Backend & Data Layer

Software Design Foundations

Author: Eng. Carlos Andrés Sierra, M.Sc.
cavirguezs@udistrital.edu.co

Computer Engineer
Lecturer
Universidad Distrital Francisco José de Caldas

2024-III

Outline

Outline

Data System Concepts

Key Points of Data Systems:

- Data modeling is the process of designing the **structure** and organization of data.
- Data storage is the process of storing data in a structured or unstructured format.
- Data retrieval is the process of accessing and retrieving data from a storage system.
- Data manipulation is the process of modifying and **transforming** data.
- Data security is the process of protecting data from unauthorized access and ensuring its **integrity** and **confidentiality**.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Data System Concepts

Key Points of Data Systems:

- Data modeling is the process of designing the **structure** and organization of data.
- Data storage is the process of storing data in a structured or unstructured format.
- Data retrieval is the process of accessing and retrieving data from a storage system.
- Data manipulation is the process of modifying and **transforming** data.
- Data security is the process of protecting data from unauthorized access and ensuring its **integrity** and **confidentiality**.

Data System Concepts

Key Points of Data Systems:

- Data modeling is the process of designing the **structure** and organization of data.
- Data storage is the process of storing data in a structured or unstructured format.
- Data retrieval is the process of accessing and retrieving data from a storage system.
- Data manipulation is the process of modifying and **transforming** data.
- Data security is the process of protecting data from unauthorized access and ensuring its **integrity** and **confidentiality**.

Data System Concepts

Key Points of Data Systems:

- Data modeling is the process of designing the **structure** and organization of data.
- Data storage is the process of storing data in a structured or unstructured format.
- Data retrieval is the process of accessing and retrieving data from a storage system.
- Data manipulation is the process of modifying and **transforming** data.
- Data security is the process of protecting data from unauthorized access and ensuring its **integrity** and **confidentiality**.

Data System Concepts

Key Points of Data Systems:

- Data modeling is the process of designing the **structure** and organization of data.
- Data storage is the process of storing data in a structured or unstructured format.
- Data retrieval is the process of accessing and retrieving data from a storage system.
- Data manipulation is the process of modifying and **transforming** data.
- Data security is the process of protecting data from unauthorized access and ensuring its **integrity** and **confidentiality**.

Relational Databases

- A database management system (DBMS) is a software system that uses a standard method to **store** and **retrieve** data.
- A relational database management system (RDBMS) is a type of database management system that stores data in a **structured format**, using `rows` and `columns`.
- An entity-relationship diagram (ERD) is a data modeling technique that graphically represents an **information** system's entities and the relationships between them.
- SQL is a domain-specific language used in programming and designed for managing data held in a **relational database** management system.

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Relational Databases

- A database management system (DBMS) is a software system that uses a standard method to **store** and **retrieve** data.
- A relational database management system (RDBMS) is a type of database management system that stores data in a **structured format**, using `rows` and `columns`.
- An entity-relationship diagram (ERD) is a data modeling technique that graphically represents an **information** system's entities and the relationships between them.
- SQL is a domain-specific language used in programming and designed for managing data held in a **relational database** management system.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Relational Databases

- A database management system (DBMS) is a software system that uses a standard method to **store** and **retrieve** data.
- A relational database management system (RDBMS) is a type of database management system that stores data in a **structured format**, using `rows` and `columns`.
- An entity-relationship diagram (ERD) is a data modeling technique that graphically represents an **information** system's entities and the relationships between them.
- SQL is a domain-specific language used in programming and designed for managing data held in a **relational database** management system.

Relational Databases

- A database management system (DBMS) is a software system that uses a standard method to **store** and **retrieve** data.
- A relational database management system (RDBMS) is a type of database management system that stores data in a **structured format**, using `rows` and `columns`.
- An entity-relationship diagram (ERD) is a data modeling technique that graphically represents an **information** system's entities and the relationships between them.
- SQL is a domain-specific language used in programming and designed for managing data held in a **relational database** management system.

ER Diagrams

- An entity-relationship diagram (ERD) is a data modeling technique that graphically represents an **information** system's entities and the relationships between them.
- An entity is a real-world `object` or concept that has a **unique identity**, such as a person, place, or thing.
- An attribute is a **property** or characteristic of an `entity`, such as a person's name or age.
- A relationship is a **connection** between two or more **entities**, such as a person's relationship to a company or a product's relationship to a customer.
- A cardinality is a constraint that specifies the number of **instances** of one entity that can be associated with the number of instances of another entity.

ER Diagrams

- An entity-relationship diagram (ERD) is a data modeling technique that graphically represents an **information** system's entities and the relationships between them.
- An entity is a real-world `object` or concept that has a **unique identity**, such as a person, place, or thing.
- An attribute is a **property** or characteristic of an `entity`, such as a person's name or age.
- A relationship is a **connection** between two or more **entities**, such as a person's relationship to a company or a product's relationship to a customer.
- A cardinality is a constraint that specifies the number of **instances** of one entity that can be associated with the number of instances of another entity.

ER Diagrams

- An entity-relationship diagram (ERD) is a data modeling technique that graphically represents an **information** system's entities and the relationships between them.
- An entity is a real-world `object` or concept that has a **unique identity**, such as a person, place, or thing.
- An attribute is a **property** or characteristic of an `entity`, such as a person's name or age.
- A relationship is a **connection** between two or more **entities**, such as a person's relationship to a company or a product's relationship to a customer.
- A cardinality is a constraint that specifies the number of **instances** of one entity that can be associated with the number of instances of another entity.

ER Diagrams

- An entity-relationship diagram (ERD) is a data modeling technique that graphically represents an **information** system's entities and the relationships between them.
- An entity is a real-world `object` or concept that has a **unique identity**, such as a person, place, or thing.
- An attribute is a **property** or characteristic of an `entity`, such as a person's name or age.
- A relationship is a **connection** between two or more **entities**, such as a person's relationship to a company or a product's relationship to a customer.
- A cardinality is a constraint that specifies the number of **instances** of one entity that can be associated with the number of instances of another entity.

ER Diagrams

- An entity-relationship diagram (ERD) is a data modeling technique that graphically represents an **information** system's entities and the relationships between them.
- An entity is a real-world `object` or concept that has a **unique identity**, such as a person, place, or thing.
- An attribute is a **property** or characteristic of an `entity`, such as a person's name or age.
- A relationship is a **connection** between two or more **entities**, such as a person's relationship to a company or a product's relationship to a customer.
- A cardinality is a constraint that specifies the number of **instances** of one entity that can be associated with the number of instances of another entity.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Study Case: ER Diagram for an Academic System

Data Access Objects and Data Transfer Objects

Data Access Objects (DAOs) and Data Transfer Objects (DTOs) are design patterns used to separate the data access logic from the business logic in an application.

- A Data Access Object (DAO) is an object that provides **an abstract interface** to some type of `database` or other persistence mechanism.
- A Data Transfer Object (DTO) is an object that **carries data** between processes in an application.
- The DAO pattern is used to **separate the** `data access logic` from the `business logic` in an application.
- The DTO pattern is used to **transfer data** between processes in an application.

Data Access Objects and Data Transfer Objects

Data Access Objects (DAOs) and Data Transfer Objects (DTOs) are design patterns used to separate the data access logic from the business logic in an application.

- A Data Access Object (DAO) is an object that provides **an abstract interface** to some type of `database` or other persistence mechanism.
- A Data Transfer Object (DTO) is an object that **carries data** between processes in an application.
- The DAO pattern is used to **separate the** `data access logic` from the `business logic` in an application.
- The DTO pattern is used to **transfer data** between processes in an application.

Data Access Objects and Data Transfer Objects

Data Access Objects (DAOs) and Data Transfer Objects (DTOs) are design patterns used to separate the data access logic from the business logic in an application.

- A Data Access Object (DAO) is an object that provides **an abstract interface** to some type of `database` or other persistence mechanism.
- A Data Transfer Object (DTO) is an object that **carries data** between processes in an application.
- The DAO pattern is used to **separate the** `data access logic` from the `business logic` in an application.
- The DTO pattern is used to **transfer data** between processes in an application.

Object-Relational Mapping

- Object-Relational Mapping (ORM) is a programming technique that **converts data** between incompatible type systems using `object-oriented programming languages`.
- An ORM framework is a tool that automates the process of mapping objects to relational databases.
- ORM frameworks include features such as data validation, data retrieval, and data manipulation.
- ORM frameworks lets you work with data in an object-oriented way, rather than in a relational way.

Object-Relational Mapping

- Object-Relational Mapping (ORM) is a programming technique that **converts data** between incompatible type systems using `object-oriented programming languages`.
- An ORM framework is a tool that automates the process of mapping objects to relational databases.
- ORM frameworks include features such as data validation, data retrieval, and data manipulation.
- ORM frameworks lets you work with data in an object-oriented way, rather than in a relational way.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

PostgreSQL and SQLAlchemy

- PostgreSQL is a powerful, **open-source object-relational database** system.
- SQLAlchemy is an **open-source SQL toolkit** and Object-Relational Mapping (ORM) library for Python.
- SQLAlchemy provides a full suite of well-known **enterprise-level** persistence patterns, designed for efficient and high-performing database access.

PostgreSQL and SQLAlchemy

- PostgreSQL is a powerful, **open-source object-relational database** system.
- SQLAlchemy is an **open-source SQL toolkit** and Object-Relational Mapping (ORM) library for Python.
- SQLAlchemy provides a full suite of well-known **enterprise-level** persistence patterns, designed for efficient and high-performing database access.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Outline

Backend Concepts

Key Points of Backend Systems:

- A backend system is a software system that provides the **logic** and functionality to support the front-end of an application.
- A backend system typically consists of a `server`, a `database`, and an `application server`.
- A server is a computer that provides **services** to other computers over a network.
- An application server is a software framework that provides an environment for **running web applications**.
- A database is a collection of data that is **organized and stored** in a defined format.

Backend Concepts

Key Points of Backend Systems:

- A backend system is a software system that provides the **logic** and functionality to support the front-end of an application.
- A backend system typically consists of a `server`, a `database`, and an `application server`.
- A server is a computer that provides **services** to other computers over a network.
- An application server is a software framework that provides an environment for **running web applications**.
- A database is a collection of data that is **organized and stored** in a defined format.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Backend Concepts

Key Points of Backend Systems:

- A backend system is a software system that provides the **logic** and functionality to support the front-end of an application.
- A backend system typically consists of a `server`, a `database`, and an `application server`.
- A server is a computer that provides **services** to other computers over a network.
- An application server is a software framework that provides an environment for **running web applications**.
- A database is a collection of data that is **organized and stored** in a defined format.

Backend Concepts

Key Points of Backend Systems:

- A backend system is a software system that provides the **logic** and functionality to support the front-end of an application.
- A backend system typically consists of a `server`, a `database`, and an `application server`.
- A server is a computer that provides **services** to other computers over a network.
- An application server is a software framework that provides an environment for **running web applications**.
- A database is a collection of data that is **organized and stored** in a defined format.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Backend Concepts

Key Points of Backend Systems:

- A backend system is a software system that provides the **logic** and functionality to support the front-end of an application.
- A backend system typically consists of a `server`, a `database`, and an `application server`.
- A server is a computer that provides **services** to other computers over a network.
- An application server is a software framework that provides an environment for **running web applications**.
- A database is a collection of data that is **organized and stored** in a defined format.

Connection with Data Layer

- The backend layer is responsible for managing the data layer and providing the logic and functionality to support the front-end of an application.
- The connection between the backend and data layers is typically managed through an application programming interface (API).
- An API is a set of **rules** and protocols that allows different software applications to **communicate** with each other.
- The API provides a way for the front-end of an application to interact with the backend and access the data stored in the database.
- ORM frameworks such as SQLAlchemy are often used to manage the **connection** between the backend and data layers.

Connection with Data Layer

- The backend layer is responsible for managing the data layer and providing the logic and functionality to support the front-end of an application.
- The connection between the backend and data layers is typically managed through an application programming interface (API).
- An API is a set of **rules** and protocols that allows different software applications to **communicate** with each other.
- The API provides a way for the front-end of an application to interact with the backend and access the data stored in the database.
- ORM frameworks such as SQLAlchemy are often used to manage the **connection** between the backend and data layers.

Connection with Data Layer

- The backend layer is responsible for managing the data layer and providing the logic and functionality to support the front-end of an application.
- The connection between the backend and data layers is typically managed through an application programming interface (API).
- An API is a set of **rules** and protocols that allows different software applications to **communicate** with each other.
- The API provides a way for the front-end of an application to interact with the backend and access the data stored in the database.
- ORM frameworks such as SQLAlchemy are often used to manage the **connection** between the backend and data layers.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Connection with Data Layer

- The backend layer is responsible for managing the data layer and providing the logic and functionality to support the front-end of an application.
- The connection between the backend and data layers is typically managed through an application programming interface (API).
- An API is a set of **rules** and protocols that allows different software applications to **communicate** with each other.
- The API provides a way for the front-end of an application to interact with the backend and access the data stored in the database.
- ORM frameworks such as SQLAlchemy are often used to manage the **connection** between the backend and data layers.

Connection with Data Layer

- The backend layer is responsible for managing the data layer and providing the logic and functionality to support the front-end of an application.
- The connection between the backend and data layers is typically managed through an application programming interface (API).
- An API is a set of **rules** and protocols that allows different software applications to **communicate** with each other.
- The API provides a way for the front-end of an application to interact with the backend and access the data stored in the database.
- ORM frameworks such as SQLAlchemy are often used to manage the **connection** between the backend and data layers.

Domain-Driven Design

- Domain-Driven Design (DDD) is an approach to software development that focuses on the **core domain** and domain logic of an application.
- The core domain is the main focus of the application and represents the **key concepts** and entities that the application is designed to manage.
- DDD domain layer is divided into **domain objects**, which represent the core concepts and entities of the application.
- DDD application layer is divided into **services**, which are responsible for coordinating the domain objects and implementing the application logic.
- DDD infrastructure layer is responsible for managing the **connection** between the application and the external systems, such as the database or data repositories.

Domain-Driven Design

- Domain-Driven Design (DDD) is an approach to software development that focuses on the **core domain** and domain logic of an application.
- The core domain is the main focus of the application and represents the **key concepts** and entities that the application is designed to manage.
- DDD domain layer is divided into **domain objects**, which represent the core concepts and entities of the application.
- DDD application layer is divided into **services**, which are responsible for coordinating the domain objects and implementing the application logic.
- DDD infrastructure layer is responsible for managing the **connection** between the application and the external systems, such as the database or data repositories.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Domain-Driven Design

- Domain-Driven Design (DDD) is an approach to software development that focuses on the **core domain** and domain logic of an application.
- The core domain is the main focus of the application and represents the **key concepts** and entities that the application is designed to manage.
- DDD domain layer is divided into **domain objects**, which represent the core concepts and entities of the application.
- DDD application layer is divided into **services**, which are responsible for coordinating the domain objects and implementing the application logic.
- DDD infrastructure layer is responsible for managing the **connection** between the application and the external systems, such as the database or data repositories.

Domain-Driven Design

- Domain-Driven Design (DDD) is an approach to software development that focuses on the **core domain** and domain logic of an application.
- The core domain is the main focus of the application and represents the **key concepts** and entities that the application is designed to manage.
- DDD domain layer is divided into **domain objects**, which represent the core concepts and entities of the application.
- DDD application layer is divided into **services**, which are responsible for coordinating the domain objects and implementing the application logic.
- DDD infrastructure layer is responsible for managing the **connection** between the application and the external systems, such as the database or data repositories.

Domain-Driven Design

- Domain-Driven Design (DDD) is an approach to software development that focuses on the **core domain** and domain logic of an application.
- The core domain is the main focus of the application and represents the **key concepts** and entities that the application is designed to manage.
- DDD domain layer is divided into **domain objects**, which represent the core concepts and entities of the application.
- DDD application layer is divided into **services**, which are responsible for coordinating the domain objects and implementing the application logic.
- DDD infrastructure layer is responsible for managing the **connection** between the application and the external systems, such as the database or data repositories.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Sockets

- A socket is an **endpoint** for communication between two machines over a network.
- A socket is a **software** structure that allows two machines to **exchange data** over a network.
- A socket is identified by an **IP address** and a **port number**.

RESTful APIs

- A Representational State Transfer (REST) is an **architectural style** that defines a set of constraints for creating web services.
- A RESTful API is an API that follows the principles of REST and uses **HTTP methods** to perform operations on resources.
- RESTful APIs use standard **HTTP headers**, such as Content-Type, Accept, and Authorization, to provide additional information about a request or response.
- RESTful APIs are typically used to build **web services** that can be accessed by other applications over the internet.

RESTful APIs

- A Representational State Transfer (REST) is an **architectural style** that defines a set of constraints for creating web services.
- A RESTful API is an API that follows the principles of REST and uses **HTTP methods** to perform operations on resources.
- RESTful APIs use standard **HTTP headers**, such as Content-Type, Accept, and Authorization, to provide additional information about a request or response.
- RESTful APIs are typically used to build **web services** that can be accessed by other applications over the internet.

RESTful APIs

- A Representational State Transfer (REST) is an **architectural style** that defines a set of constraints for creating web services.
- A RESTful API is an API that follows the principles of REST and uses **HTTP methods** to perform operations on resources.
- RESTful APIs use standard **HTTP headers**, such as Content-Type, Accept, and Authorization, to provide additional information about a request or response.
- RESTful APIs are typically used to build **web services** that can be accessed by other applications over the internet.

RESTful APIs

- A Representational State Transfer (REST) is an **architectural style** that defines a set of constraints for creating web services.
- A RESTful API is an API that follows the principles of REST and uses **HTTP methods** to perform operations on resources.
- RESTful APIs use standard **HTTP headers**, such as Content-Type, Accept, and Authorization, to provide additional information about a request or response.
- RESTful APIs are typically used to build **web services** that can be accessed by other applications over the internet.

HTTP Methods

- The Hypertext Transfer Protocol (HTTP) is a protocol that defines how data is transmitted over the **internet**.
- HTTP methods are used to perform operations on resources, such as retrieving, creating, updating, or deleting data **(CRUD)**.
- The most common HTTP methods are `GET`, `POST`, `PUT`, `PATCH`, and `DELETE`.
  - GET is used to **retrieve** data from a server.
  - POST is used to **create** new data on a server.
  - PUT is used to **update** existing data on a server.
  - PATCH is used to **partially update** existing data on a server.
  - DELETE is used to **delete** data from a server.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

HTTP Methods

- The Hypertext Transfer Protocol (HTTP) is a protocol that defines how data is transmitted over the **internet**.
- HTTP methods are used to perform operations on resources, such as retrieving, creating, updating, or deleting data **(CRUD)**.
- The most common HTTP methods are `GET`, `POST`, `PUT`, `PATCH`, and `DELETE`.
  - GET is used to **retrieve** data from a server.
  - POST is used to **create** new data on a server.
  - PUT is used to **update** existing data on a server.
  - PATCH is used to **partially update** existing data on a server.
  - DELETE is used to **delete** data from a server.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

HTTP Codes

- HTTP status codes are standard response codes given by **web servers** on the internet.
- The status codes are divided into five categories:
  - 1xx: Informational — Request received, continuing process.
  - 2xx: Success — The action was successfully received, understood, and accepted.
  - 3xx: Redirection — Further action must be taken to complete the request.
  - 4xx: Client Error — The request contains bad syntax or cannot be fulfilled.
  - 5xx: Server Error — The server failed to fulfill an apparently valid request.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

HTTP Codes

- HTTP status codes are standard response codes given by **web servers** on the internet.
- The status codes are divided into five categories:
  - 1xx: Informational — Request received, continuing process.
  - 2xx: Success — The action was successfully received, understood, and accepted.
  - 3xx: Redirection — Further action must be taken to complete the request.
  - 4xx: Client Error — The request contains bad syntax or cannot be fulfilled.
  - 5xx: Server Error — The server failed to fulfill an apparently valid request.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

HTTP Codes

- HTTP status codes are standard response codes given by **web servers** on the internet.
- The status codes are divided into five categories:
  - 1xx: Informational — Request received, continuing process.
  - 2xx: Success — The action was successfully received, understood, and accepted.
  - 3xx: Redirection — Further action must be taken to complete the request.
  - 4xx: Client Error — The request contains bad syntax or cannot be fulfilled.
  - 5xx: Server Error — The server failed to fulfill an apparently valid request.

HTTP Codes

- HTTP status codes are standard response codes given by **web servers** on the internet.
- The status codes are divided into five categories:
  - 1xx: Informational — Request received, continuing process.
  - 2xx: Success — The action was successfully received, understood, and accepted.
  - 3xx: Redirection — Further action must be taken to complete the request.
  - 4xx: Client Error — The request contains bad syntax or cannot be fulfilled.
  - 5xx: Server Error — The server failed to fulfill an apparently valid request.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

HTTP Codes

- HTTP status codes are standard response codes given by **web servers** on the internet.
- The status codes are divided into five categories:
  - 1xx: Informational — Request received, continuing process.
  - 2xx: Success — The action was successfully received, understood, and accepted.
  - 3xx: Redirection — Further action must be taken to complete the request.
  - 4xx: Client Error — The request contains bad syntax or cannot be fulfilled.
  - 5xx: Server Error — The server failed to fulfill an apparently valid request.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

HTTP Codes

- HTTP status codes are standard response codes given by **web servers** on the internet.
- The status codes are divided into five categories:
  - 1xx: Informational — Request received, continuing process.
  - 2xx: Success — The action was successfully received, understood, and accepted.
  - 3xx: Redirection — Further action must be taken to complete the request.
  - 4xx: Client Error — The request contains bad syntax or cannot be fulfilled.
  - 5xx: Server Error — The server failed to fulfill an apparently valid request.

FastAPI

- FastAPI is a modern, fast (high-performance), **web framework** for building APIs with Python 3.6+.
- FastAPI is based on standard Python **type hints**, which makes it easy to use and understand.
- FastAPI is designed to be easy to use and understand, with a focus on **performance and scalability**.
- FastAPI is built on top of `Starlette` for the web parts and `Pydantic` for the data parts.
- FastAPI could use RPC (Remote Procedure Call) to improve the `performance` of the API.

FastAPI

- FastAPI is a modern, fast (high-performance), **web framework** for building APIs with Python 3.6+.
- FastAPI is based on standard Python **type hints**, which makes it easy to use and understand.
- FastAPI is designed to be easy to use and understand, with a focus on **performance and scalability**.
- FastAPI is built on top of `Starlette` for the web parts and `Pydantic` for the data parts.
- FastAPI could use RPC (Remote Procedure Call) to improve the `performance` of the API.

FastAPI

- FastAPI is a modern, fast (high-performance), **web framework** for building APIs with Python 3.6+.
- FastAPI is based on standard Python **type hints**, which makes it easy to use and understand.
- FastAPI is designed to be easy to use and understand, with a focus on **performance and scalability**.
- FastAPI is built on top of `Starlette` for the web parts and `Pydantic` for the data parts.
- FastAPI could use RPC (Remote Procedure Call) to improve the `performance` of the API.

FastAPI

- FastAPI is a modern, fast (high-performance), **web framework** for building APIs with Python 3.6+.
- FastAPI is based on standard Python **type hints**, which makes it easy to use and understand.
- FastAPI is designed to be easy to use and understand, with a focus on **performance and scalability**.
- FastAPI is built on top of `Starlette` for the web parts and `Pydantic` for the data parts.
- FastAPI could use RPC (Remote Procedure Call) to improve the `performance` of the API.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

FastAPI

- FastAPI is a modern, fast (high-performance), **web framework** for building APIs with Python 3.6+.
- FastAPI is based on standard Python **type hints**, which makes it easy to use and understand.
- FastAPI is designed to be easy to use and understand, with a focus on **performance and scalability**.
- FastAPI is built on top of `Starlette` for the web parts and `Pydantic` for the data parts.
- FastAPI could use RPC (Remote Procedure Call) to improve the `performance` of the API.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Postman

- Postman is a collaboration **platform** for API development that allows you to design, build, and test APIs.
- Postman provides a user-friendly interface for creating and managing API **requests**.
- Postman allows you to create **collections** of API requests, which can be shared with other team members.
- Postman provides a powerful **testing environment** for running automated tests on your APIs.
- Postman provides a variety of tools for **debugging** and troubleshooting API requests.

Postman

- Postman is a collaboration **platform** for API development that allows you to design, build, and test APIs.
- Postman provides a user-friendly interface for creating and managing API **requests**.
- Postman allows you to create **collections** of API requests, which can be shared with other team members.
- Postman provides a powerful **testing environment** for running automated tests on your APIs.
- Postman provides a variety of tools for **debugging** and troubleshooting API requests.

Postman

- Postman is a collaboration **platform** for API development that allows you to design, build, and test APIs.
- Postman provides a user-friendly interface for creating and managing API **requests**.
- Postman allows you to create **collections** of API requests, which can be shared with other team members.
- Postman provides a powerful **testing environment** for running automated tests on your APIs.
- Postman provides a variety of tools for **debugging** and troubleshooting API requests.

Postman

- Postman is a collaboration **platform** for API development that allows you to design, build, and test APIs.
- Postman provides a user-friendly interface for creating and managing API **requests**.
- Postman allows you to create **collections** of API requests, which can be shared with other team members.
- Postman provides a powerful **testing environment** for running automated tests on your APIs.
- Postman provides a variety of tools for **debugging** and troubleshooting API requests.

Postman

- Postman is a collaboration **platform** for API development that allows you to design, build, and test APIs.
- Postman provides a user-friendly interface for creating and managing API **requests**.
- Postman allows you to create **collections** of API requests, which can be shared with other team members.
- Postman provides a powerful **testing environment** for running automated tests on your APIs.
- Postman provides a variety of tools for **debugging** and troubleshooting API requests.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Outline

Containers & Docker

- A Container is a standard unit of software that **packages up code** and all its dependencies so the application runs quickly and reliably from one computing environment to another.
- A Docker is a platform for developing, shipping, and running **applications in containers**.
- A Dockerfile is a text document that contains all the commands a user could call on the command line to **assemble a docker image**.
- A Docker image is a file that contains all the necessary **files and dependencies** to run a container.

Containers & Docker

- A Container is a standard unit of software that **packages up code** and all its dependencies so the application runs quickly and reliably from one computing environment to another.
- A Docker is a platform for developing, shipping, and running **applications in containers**.
- A Dockerfile is a text document that contains all the commands a user could call on the command line to **assemble a docker image**.
- A Docker image is a file that contains all the necessary **files and dependencies** to run a container.

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Containers & Docker

- A Container is a standard unit of software that **packages up code** and all its dependencies so the application runs quickly and reliably from one computing environment to another.
- A Docker is a platform for developing, shipping, and running **applications in containers**.
- A Dockerfile is a text document that contains all the commands a user could call on the command line to **assemble a docker image**.
- A Docker image is a file that contains all the necessary **files and dependencies** to run a container.

Containers & Docker

- A Container is a standard unit of software that **packages up code** and all its dependencies so the application runs quickly and reliably from one computing environment to another.
- A Docker is a platform for developing, shipping, and running **applications in containers**.
- A Dockerfile is a text document that contains all the commands a user could call on the command line to **assemble a docker image**.
- A Docker image is a file that contains all the necessary **files and dependencies** to run a container.

Docker Compose

- Docker Compose is a tool for defining and running **multi-container Docker applications**.
- With Docker Compose, you use a `YAML` file to configure your application's services.
- Then, with a single command, you create and start **all the services** from your configuration.

Docker Compose

- Docker Compose is a tool for defining and running **multi-container Docker applications**.
- With Docker Compose, you use a `YAML` file to configure your application's services.
- Then, with a single command, you create and start **all the services** from your configuration.

Docker Compose

- Docker Compose is a tool for defining and running **multi-container Docker applications**.
- With Docker Compose, you use a `YAML` file to configure your application's services.
- Then, with a single command, you create and start **all the services** from your configuration.

Outline

Thanks!

Questions?

Repo: *https://github.com/EngAndres/ud-public/tree/main/courses/software-modeling*