
    <input type="text" class="form-control" name="ind" placeholder="" required>  
  </div>  
</div>  
<br><br>
```

```
<div class="form-group row">  
  <label for="bd" class="control-label col-sm-2"><b>Current Percentage in BTECH</b></label>  
  <div class="col-sm-10">  
    <input type="text" class="form-control" name="bd" placeholder="" required>  
  </div>
```

STUDENT MANAGEMENT AND MONITORING SYSTEM

```
</div>
<br><br>
```

```
<div class="form-group row">
  <label for="blog" class="control-label col-sm-2"><b>Number of Backlogs</b></label>
  <div class="col-sm-10">
    <input type="number" class="form-control" name="blog" placeholder="" required>
  </div>
</div>
<br><br>
```

```
<div class="form-group row">
  <label for="hob" class="control-label col-sm-2"><b>Hobbies</b></label>
  <div class="col-sm-10">
    <input type="text" class="hob" name="hobbies" placeholder="" required>
  </div>
</div>
<br><br>
```

```
<div class="form-group row">
  <label for="city" class="control-label col-sm-2"><b>City</b></label>
  <div class="col-sm-10">
    <input type="text" class="form-control" name="city" placeholder="" required>
  </div>
</div>
<br><br>
```

```
<div class="form-group row">
  <label for="pcode" class="control-label col-sm-2"><b>ZIP Code</b></label>
  <div class="col-sm-10">
    <input type="number" class="form-control" name="pcode" placeholder="" required>
  </div>
</div>
<br><br>
```

```
<div class="form-group row">
  <label for="state" class="control-label col-sm-2"><b>State</b></label>
  <div class="col-sm-10">
    <input type="text" class="form-control" name="state" placeholder="" required>
  </div>
</div>
<br><br>
```

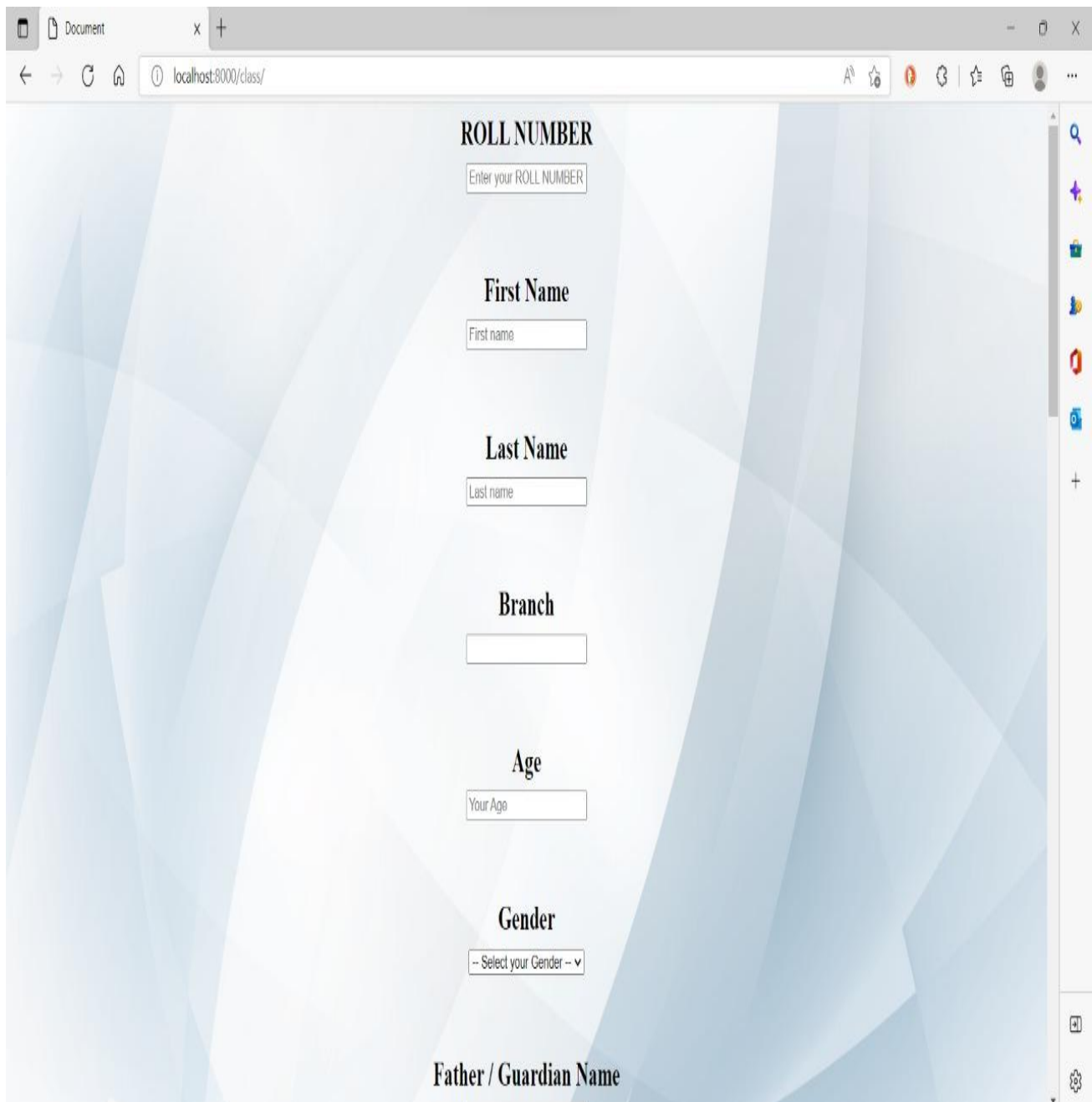
```
<div class="form-group row">
  <label for="cntry" class="control-label col-sm-2"><b>Country</b></label>
  <div class="col-sm-10">
    <input type="text" class="form-control" name="cntry" value="India" required>
  </div>
</div>
<br><br>

<div class="form-group row">
  <div class="offset-sm-2 col-sm-10 pull-right">
    <button type="submit" class="btn btn-danger" onclick="redirect()" >Submit</button>
```

```
    <button type="reset" value="reset" class="btn btn-danger">Reset</button>
  </div>
</div>
```

```
</form>  
</div>  
</center>  
  
</body>  
</html>
```

6.3 Results



The screenshot shows a web browser window with the address bar displaying 'localhost:8000/class/'. The page has a light blue geometric background. The form contains the following fields:

- ROLL NUMBER**: A text input field with the placeholder text 'Enter your ROLL NUMBER'.
- First Name**: A text input field with the placeholder text 'First name'.
- Last Name**: A text input field with the placeholder text 'Last name'.
- Branch**: A text input field.
- Age**: A text input field with the placeholder text 'Your Age'.
- Gender**: A dropdown menu with the text '-- Select your Gender --' and a downward arrow.
- Father / Guardian Name**: A text input field.

Fig 6.3.1 Entering Student Details(a)

STUDENT MANAGEMENT AND MONITORING SYSTEM

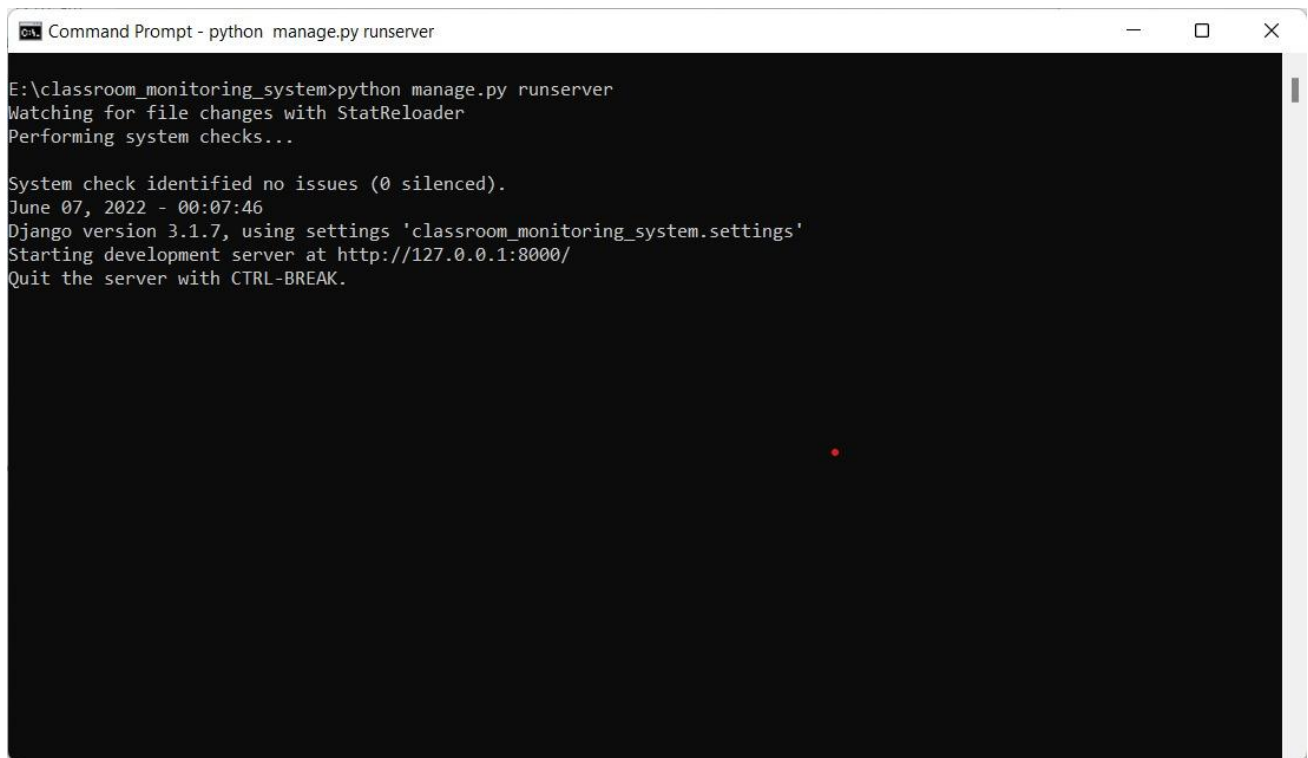


A screenshot of a web browser window displaying a student details form. The browser's address bar shows 'localhost:8000/class/'. The form is centered on a light blue background with a subtle geometric pattern. It contains several input fields with labels above them: 'Number of Backlogs' (value: 0), 'Hobbies' (value: 'Playing cricket, Cooking'), 'City' (value: 'Vijayawada'), 'ZIP Code' (value: '522501'), 'State' (value: 'Andhra Pradesh'), and 'Country' (value: 'India'). At the bottom of the form are two buttons: 'Submit' and 'Reset'.

Field	Value
Number of Backlogs	0
Hobbies	Playing cricket, Cooking
City	Vijayawada
ZIP Code	522501
State	Andhra Pradesh
Country	India

Fig 6.3.2 Entering Student Details(b)

STUDENT MANAGEMENT AND MONITORING SYSTEM



```
Command Prompt - python manage.py runserver

E:\classroom_monitoring_system>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
June 07, 2022 - 00:07:46
Django version 3.1.7, using settings 'classroom_monitoring_system.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Fig 6.3.3 Running Server

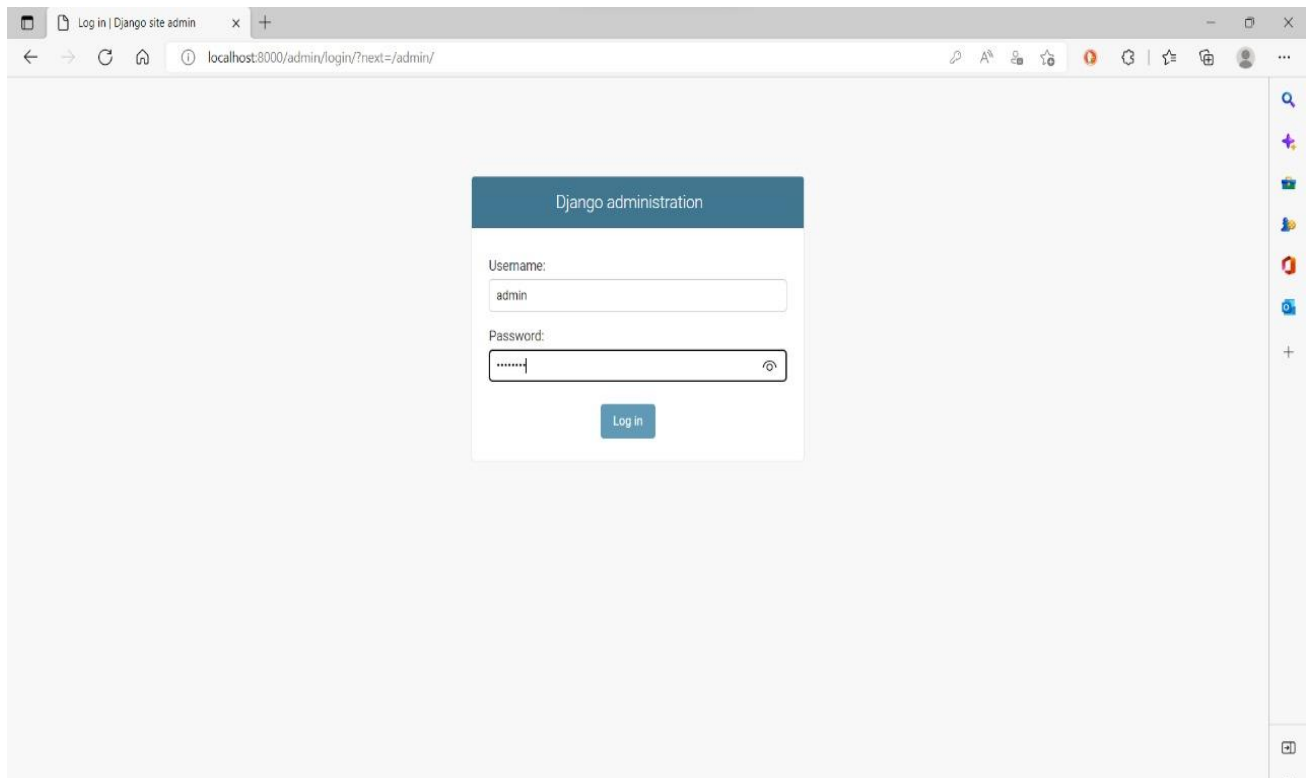


Fig 6.3.4 Admin Login

STUDENT MANAGEMENT AND MONITORING SYSTEM

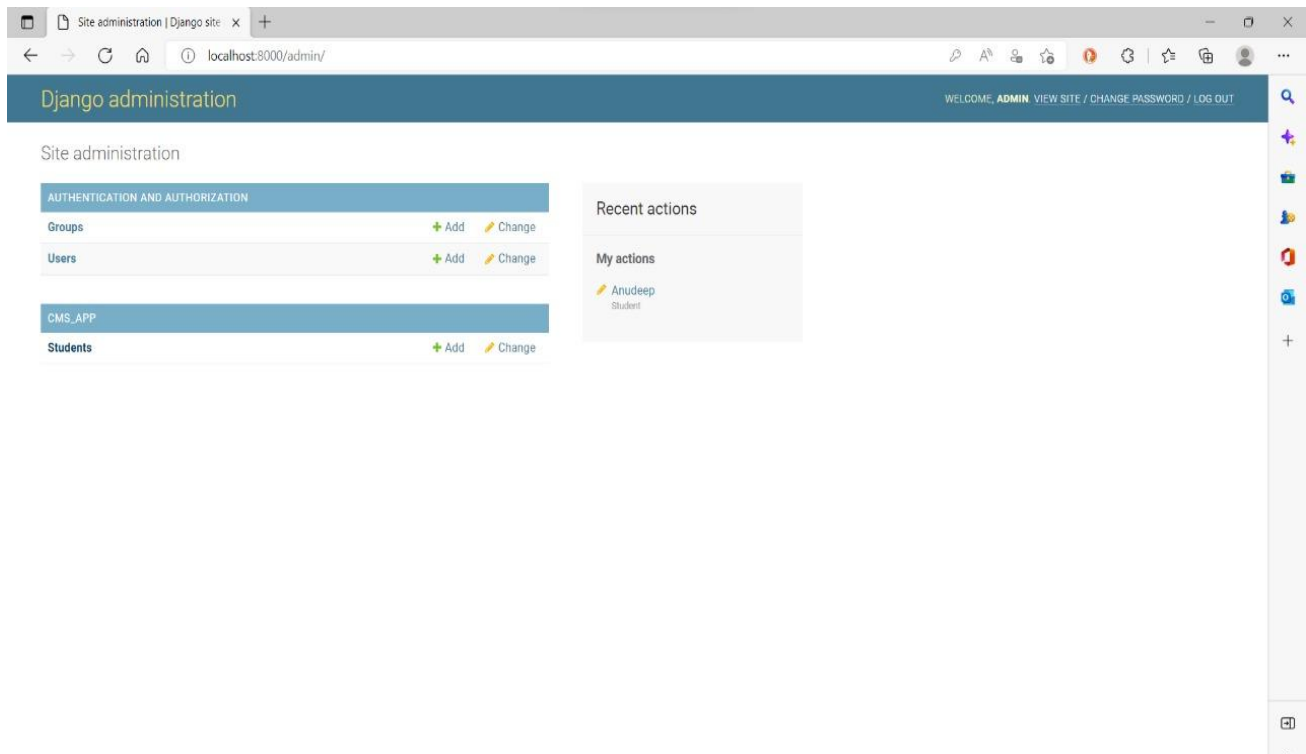


Fig 6.3.5 Admin Homepage

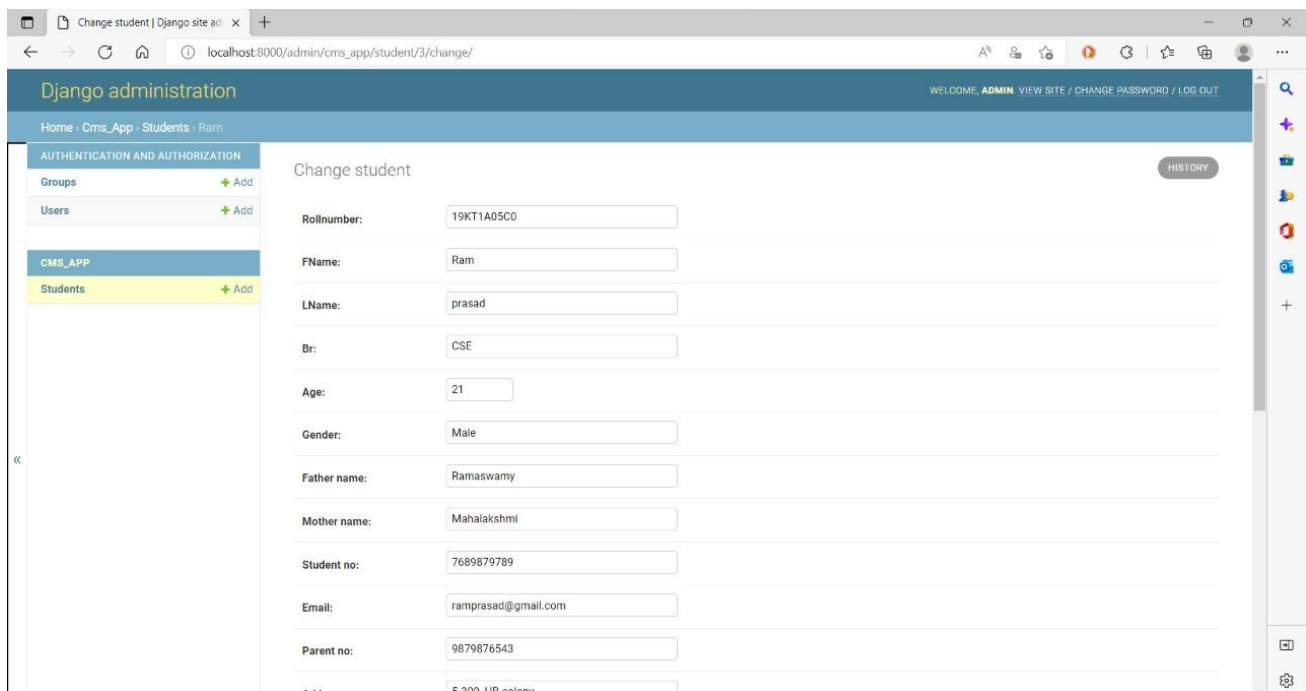


Fig 6.3.6 Viewing of Entered student details(a)

STUDENT MANAGEMENT AND MONITORING SYSTEM

The screenshot shows a web browser window with the URL `localhost:8000/admin/cms_app/student/3/change/`. The page is titled "Change student | Django site admin". On the left, a sidebar menu shows "AUTHENTICATION AND AUTHORIZATION" with "Groups" and "Users" links, and "CMS_APP" with "Students" selected. The main content area displays a form for editing a student's details. The form fields are as follows:

Field	Value
Email:	ramprasad@gmail.com
Parent no:	9879876543
Address:	5-309, HB colony
Tenth:	9.8
Inter:	9.7
Btech:	8.7
Backlog:	0
Hobbies:	Playing cricket, Cooking
City:	Vijayawada
Pcode:	522501
State:	Andhra Pradesh
Cntry:	India

At the bottom of the form, there are four buttons: "Delete" (red), "Save and add another" (blue), "Save and continue editing" (blue), and "SAVE" (blue).

Fig 6.3.7 Viewing of Entered student details(b)

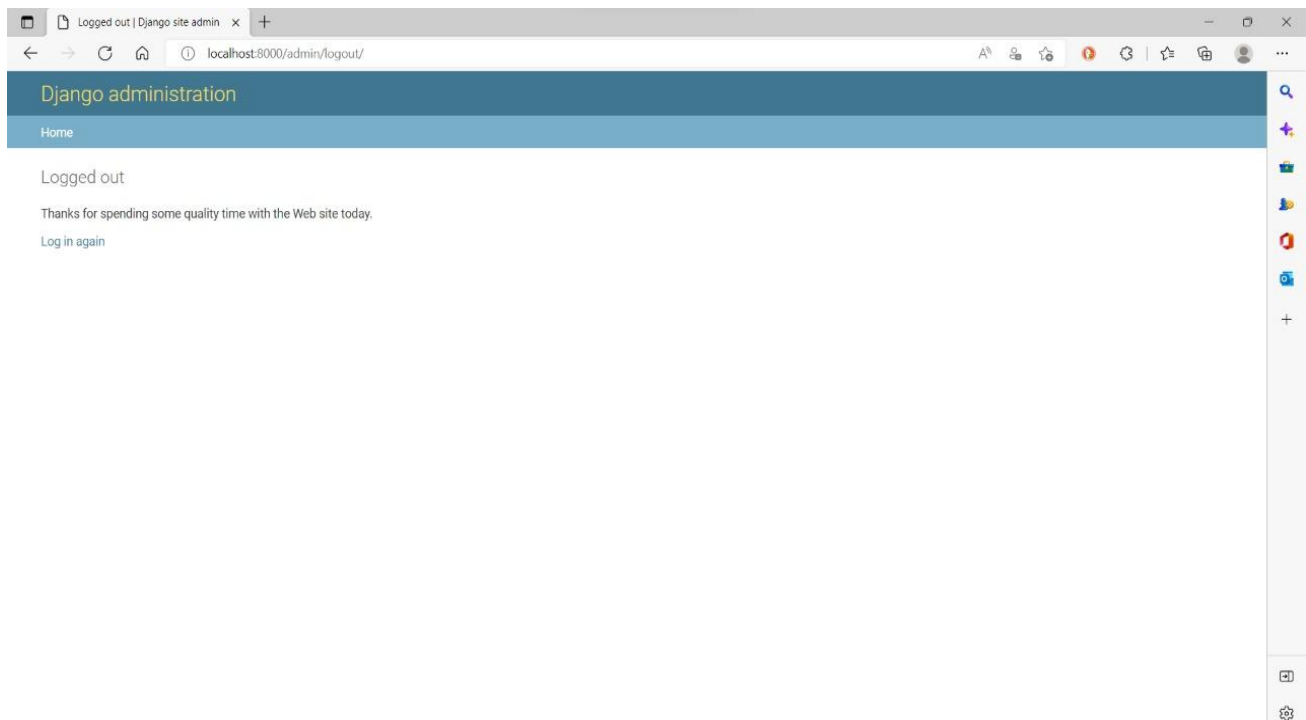


Fig 6.3.8 Logging out from Admin Page

TESTING

7. TESTING

7.1 About Testing

Software Testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs (errors or other defects). It involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:

- Meets the requirements that guided its design and development
- Responds correctly to all kinds of inputs
- Performs its functions within an acceptable time.
- Is sufficiently usable.
- Can be installed and run in its intended environments

As the number of possible tests for even simple software components is practically infinite, all software testing uses some strategy to select tests that are feasible for the available time and resources. As a result, software testing typically (but not exclusively) attempts to execute a program or application with the intent of finding software bugs (errors or other defects).

Software Testing can provide objective, independent information about the quality of software and risk of its failure to users and/or sponsors. Software testing can be conducted as soon as executable software (even if partially complete) exists. The overall approach to software development often determines when and how testing is conducted. For example, in a phased process, most testing occurs after system requirements have been defined and then implemented in testable programs. In contrast, under an Agile approach, requirements, programming, and testing are often done concurrently.

7.2 TESTING METHODS (levels of Testing)

7.2.1 Static vs. Dynamic Testing

There are many approaches in software testing. Reviews, walkthroughs, or inspections are referred to as static testing, whereas actually executing programmed code with a given set of test cases is referred to as dynamic testing. Static testing is often implicit, as proofreading, plus when programming tools/text editors check source code structure or compilers (pre compilers) check syntax and data flow as static program analysis. Dynamic testing takes place when the program itself is run. Dynamic testing may begin before the program is 100% complete in order to test particular sections of code and are applied to discrete functions or modules. Typical techniques for this are either using stubs/drivers or execution from a debugger environment.

Static testing involves verification, whereas dynamic testing involves validation. Together they help improve software quality. Among the techniques for static analysis, mutation testing can be used to ensure the test-cases will detect errors which are introduced by mutating the source code.

7.2.2 The Box Approach

Software testing methods are traditionally divided into white- and black-box testing. These two approaches are used to describe the point of view that a test engineer takes when designing test cases.

7.2.2.1 White-Box Testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing and structural testing) tests internal structures or workings of a program, as opposed to the functionality exposed to the end-user. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT).

While white-box testing can be applied at the unit, integration and system levels of the software testing process, it is usually done at the unit level. It can test paths within a unit, paths between units during integration, and between subsystems during a system-level test. Though this method of test design can uncover many errors or problems, it might not detect unimplemented parts of the specification or missing requirements.

Techniques used in white-box testing include:

- API testing: It tests the application using public and private APIs.
- Code coverage: It creates tests to satisfy some criteria of code coverage.

Code coverage tools can evaluate the completeness of a test suite that was created with any method, including black-box testing. This allows the software team to examine parts of a system that are rarely tested and ensures that the most important function points have been tested.

Code coverage as a software metric can be reported as a percentage for:

- Function coverage, which reports on functions executed
 - Statement coverage, which reports on the number of lines executed to complete the test
- 100% statement coverage ensures that all code paths or branches (in terms of control flow) are executed at least once. This is helpful in ensuring correct functionality, but not sufficient since the same code may process different inputs correctly or incorrectly.

7.2.2.2 Black-Box Testing

Black-box testing treats the software as a "black box", examining functionality without any knowledge of internal implementation. The testers are only aware of what the software is supposed to do, not how it does it. Black-box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, state transition tables, decision table testing, fuzz testing, model-based testing, use case testing, exploratory testing and specification-based testing.

Specification-based testing aims to test the functionality of software according to the applicable requirements. This level of testing usually requires thorough test cases to be provided to the tester, who then can simply verify that for a given input, the output value (or behavior), either "is" or "is not" the same as the expected value specified in the test case. Test cases are built around specifications and requirements, i.e., what the application is supposed to do. It uses external descriptions of the

software, including specifications, requirements, and designs to derive test cases. These tests can be functional or non-functional, though usually functional.

Specification-based testing may be necessary to assure correct functionality, but it is insufficient to guard against complex or high-risk situations. One advantage of the black box technique is that no programming knowledge is required. Whatever biases the programmers may have had, the tester likely has a different set and may emphasize different areas of functionality. On the other hand, black-box testing has been said to be “like a walk in a dark labyrinth without a flashlight.” Because they do not examine the source code, there are situations when a tester writes many test cases to check something that could have been tested by only one test case, or leaves some parts of the program untested. This method of test can be applied to all levels of software testing: unit, integration, system and acceptance. It typically comprises most if not all testing at higher levels, but can also dominate unit testing as well.

7.2.3 Testing Levels

There are generally four recognized levels of tests: unit testing, integration testing, system testing, and acceptance testing. Tests are frequently grouped by where they are added in the software development process, or by the level of specificity of the test. The main levels during the development process as defined by the SWEBOK guide are unit-, integration-, and system testing that are distinguished by the test target without implying a specific process model. Other test levels are classified by the testing objective.

.
Client Needs
Acceptance Testing
Requirements
System Testing
Design
Integration Testing
Code
Unit Testing

7.2.3.1 Unit Testing

Unit testing, also known as component testing, refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors. These types of tests are usually written by developers as they work on code (white-box style), to ensure that the specific function is working as expected. One function might have multiple tests, to catch corner cases or other branches in the code. Unit testing alone cannot verify the functionality of a piece of software, but rather is used to ensure that the building blocks of the software work independently from each other. Unit testing is a software development process that involves synchronized application of a broad spectrum of defect prevention and detection strategies in order to reduce software development risks, time, and costs. It is performed by the software developer or engineer during the construction phase of the software development lifecycle. Rather than replace traditional QA focuses, it augments it. Unit testing aims to eliminate construction errors before code is promoted to QA; this strategy is intended to increase the quality of the resulting software as well as the efficiency of the overall development and QA process. Depending on the organization's expectations for software development, unit testing might include static code analysis, data flow

analysis, metrics analysis, peer code reviews, code coverage analysis and other software verification practices.

7.2.3.2 Integration Testing

Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design. Software components may be integrated in an iterative way or all together (“big bang”). Normally the former is considered a better practice since it allows interface issues to be located more quickly and fixed. Integration testing works to expose defects in the interfaces and interaction between integrated components (modules). Progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a system.

7.2.3.3 Component Interface Testing

The practice of component interface testing can be used to check the handling of data passed between various units, or subsystem components, beyond full integration testing between those units. The data being passed can be considered as “message packets” and the range or data types can be checked, for data generated from one unit, and tested for validity before being passed into another unit. One option for interface testing is to keep a separate log file of data items being passed, often with a time stamp logged to allow analysis of thousands of cases of data passed between units for days or weeks. Tests can include checking the handling of some extreme data values while other interface variables are passed as normal values. Unusual data values in an interface can help explain unexpected performance in the next unit. Component interface testing is a variation of black-box testing, with the focus on the data values beyond just the related actions of a subsystem component.

7.2.3.4 System Testing

System testing, or end-to-end testing, tests a completely integrated system to verify that it meets its requirements. For example, a system test might involve testing a logon interface, then creating and editing an entry, plus sending or printing results, followed by summary processing or deletion (or archiving) of entries, then logoff. In addition, the software testing should ensure that the program, as well as working as expected, does not also destroy or partially corrupt its operating environment or cause other processes within that environment to become inoperative (this includes not corrupting shared memory, not consuming or locking up excessive resources and leaving any parallel processes unharmed by its presence). Software testing verification and validation are the most important components to be considered. In this article we will discuss the details about verification and validation part of software testing.

7.3 VALIDATION AND VERIFICATION

Software verification and validation actions confirm the software aligned with its terms. All assignment should validate and verify the software it produces. This is made by:

- Verify that every software object meets particular necessities.
- Verify all software objects it is used as a key to a new action.
- Ensure that checking every software entities are done, as possible, by someone else other than the developer.
- Ensure that the sum of validation and verification effort is enough to explain every software objects are appropriate for equipped use.

Project management is accountable for organizing the classification of software verification and validation roles, software verification and validation behaviors and the allotment of employees to those roles. Whatsoever the volume of project, software verification and validation very much affects software value. Populace is not reliable, and software that has not been confirmed has little possibility of functioning. Characteristically, 2%-5% errors per 1000 lines of code are found through development and 0.15%-4% error per 1000 lines of code remain still after testing of the system. Every error might lead to an equipped breakdown or non-cooperation with a necessity. The purpose of software verification and validation is to decrease software errors to a satisfactory level. The effort wanted can vary from 30%-90% of the whole project property, depend upon the difficulty and criticality of the software. The following diagram shows the process flow.

7.3.1 Verification in Software Testing:

Now let us see what the verification does in a software testing.

- Verification makes ensure that the result is intended to give all functionality to the client.
- It is done at the initial of the development method. It includes reviews and meetings, walkthroughs, check, etc. to assess credentials, strategy, code, necessities and stipulation.
- It ensures for building a right product.
- It also checks for accessing the data correct in the correct place and in the correct way.
- Verification is a LowLevel action.
- It is performed at the time of development on walkthroughs, reviews and inspections, adviser comment, guidance, checklists and principles.
- Manifestation of reliability, wholeness, and accuracy of the software at every phase and among every phase of the development life cycle.

According to the Capability Maturity Model, we can also describe verification as the method of evaluating software to establish whether the yield of a particular development stage please the situation forced at the beginning of that stage.

7.3.2 Validation in Software Testing:

Following points discuss about the validation process in software testing

- Validation determines how the system compiles with the necessities and performs functions for which it is proposed and meets the organization's goals and user requirements
- It is for building a right product.
- It also checks for accessing the accurate data.
- Validation is a HighLevel action
- Also determines exactness of the ultimate software product by an advanced project with respect to the consumer desires and necessities.
- According to the capability maturity model we can also describe validation as procedure of evaluate software at the time of development procedure to establish whether it satisfies particular necessities.

CONCLUSION

8. CONCLUSION

Our project is only a humble venture to satisfy the needs in an Institution. Several user-friendly coding has also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the organization. The objective of software planning is to provide a frame work that enables the manger to makereasonable estimates made within a limited time frame at the beginning of the software project andshould be updated regularly as the project progresses. Last but not least it is no the work that played theways to success but ALMIGHTY

Student information management system lead to a better organization structure since the information management of the students is well structured and also lead to better as well as efficient utilization of resources. Student Information Management System can be used by education institutes to maintain the records of students easily. Achieving this objective is difficult using a manual system as the information is scattered, can be redundant and collecting relevant information may be very time consuming. All these problems are solved using this project

REFERENCES

9. REFERENCES

- [1] Zhibing Liu, Huixia Wang, Hui Zan “Design and implementation of student information management system.” 2010 International symposium on intelligence information processing and trusted computing. 978-0-7695- 4196-9/10 IEEE.
- [2] Zhi-gang YUE, You-wei JIN, “The development and design of the student management system based on the network environment”, 2010 International Conference on Multimedia Communications, 978-0-7695- 4136-5/10 2010 IEEE.
- [3] TANG Yu-fang, ZHANG Yong-sheng, “Design and implementation of college student information management system based on the web services”. Natural Science Foundation of Shandong Province(Y2008G22), 978-1-4244-3930-0/09 2009 IEEE.
- [4] M.A. Norasiah and A. Norhayati. “Intelligent student information system”. 4th International conference on telecommunication technology proceedings, Shah Alam, Malaysia, 0-7803-7773-7/03 2003 IEEE.
- [5]
- [6] [5] Jin Mei-shan¹ Qiu Chang-li² Li Jing³. “The Designment of student information management system based on B/S architecture”. 978-1- 4577-1415-3/12 2012 IEEE.
- [2] Newman-Ford, L.E., Fitzgibbon, K., Llyod, S. & Thomas, S.L., “A Large-Scale Investigation into the Relationship between Attendance and Attainment: A Study Using an Innovative, Electronic Attendance Monitoring System”, *Studies in Higher Education*, 33(6), pp. 699-717, 2008
- [3]
- [4] 2. Marr, Liz & Lancaster, Guy, “Attendance System”, *Learning and Teaching in Action*, 4 (1), pp. 21-26, 2005
- [5]
- [6] 3. Mazza, R. & Dimitrova, V., “Visualising student tracking data to support instructors in web-based distance education”, *Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters Press*, pp.154-161, New York: USA, 2004
- [7]
- [8] 4. Mehmet Kizildag, Erden Basar, Murude Celikag, Emine Atasoylu and Sayedali Mousavi, “An Automated Attendance Monitoring and Registration System for EMU’s SPIKE Seminar Series”, *Proceedings in Academia.edu*.
- [9]
- [10] 5. RESEARCH NOTE, AUTOMATING TIME AND ATTENDANCE: LOW HANGING ROI, *Proceeding in Nucleus Research*, January 2008.
- [11]
- [12] 6. S. K. Jain, U. Joshi, and B. K. Sharma, “Attendance Management System,” *Masters Project Report, Rajasthan Technical University, Kota*.
- [13]
- [14] 7. M. Mattam, S. R. M. Karumuri, and S. R. Meda, “Architecture for Automated Student Attendance,” in *Proc. IEEE Fourth International Conference on Technology for Education (T4E 2012)*, pp.164-167, 18-20 July 2012, doi: 10.1109/T4E.2012.39.
- [15]
- [16] 8. M. Strommer et al., Smart NFC Interface Platform and its Applications, in T. Tuikka and M.

Isomursu, (Eds.), Touch the Future with a Smart Touch, 2009

[17]

[18] 9. M. K. P. Basheer and C. V. Raghu, "Fingerprint attendance system for classroom needs," in Proc. India Conference (INDICON), 2012 Annual IEEE, pp. 433-438, 7-9 Dec. 2012.

[19]

[20] 10. BISAM-BIS attendance Management System by BIS Software Development Services PVT Limited. [Online]. Available: <http://www.softwarehouse.co/school-attendance-brochure.pdf>

[21]

[22] 11. S.-H. Geng, G.-M. Li, and W. Liu, "Design and Implement of Attendance Management System Based on Contactless Smart IC Card," in Proc. International Conference on Computer Science and Electronics Engineering (ICCSEE), vol. 3, pp. 290-294, 23-25 March 2012, doi: 10.1109/ICCSEE.2012.196.

[23]

[24] 12. T. S. Lim, S. C. Sim, and M. M. Mansor, "RFID based attendance system," IEEE Symposium on Industrial Electronics & Applications 2009 (ISIEA 2009), vol.2, pp. 778-782, 4-6 Oct. 2009, doi: 10.1109/ISIEA.2009.5356360.

[25]

[26] 13. M. Kassim, H. Mazlan, N. Zaini, and M. K. Salleh, "Web-based student attendance system using RFID technology," in Proc. IEEE Control and System Graduate Research Colloquium (ICSGRC 2012), pp. 213-218, 16-17 July 2012, doi: 10.1109/ICSGRC.2012.6287164.

[27]

[28] 14. Vishal Bhalla, Tapodhan Singla, Ankit Gahlot and Vijay Gupta, "Bluetooth Based Attendance Management System", International Journal of Innovations in Engineering and Technology (IJET) Vol. 3 Issue 1 October 2013, ISSN: 2319 – 1058.

[29]

[30] 15. Josphineleela.R and Dr.M.Ramakrishnan, "An Efficient Automatic Attendance System Using Fingerprint Reconstruction Technique", (IJCSIS) International Journal of Computer Science and Information Security, Vol. 10, No. 3, March 2012.

[31]

[32] 16. Seema Rao and Prof.K.J.Satoa, "An Attendance Monitoring System Using Biometrics Authentication", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 4, April 2013, ISSN: 2277 128X.

[33]

[34] 17. Neha Verma, Komal Sethi and Megha Raghav, "AN EFFICIENT AUTOMATIC ATTENDANCE SYSTEM USING FINGERPRINT RECONSTRUCTION TECHNIQUE", International Journal Of Advance Research In Science And Engineering (IJARSE), Vol. No.2, Issue No.3, March, 2013, ISSN-2319-8354(E).

[35]

[36] 18. APARNA BEHARA and M.V. RAGHUNADH, "REAL TIME FACE RECOGNITION SYSTEM FOR TIME AND ATTENDANCE APPLICATIONS", International Journal of Electrical, Electronics and Data Communication, Volume- 1, Issue- 4, ISSN: 2320-2084.

[37]

[38] 19. MuthuKalyani.K and VeeraMuthu.A, "SMART APPLICATION FOR AMS USING FACE RECOGNITION", Computer Science & Engineering: An International Journal (CSEIJ), Vol. 3, No. 5, October 2013, DOI : 10.5121/cseij.2013.3502.

[39]

[40] 20. Seifedine Kadry* and Mohamad Smaili, "Wireless attendance management system based on iris Recognition", Scientific Research and Essays Vol. 5(12), pp. 1428-1435, 18 June, 2010, ISSN

APPENDIX

10. APPENDIX

DJANGO

Django is a Python framework that makes it easier to create web sites using Python.

Django takes care of the difficult stuff so that you can concentrate on building your web applications.

Django emphasizes reusability of components, also refereed to as DRY (Don't Repeat Yourself), and comes with ready-to-use features like login system, database connection and CRUD operations (Create Read Update Delete).

Django follows the MVT design pattern (Model View Template).

- Model - The data you want to present, usually data from a database.
- View - A request handler that returns the relevant template and content - based on the request from the user.
- Template - A text file (like an HTML file) containing the layout of the web page, with logic on how to display the data.

The model provides data from the database.

In Django, the data is delivered as an Object Relational Mapping (ORM), which is a technique designed to make it easier to work with databases.

The most common way to extract data from a database is SQL. One problem with SQL is that you have to have a pretty good understanding of the database structure to be able to work with it.

Django, with ORM, makes it easier to communicate with the database, without having to write complex SQL statements.

The models are usually located in a file called models.py.

A view is a function or method that takes http requests as arguments, imports the relevant model(s), and finds out what data to send to the template, and returns the final result.

The views are usually located in a file called views.py.

A template is a file where you describe how the result should be represented.

Templates are often .html files, with HTML code describing the layout of a web page, but it can also be in other file formats to present other results, but we will concentrate on .html files.

Django uses standard HTML to describe the layout, but uses Django tags to add logic:

```
<h1>My Homepage</h1>
```

```
<p>My name is {{ firstname }}.</p>
```

The templates of an application is located in a folder named templates.

URLs

Django also provide a way to navigate around the different pages in a website.

When a user requests a URL, Django decides which view it will send it to.

This is done in a file called urls.py.

Django is a high-level Python Web framework that encourages rapid development and clean pragmatic design. A Web framework is a set of components that provide a standard way to develop websites fast and easily. Django's primary goal is to ease the creation of complex database-driven websites. Some well known sites that use Django include PBS, Instagram, Disqus, Washington Times, Bitbucket and Mozilla.

- Testing is vital. The articles on testing will introduce you to unit and integration testing for your Django applications. You will also learn about the different packages and libraries available to assist with writing and running test suites.

REST API

- Learn how to create RESTful APIs using the Django Rest Framework(DRF), an application used for rapidly building RESTful APIs based on Django models.

Best practices

- Learn Django best practices, recommended workflow, project structure and also how to avoid common pitfalls when building Django projects.

Deployment

- When your application is ready to leave the room and be deployed, the tutorials and articles on deployment will cover deployment options available to you and how to deploy your site to each one.

Caching

- Fast page loads improve the experience of visiting your site. Here you'll learn about factors that slow web applications down and how you can boost performance by implementing caching.

Django is a widely-used Python web application framework with a "batteries-included" philosophy. The principle behind batteries-included is that the common functionality for building web applications should come with the framework instead of as separate libraries.

STUDENT MANAGEMENT AND MONITORING SYSTEM

The Django project's stability, performance and community have grown tremendously over the past decade since the framework's creation. Detailed tutorials and good practices are readily available on the web and in books. The framework continues to add significant new functionality such as database migrations with each release.

I highly recommend the Django framework as a starting place for new Python web developers because the official documentation and tutorials are some of the best anywhere in software development. Many cities also have Django-specific groups such as Django District, Django Boston and San Francisco Django so new developers can get help when they are stuck.

History of Django

2003 – Started by Adrian Holovaty and Simon Willison as an internal project at the Lawrence Journal-World newspaper.

2005 – Released July 2005 and named it Django, after the jazz guitarist Django Reinhardt.

2005 – Mature enough to handle several high-traffic sites.

Current – Django is now an open source project with contributors across the world.

Django – Design Philosophies

Django comes with the following design philosophies –

Loosely Coupled – Django aims to make each element of its stack independent of the others.

Less Coding – Less code so in turn a quick development.

Don't Repeat Yourself (DRY) – Everything should be developed only in exactly one place instead of repeating it again and again.

Fast Development – Django's philosophy is to do all it can to facilitate hyper-fast development.

Clean Design – Django strictly maintains a clean design throughout its own code and makes it easy to follow best web-development practices.

Advantages of Django

STUDENT MANAGEMENT AND MONITORING SYSTEM

Here are few advantages of using Django which can be listed out here –

Object-Relational Mapping (ORM) Support – Django provides a bridge between the data model and the database engine, and supports a large set of database systems including MySQL, Oracle, Postgres, etc. Django also supports NoSQL database through Django-nonrel fork. For now, the only NoSQL databases supported are MongoDB and google app engine.

Multilingual Support – Django supports multilingual websites through its built-in internationalization system. So you can develop your website, which would support multiple languages.

Framework Support – Django has built-in support for Ajax, RSS, Caching and various other frameworks.

Administration GUI – Django provides a nice ready-to-use user interface for administrative activities.

Development Environment – Django comes with a lightweight web server to facilitate end-to-end application development and testing

As you already know, Django is a Python web framework. And like most modern framework, Django supports the MVC pattern. First let's see what is the Model-View-Controller (MVC) pattern, and then we will look at Django's specificity for the Model-View-Template (MVT) pattern.

MVC Pattern

When talking about applications that provides UI (web or desktop), we usually talk about MVC architecture. And as the name suggests, MVC pattern is based on three components: Model, View, and Controller. Check our MVC tutorial here to know more.

DJANGO MVC - MVT Pattern

The Model-View-Template (MVT) is slightly different from MVC. In fact the main difference between the two patterns is that Django itself takes care of the Controller part (Software Code that controls the interactions between the Model and View), leaving us with the template. The template is a HTML file mixed with Django Template Language (DTL).

	Andhra Pradesh State Skill Development Corporation (APSSDC) (Department of Skills Development & Training, Govt. of Andhra Pradesh)	
Certificate of Participation		
Cert.No. SDC/DJPY1357	Regd Id : 19KT1A05A2	
This is to certify that Mr./Ms./Mrs.....Medikondur Nithin.....		
ofPSCMR College of Engineering and Technology..... has successfully		
participated Training Program onWeb Development Using Django.....		
held from17-02-2022.....to24-02-2022.....		
Principal (or) HOD	 Dr. Ravi K Gujjula Chief General Manager (Technical) APSSDC	 Sri S. Satyanarayana, IAS Managing Director APSSDC