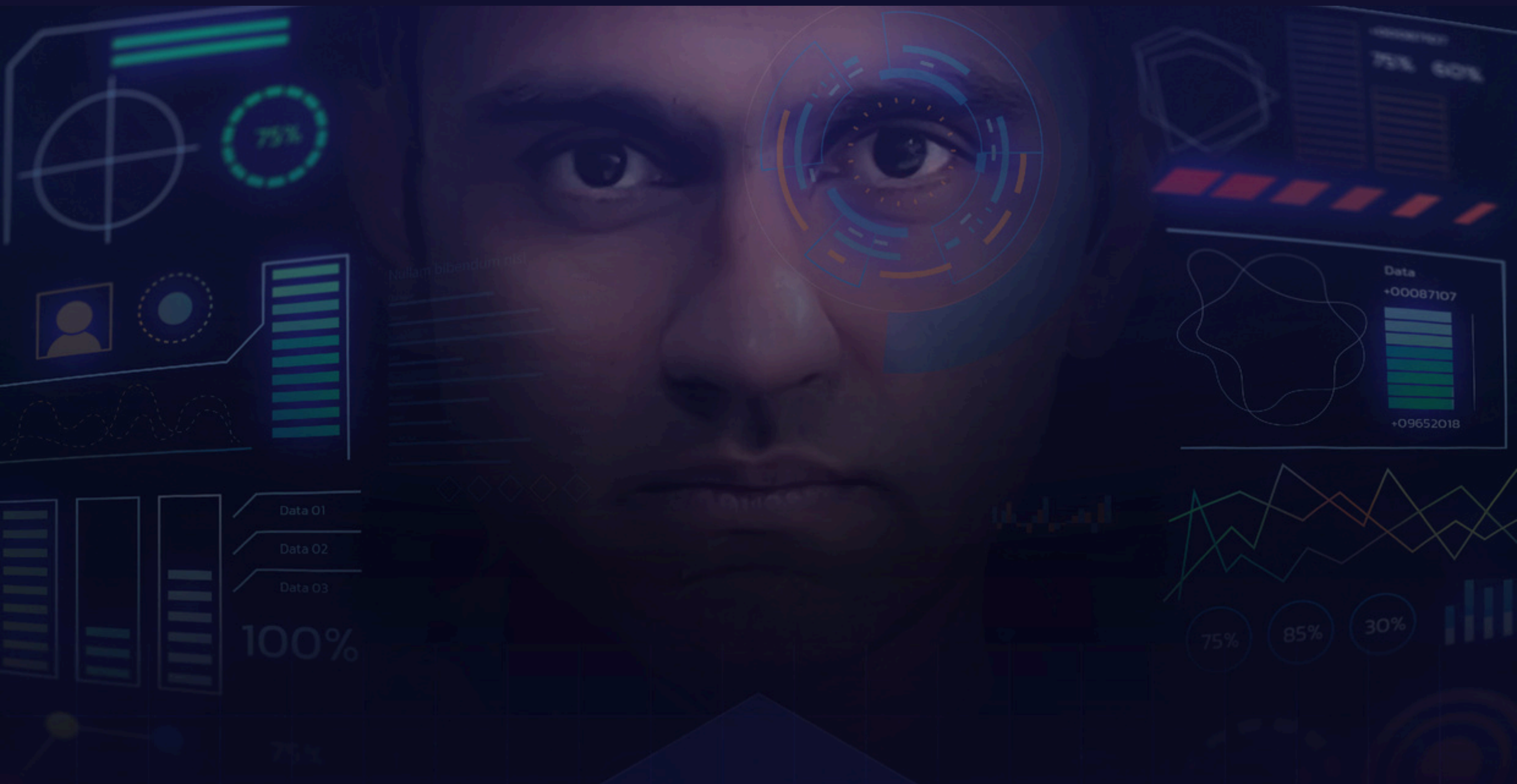

CODE BASICS

Python Advanced: Logging, PyTest, Pydantic

TAKEAWAYS

codebasics.io

Logging

1. Python's **logging** module provides a flexible framework for emitting log messages from Python programs, offering various severity levels like DEBUG, INFO, WARNING, ERROR, and CRITICAL.
2. Configuring logging can be done via basic configurations using **logging.basicConfig()** or by setting up more complex loggers, handlers, and formatters for customized logging behavior.
3. Logging provides a way to track events in a software application, which is crucial for debugging and understanding the application's operational flow without using print statements.

Logging

4. Loggers can be configured to write to different destinations, such as console, file, or even remote servers, allowing for easy monitoring and analysis of logs in different environments.
5. Proper logging practices can aid in compliance with audit trails and provide insights into application behavior, making it an essential part of maintaining and troubleshooting software in production.

Automated Testing With pytest

1 **pytest** is a powerful testing framework for Python that allows for writing simple test cases while also supporting complex functional testing for applications.

2 Tests in **pytest** are straightforward to write; any function prefixed with test_ in a module will be recognized and executed as a test case.

3 **pytest** provides features like fixtures for setup and teardown operations, making it easy to manage test state and dependencies.

Automated Testing With pytest

4 It offers advanced test discovery to automatically find and run tests across multiple files and directories, increasing the efficiency of managing large test suites.

5 With **pytest**, you can integrate with other testing services and plugins for additional functionalities like parallel testing, test coverage reports, and more, enhancing the test-driven development (TDD) process.

Working With MySQL in Python

1 Python can interact with MySQL databases using libraries like **mysql-connector-python** or **pymysql**, which provide methods for connecting to the database, executing SQL queries, and handling transactions.

2 Establishing a connection to a MySQL database requires credentials such as hostname, database name, user ID, and password, which are used to create a connection object.

3 After connecting, you can execute SQL commands like SELECT, INSERT, UPDATE, and DELETE using cursor objects, which facilitate sending commands and receiving results.

4 Closing database connections and cursors properly is essential to free up system resources and prevent data leaks, especially in applications with high database interaction.