

Software Engineering Foundations with Python, Data and SQL

SDLC, system modelling, introductory machine learning and network security

FA-0756 · 10 days

Software Development · Data & Analytics · Cybersecurity

Introduction

Follow the development lifecycle of a small multi-tier software project, from scope and modelling through implementation and review. Build Python foundations, explore data and machine-learning examples, and work with a prepared SQL Server environment.

The ten-day programme is a broad foundation with guided exercises and selected specialist demonstrations. Networking and certificate topics explain the supporting security concepts; cryptographic mathematics, high-performance design and production deployment are introductory discussions rather than advanced engineering qualifications.

Learning outcomes

- Explain SDLC phases and the impact of pre-development, implementation and post-development decisions.
- Model use cases, software architecture, data flow, inter-process communication and system scope.
- Write basic Python programs using collections, control flow, functions, modules, exceptions, files and objects.
- Acquire and convert permitted sample data, perform introductory statistics and visualise results.
- Build and evaluate selected scikit-learn models; recognise preprocessing leakage, overfitting and class imbalance.
- Describe Linux, SSH, networking, HTTP/HTTPS, certificate trust and basic TLS operation.

- Use a prepared relational database to create objects, query and modify data, and understand permissions and deployment considerations.

Prerequisites

Basic computer, internet and file-management skills. No previous Python experience is required; comfort with introductory mathematics supports the statistics demonstrations.

A reliable connection and authorised access to prepared notebook, Linux, database and packet-analysis training environments. Use approved installation permissions and lab endpoints rather than unrestricted root or network access.

Training outline

1. Module 1 – Programming and the Python environment

Programming versus software engineering, algorithms, machine language, programming languages and translators.

- Brief History of Python
- Supported Python 3 versions and library compatibility
- Installing Python
- IDEs
- Environment Variables
- Python Documentation
- Hello world with python
- Modes of Programming
- What Is Colab Notebook
- Why Colab Notebook Is Important
- Accessing hosted Colab notebooks or a prepared local Jupyter environment
- Main Components of Colab Notebook
- Modes
- Exporting Notebook Documents

2. Module 2 – Python syntax, collections and control flow

- Identifiers
- Reserved Words
- Lines and Indentation

- Comments
- Numbers
- Python Lists
- Python Tuples
- Python Dictionaries
- Python Sets
- Copying
- Python Strings
- String formatting
- Regular Expressions
- Arithmetic Operators
- Comparison (Relational) Operators
- Assignment Operators
- Logical Operators
- Membership Operators
- Operators Precedence
- Decision Making
- The if Statement
- The if else Statement
- For Loop
- While Loop
- Break And Continue

3. Module 3 — Functions, modules, files and objects

- Defining Your Own Functions
- Parameters
- Function Documentation
- Passing Collections to a Function
- Variable Number of Arguments
- Scope
- Map
- Filter
- Lambda
- What Are Modules
- Importing Modules

- Aliasing
- Importing Set Of Element From A Module
- Namespace
- What Are Packages
- dir function
- help function
- What Is Exception
- Exception Types
- Try Except Component
- Handle exceptions deliberately; avoid swallowing broad failures
- Handling Specific Exception
- Raise
- Access Modes
- Writing Data to a File
- Reading Data From a File
- Read Functions
- PDF reading
- XML files
- What is OOP
- Why OOP Is Important
- Classes
- Attributes
- Methods
- Constructor
- Inheritance
- Polymorphism
- Class-level attributes and static methods in Python

4. Module 4 – Lifecycle, modelling and development practice

- The SDLC Overview.
- Pre-development phases of SDLC.
- Specific phases of SDLC.
- Post-development phases of the SDLC.

System modelling

- System Modeling Overview.
- Software Architecture.
- Use Cases.
- Data Structure and Flow.
- Interprocess Communication.
- System Scope and Scale.

Methodologies and practice

- Methodologies, Paradigms, and Practices Overview.
- Process Methodologies.
- Development Paradigms.
- Development Practices.

Consider API contracts within inter-process communication and data-acquisition examples; this is not a separate API-framework development track.

5. Module 5 — Data acquisition, cleaning and visualisation

- Use external libraries, file encodings, PDF/XML extraction and permitted sample web data.
- Statistical summaries and Seaborn visualisation; covariance, correlation, conditional probability and Bayes concepts.
- Clean sample web-log data, normalise numerical features, detect outliers and impute missing values.
- Feature engineering and class imbalance, including oversampling within training data only.

6. Module 6 — Introductory machine learning

- Supervised and unsupervised learning, train/test separation and polynomial regression.
- Naive Bayes spam-classifier example, K-means clustering, entropy and decision trees.
- Bias/variance trade-offs and K-fold cross-validation; evaluation estimates generalisation but does not guarantee absence of overfitting.
- Use consistent preprocessing fitted on training data, and keep evaluation data separate.
- Selected XGBoost demonstration alongside scikit-learn examples; interpret limitations of predictions.

7. Module 7 — Linux and network foundations

- Linux basics, command-line use, SSH and authorised remote access.

- TCP/IP, HTTP and HTTPS; inspect prepared or authorised traffic with Wireshark.
- Symmetric encryption, hashing and HMAC; MD5 as a legacy algorithm, not a recommended security primitive.
- Public-key encryption, signatures and RSA concepts.

8. Module 8 — Certificates and TLS

- Inspect sample website certificates: subject, issuer, validity, signatures and domain scope.
- OpenSSL examples, RSA key generation, root trust, certificate chains and chain verification.
- Guided certificate setup in a prepared web-server lab; TLS versions and handshake inspection.
- Distinguish symmetric record encryption, authentication and key agreement. TLS 1.3 does not use static RSA key transport.
- Conceptual Diffie-Hellman and modular arithmetic; elliptic-curve point addition, doubling, discrete-log ideas and ECDHE/ECDSA comparisons. These are demonstrations, not implementation of custom cryptography.

9. Module 9 — Relational database environment and objects

- RDBMS and SQL Server; SQL versus T-SQL.
- Prepared supported SQL Server container or remote instance; containers versus virtual machines and production-configuration considerations.
- Connect with SSMS, sqlcmd or Visual Studio Code with the MSSQL extension.
- Databases, tables, stored procedures, functions and access permissions.
- Import and export sample data and discuss database-security responsibilities.

10. Module 10 — SQL practice and assessment

- Select
- Where
- Like
- Order
- Insert
- Update
- Delete
- IN Operator
- Between
- Aggregate
- Group

- Alter
- Sub queries
- Stored Procedures
- Functions

Review the small project across successive iterations and assess programming, data handling and system-design decisions.