15 FastAPI Project Ideas For Data Scientists

Discover 15 End-to-End FastAPI Project Ideas by ProjectPro designed for data scientists to build scalable and efficient data science applications.

18 min. read · View original

Working on FastAPI projects is important for data scientists, enabling them to build and deploy end-to-end data science applications quickly and efficiently. With FastAPI, data scientists can create web applications incorporating machine learning models, visualizations, and other data processing functionality.

Build Real Estate Price Prediction Model with NLP and FastAPI

Downloadable solution code | Explanatory videos | Tech Support

Start Project

Let's say you're a data scientist working for a retail company, and you've built a machine learning model that predicts customer churn based on their purchase history. Your company wants to deploy this model as a web application for their customer service team. To accomplish this, you could use FastAPI to build an API endpoint that takes in customer purchase data, runs it through your machine learning model, and returns a prediction of whether or not that customer is likely to churn. This would allow the customer service team to quickly and easily access the prediction without going through a cumbersome process of manually inputting the data and running the model.

Not only does using FastAPI make it easier to deploy your machine learning models as web applications, but it also allows you to do so in a fast, efficient, and scalable way. This can help your company make more informed decisions and improve its overall customer experience.

According to a Stack Overflow survey report, FastAPI is the third most commonly used Python web framework, used by 6.02% of developers, according to the same survey.

56.3k stars and 163k users on GitHub and 4,046,990 weekly downloads indicate the growing popularity of FastAPI!

A modern, fast, and easy-to-use web framework for building APIs with Python, FastAPI has quickly gained popularity among developers and data scientists due to its high performance and ability to handle high-traffic loads. FastAPI is designed to be simple, efficient, and developer-friendly, making it an excellent choice for building data-driven web applications. FastAPI has become a go-to choice for building APIs in the data science industry with its support for asynchronous programming and automatic API documentation. This blog will walk you through the essential steps to structure a FastAPI project and explore 15 FastAPI project ideas to help you learn how to build APIs using this robust framework.

How To Structure A Fast API Project?

Let us consider a Twitter FastAPI project example that involves retrieving and preprocessing tweets from the Twitter API using the Twitter API client.

Here is how you can structure such a FastAPI project by following a few simple steps-

1. **Define The Project Requirements:** The first step is defining the project requirements, such as the API endpoints, data sources, and user authentication.

For our Twitter FastAPI project example, the requirements could be to retrieve tweets based on user input, analyze the sentiment of the tweets using a machine learning model, and return the results to the user.

2. **Create A Project Structure:** The next step is to create a project structure, including the main FastAPI file, the machine learning model, and any other dependencies.

In the case of our example Twitter FastAPI project, the project structure would include a main FastAPI file to define the API endpoints, a separate file for the machine learning model to analyze the sentiment of the tweets, and any necessary dependencies, such as a Twitter API wrapper.

3. **Define The API Endpoints:** Define the API endpoints, including the input parameters, output structure, and authentication requirements.

For our Twitter FastAPI project, the API endpoints would include one to receive user input, such as the Twitter handle or keyword to search for, and another to return the sentiment analysis results.

4. **Implement The Business Logic:** The next step is implementing the business logic for each endpoint.

For our project example, the business logic for the first endpoint would include retrieving the tweets from the Twitter API, preprocessing the text data, and passing it to the sentiment analysis model. The business logic for the second endpoint would include returning the sentiment analysis

results to the user.

5. **Test The API:** Once the API endpoints and business logic are implemented, test the API using automated testing tools such as pytest.

6. **Deploy The API:** Finally, deploy the API using a platform such as Heroku or AWS to make it accessible to users.

Before moving to the FastAPI project ideas, let us quickly get an overview of why you must work on FastAPI projects.

Why Practice FastAPI Projects?

Practicing FastAPI projects can benefit any aspiring data scientist or software developer. Here are a few reasons you should practice working on FastAPI projects-

- FastAPI is a modern and efficient framework offering a wide range of tools and functionalities, making it easier to build high-performance web services and APIs. By working on FastAPI projects, one can learn the best practices for building scalable and maintainable web services, which are essential skills for a career in data science.

- FastAPI is gaining popularity in the industry, and many companies are looking for professionals with experience in this framework. By practicing FastAPI projects, one can gain hands-on experience with this framework, making them stand out in the job market.

- Working on FastAPI projects can help individuals develop their coding skills, such as Python programming and database management. This can help them become more confident and proficient in coding, boosting their career growth.

Now that you know the benefits of working on FastAPI projects, let us discuss 15 innovative FastAPI project ideas every data science enthusiast must explore.

15 Awesome FastAPI Projects For Data Scientists

From building a movie recommendation API to a book library API and even a voice assistant API, this section will cover various FastAPI project ideas that showcase the power and versatility of FastAPI.

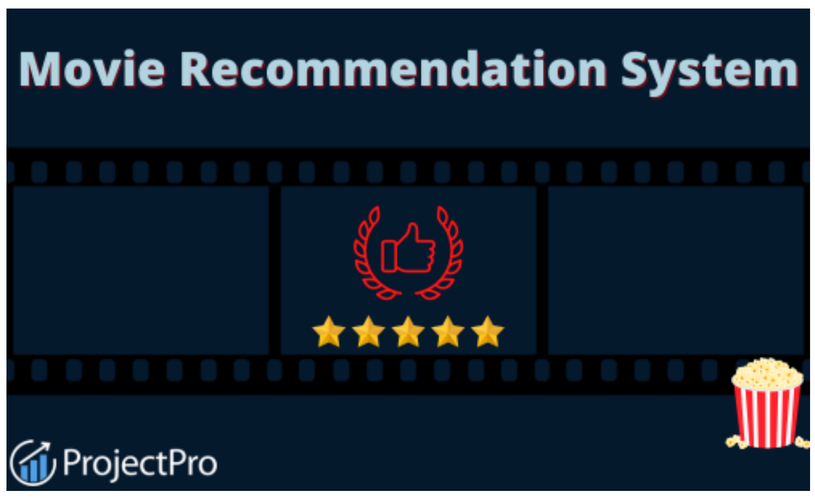
FastAPI Projects For Beginners

Below are four beginner-friendly FastAPI project ideas for those just starting with this powerful framework-

1. **Build A To-Do List API For Prioritizing Your Tasks**

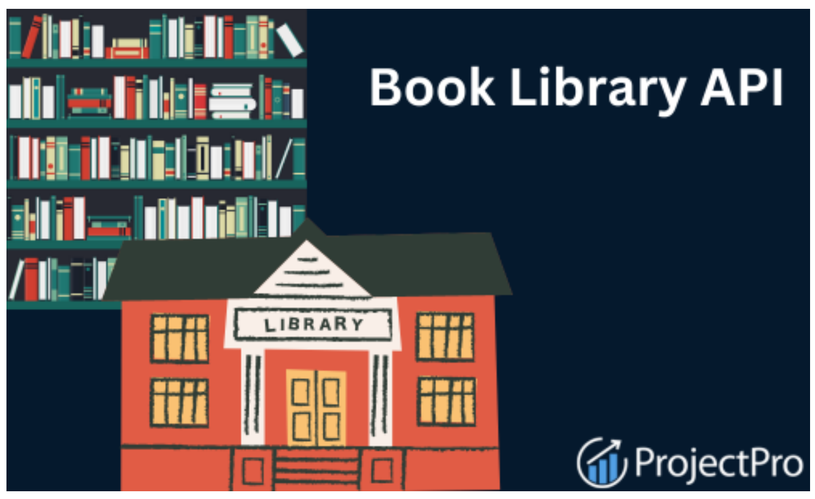
- **Tools and Technologies:** Python, FastAPI, SQLAlchemy, Docker, SQLite
- **Project Solution Approach:** The first step in this project is to set up a new FastAPI project using your preferred Python environment. Next, you will create a Task model using SQLAlchemy and define the columns for the task ID, task name, task description, and completion status. Create a new SQLAlchemy session and connect to the SQLite database. You will then implement authentication and authorization mechanisms such as OAuth2 or JWT to secure the API. Test the API using tools such as Postman or FastAPI TestClient. Containerize the application using Docker and deploy it to a cloud platform such as AWS.

2. **Build A Movie Recommendation API**



- **Tools and Technologies:** Python, FastAPI, Machine Learning (Collaborative/Content-based Filtering), Tensorflow
- **Project Solution Approach:** To build the Movie Recommendation API project, you would need a dataset containing information about movies, such as the MovieLens dataset, IMDb dataset, or TMDB dataset. Next, you must preprocess the dataset to extract relevant features such as genre, director, actors, and ratings. You can use tools like Pandas and NumPy for data cleaning and manipulation. Then, you will train a machine learning algorithm, such as collaborative or content-based filtering, using Python-based machine learning libraries like scikit-learn or TensorFlow to generate recommendations based on user preferences. After training the model, you will use FastAPI to create the API endpoints for user input and output. Users can input their preferred genres, actors, and directors, and the API will return a list of recommended movies based on the machine learning model's predictions. To handle user input, you will use FastAPI's request body feature to receive the user's input as a JSON object. You will also have to define the response model using Pydantic to ensure that the API returns a JSON object with the correct structure.

3. **Build A Book Library API For Cataloging Your Books**



- **Tools and Technologies:** Python, FastAPI, MongoDB, Docker, SQLAlchemy, SQLite
- **Project Solution Approach:** Start by defining the API endpoints for your Book Library API. For example, endpoints for retrieving, adding, updating, and deleting books. Next, set up a database to store your book data. MongoDB can be a good choice for this project since it provides a flexible schema-less data model. Then, you will use Fast API to create the endpoints for your Book Library API. To fetch additional details about books, you will integrate your Book Library API with an external API like the Google Books API or the Open Library API. The next step is implementing authentication and authorization to ensure only authorized users can access the API and perform CRUD operations. Finally, containerize your application using Docker and deploy it to a cloud provider like AWS or Heroku.

4. **Build A News API For Trending News Headlines**

- **Tools and Technologies:** FastAPI, News API, MongoDB, Docker

- **Project Solution Approach:** With this project, you can create a useful tool for yourself and others who want to stay up-to-date with the latest news. This project involves building an API to retrieve and present news articles from various sources. To build this project, you will use FastAPI, a modern, fast web framework for building APIs. The News API provides access to a large database of news articles from various sources. You can use MongoDB as a database to store the retrieved news articles. Docker can be used for containerization and deployment. You will create a new FastAPI project and install the necessary packages to get started with this project. You can then use the News API to retrieve news articles and store them in MongoDB. Finally, you will build the API endpoints using FastAPI to retrieve and present the news articles to the user.

Intermediate-Level FastAPI Projects

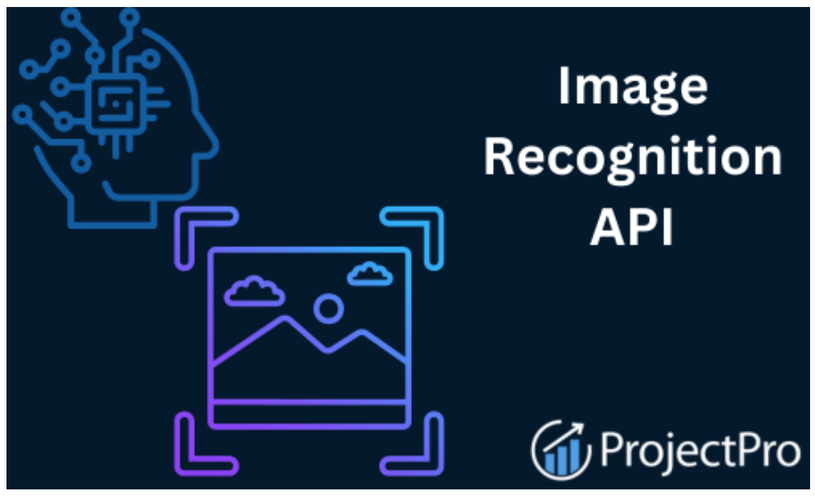
Below are four intermediate-level FastAPI project ideas for those familiar with this framework and looking to gain a deeper understanding of how to run a FastAPI app-

5. **Build A Fraud Detection API For Real-Time Fraud Detection**

- **Tools and Technologies:** FastAPI, Python, Keras, Machine Learning Algorithms

- **Project Solution Approach:** For this fraud detection project, you will collect transaction data, such as the transaction amount, timestamp, and location. You can also collect data on the user, such as their account history and demographics. You will use Python libraries such as Pandas and Scikit-learn to perform data cleaning, feature selection, and normalization tasks. You can use Scikit-learn or Keras libraries to build the model. You will use a machine learning algorithm like Logistic Regression or Random Forest to train your fraud detection model. Once you have trained the model, you can integrate it into the FastAPI application using the model's library. You can define API endpoints that take in transaction data as input and output the fraud prediction. You will test the API using tools such as Swagger UI or Postman. Once the API works correctly, you can deploy it using cloud services such as AWS or Heroku.

6. **Build An Image Recognition API**



- **Tools and Technologies:** Python, FastAPI, Tensorflow, Google Cloud Vision

- **Project Solution Approach:** There are several image recognition APIs available, such as Google Cloud Vision, AWS Rekognition, and IBM Watson Visual Recognition. You can choose the API that suits your requirements and sign up for an API key. Next, you will set up a FastAPI application using a command-line interface or a Python code editor. Using the API key, you will send an image file to the image recognition API using HTTP requests. The API will analyze the image and return a response containing object detection, facial recognition, and text recognition information. Once you have retrieved it, you can process it using Python libraries such as Pillow, NumPy, and TensorFlow. You can perform image transformations, feature extraction, and classification. Then, you will define API endpoints using FastAPI's decorator syntax, specifying the request method and the response model. For example, you can define an endpoint to detect objects in an image. Then, you will test the API using tools such as Swagger UI or Postman. Once the API works correctly, you can deploy it using cloud services such as Heroku or AWS.

7. **Build A Music Recommendation API**

- **Tools And Technologies:** Python, FastAPI, Machine Learning, PyTorch, Tensorflow

- **Project Solution Approach:** Choose a music dataset such as the Million Song Dataset, Last.fm, or Spotify's API for this project idea. These datasets contain information about songs, artists, genres, and user preferences. You will first preprocess the dataset using Python libraries such as Pandas and Numpy. You will clean the data, handle missing values, and transform the data into a format suitable for machine learning

algorithms. Next, you will choose a machine learning algorithm such as collaborative filtering, content-based filtering, or hybrid filtering. Collaborative filtering algorithms analyze user behavior and recommend music based on similarities in user preferences. Content-based filtering algorithms analyze music features such as genre, tempo, and mood and recommend music based on similarities in music features. Hybrid filtering algorithms combine collaborative and content-based filtering to provide more accurate recommendations. Then, you will train the ML algorithm using the preprocessed dataset. You can use libraries such as Scikit-learn, Tensorflow, or PyTorch for training the ML algorithm. Next, you will define API endpoints using FastAPI's decorator syntax, specifying the request method and the response model. For example, you can define an endpoint to retrieve music recommendations for a given user.

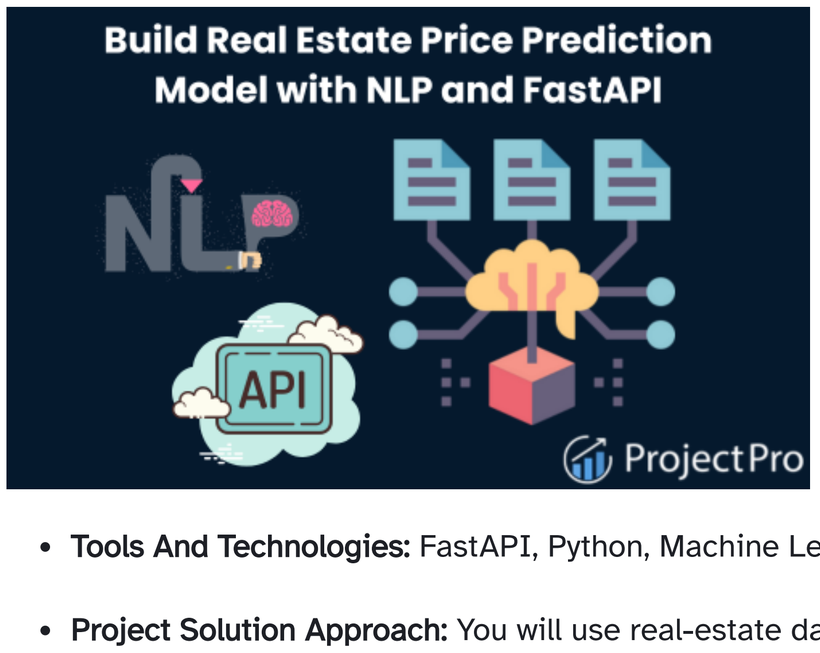
8. **Build A Stock Market API**

- **Tools and Technologies:** Python, FastAPI, Machine Learning (ARIMA, LSTM, Prophet)
- **Project Solution Approach:** The first step is to choose any of the several stock market data providers available, such as Alpha Vantage, Yahoo Finance, and Quandl, and sign up for an API key. You will then create a new FastAPI application using a command-line interface or a Python code editor. Next, using the API key, you will retrieve financial data from the financial data API using HTTP requests. The data can be in JSON or CSV format. Once you have retrieved the financial data, you will process it using Python libraries such as NumPy, Pandas, and Plotly, and perform computations, analyze trends, and create visualizations. You will implement time-series forecasting algorithms such as ARIMA, LSTM, or Prophet to predict future stock market trends using Python libraries such as statsmodels, Keras, or Prophet. You will define API endpoints using FastAPI's decorator syntax, specifying the request method and the response model. For example, you can define an endpoint to retrieve stock market data for a given stock symbol. The final step is to test your API and deploy it using any popular cloud service like AWS.

Advanced-Level FastAPI Projects

Below are four advanced-level FastAPI project ideas for those looking to become an expert at using the FastAPI framework-

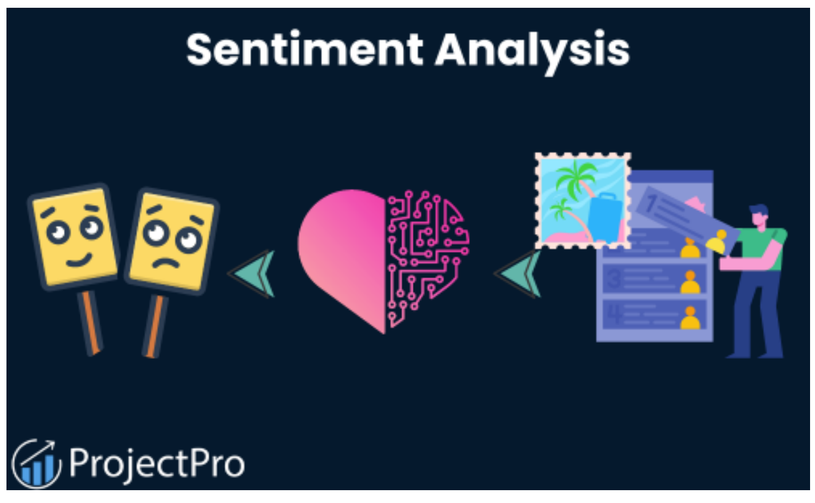
9. **Build Real Estate Price Prediction Model with NLP and FastAPI**



- **Tools And Technologies:** FastAPI, Python, Machine Learning, NLTK
- **Project Solution Approach:** You will use real-estate data, including features such as area, amenities, description, apartment type, etc. You will clean and preprocess the data using Python libraries such as Pandas, NumPy, and Scikit-learn. Use NLP techniques such as text mining and sentiment analysis and Python libraries such as NLTK to extract features such as descriptions, reviews, and comments from real estate listings. Next, you will train the machine learning model using linear, lasso, and ridge regression algorithms. Use the extracted features as inputs to predict the final selling price. Create a FastAPI application that integrates the trained model and provides an API endpoint for users to input real-estate data and get a predicted price. Deploy the FastAPI application on a cloud server such as AWS or Heroku.

Source Code: Build Real Estate Price Prediction Model with NLP and FastAPI

10. **Build A Sentiment Analysis API**



- **Tools And Technologies:** FastAPI, Python (NLTK, SpaCy), Machine Learning (Naive Bayes, SVM, etc.)
- **Project Solution Approach:** Start working on this sentiment analysis project by choosing a suitable dataset for sentiment analysis, such as the IMDB Movie Reviews dataset or the Amazon Product Reviews dataset. Next, you will preprocess the data using Python libraries such as NLTK or spaCy and perform tokenization, stemming, and lemmatization tasks. You will then train a machine learning model using Python libraries such as scikit-learn or Keras and popular algorithms such as Naive Bayes, Support Vector Machines, and Recurrent Neural Networks. Once the model

is trained, you will use a test dataset or cross-validation to test your model. You will then evaluate the model's accuracy, precision, recall, and F1 score. Using FastAPI, define API endpoints for the sentiment analysis model. The API can accept text input and return a sentiment label (positive or negative) with a confidence score.

These data science projects with R will give you the best idea of importance of R programming language in data science. Explore them today!

11. **Build A Facial Recognition API**

- **Tools And Technologies:** Python, FastAPI, Machine Learning, PyTorch/TensorFlow
- **Project Solution Approach:** For this facial recognition project, choose any of the several publicly available facial recognition datasets, such as the Labeled Faces in the Wild (LFW) dataset, the CelebA dataset, and the FaceNet dataset. Next, you will create a new FastAPI application using a command-line interface or a Python code editor. You can preprocess the images in the chosen dataset using Python libraries such as OpenCV or Pillow. You can perform operations such as resizing, cropping, and normalization. Then, you will use a machine learning algorithm such as Convolutional Neural Networks (CNN) and popular deep learning frameworks like TensorFlow or PyTorch to train your model on the preprocessed dataset. You will then define API endpoints using FastAPI's decorator syntax, specifying the request method and the response model. For example, you can define an endpoint to recognize a face in an image and return the individual's name.

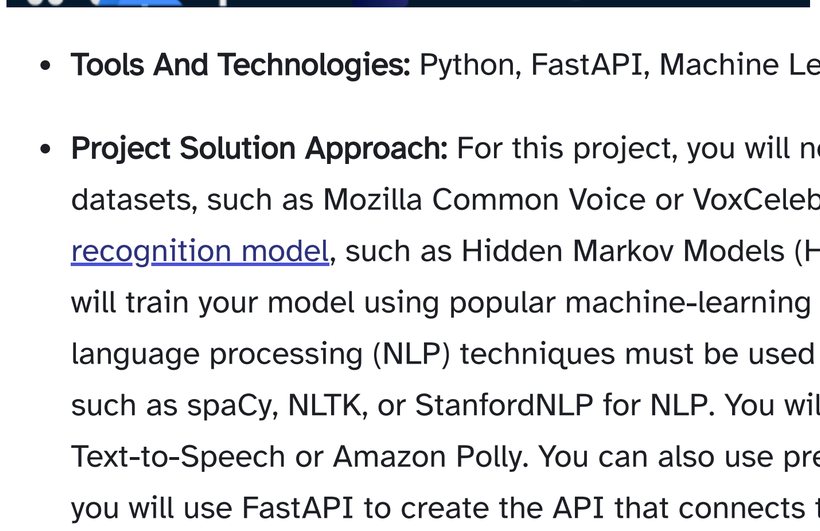
12. **Build A Voice Assistant API**

- **Tools And Technologies:** Python, FastAPI, Machine Learning, NLP, Google Text-to-Speech/Amazon Polly
- **Project Solution Approach:** For this project, you will need a dataset of audio files with corresponding transcriptions. Use any open-source datasets, such as Mozilla Common Voice or VoxCeleb. You will then use any of the several machine learning algorithms to train a speech recognition model, such as Hidden Markov Models (HMMs), Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), etc. You will train your model using popular machine-learning libraries such as TensorFlow, PyTorch, or Keras. Once the speech is recognized, natural language processing (NLP) techniques must be used to interpret the user's intent and generate appropriate responses. You can use libraries such as spaCy, NLTK, or StanfordNLP for NLP. You will then create a voice interface using text-to-speech (TTS) synthesis tools such as Google Text-to-Speech or Amazon Polly. You can also use pre-trained TTS models from open-source libraries such as Mozilla TTS or DeepSpeech. Next, you will use FastAPI to create the API that connects the voice interface with the speech recognition and NLP modules. You can define API endpoints for receiving voice commands, interpreting the intent, generating responses, and synthesizing the response in the form of speech.

FastAPI Projects GitHub

Below are three FastAPI project ideas from Github for those looking to try their hands on some unique FastAPI projects-

13. **Build An Asynchronous FastAPI To Perform CRUD On Notes**



In this FastAPI example project from Github, you will build a simple CRUD API using FastAPI. Start by creating a new Fast-Api project and run the project locally. Then, you must connect to a Postgres database to perform CRUD operations. Next, you will dockerize the project, commit the code and push it to GitHub. Then, use GitHub Actions as your CI/CD pipeline to test and build the Docker image and container. Finally, you will interact with the API via the browser or third-party tools like Postman, Insomnia, etc.

Source Code: Build An Asynchronous FastAPI To Perform CRUD on Notes

14. **Build A Single Page App With FastAPI And Vue.Js**

This FastAPI project aims to design a RESTful backend API powered by Python and FastAPI for two resources -- users and notes. The API should follow RESTful design principles, using the basic HTTP verbs: GET, POST, PUT, and DELETE. Start by creating a new " fastapi-vue " project folder and then scaffold a Vue project using the Vue CLI. Next, create and render Vue components in the browser and create a Single Page Application (SPA) with Vue components. Then, connect a Vue application to a FastAPI backend and style Vue Components with Bootstrap. Use the Vue Router to create routes, render components, and manage user auth with token-based authentication.

Source Code: Build A Basic CRUD App With FastAPI And Vue.Js

15. **Build A Product Recommendation App Using FastAPI**

This project entails building a basic application with multiple functionalities built with FastAPI to help users buy new items provided by PaypalAPI to complete the payment and check it. To run the main app, you need to use uvicorn, a lightning-fast ASGI server implementation, using uvloop and httptools. You can Switch Between using SWAGGER UI or Redoc to play around with the API. Choose an SQLite Database using SQLAlchemy for this project. You will create and use the Dockerfile to create an image of the FastAPI app and start the FastAPI app container.

Source Code: Build A Product Recommendation App Using FastAPI

Best Practices For Building FastAPI Projects

Here are some best practices for building efficient, maintainable, and scalable FastAPI projects-

1. **Use Python 3.7 or Higher:** FastAPI requires Python 3.7 or higher, so use the latest version of Python to take advantage of the latest features and optimizations. For example, if you use Python 3.7 or higher, you can use the data classes feature, which simplifies the creation of classes to represent data structures in your application.

2. **Use Asynchronous Code:** FastAPI is designed to take advantage of asynchronous programming, which allows for more efficient handling of requests and better performance. Use async/await syntax when defining endpoints and use asynchronous libraries whenever possible. For example, using the aiohttp library for asynchronous HTTP requests can help improve performance.

3. **Follow The Single Responsibility Principle:** Each function, class, or module in your project should have a single responsibility. This makes your code more modular, easier to maintain, and easier to test. For example, a function that handles authentication should only be responsible for that task and not also handle database queries or send emails.

4. **Use Type Hints And Pydantic Models:** FastAPI relies heavily on type hints and pydantic models to provide automatic data validation, serialization, and documentation. Use these features to ensure your API is well-documented and that data is properly validated. For example, defining a pydantic model for the request body of an endpoint can ensure that the correct data types are used and that required fields are not missing.

5. **Use Dependency Injection:** FastAPI supports dependency injection, allowing you to easily manage dependencies and ensure your code is testable and maintainable. For example, defining a dependency for a database connection allows you to easily switch to a different database implementation without changing the code in the endpoint that uses the database.

6. **Use A Consistent Project Structure:** Use a consistent project structure to make your code more organized and easier to navigate. Follow the recommended project structure provided by FastAPI or use a popular project structure such as cookiecutter. For example, organizing your code by domain or feature can make finding and understanding the code easier.

7. **Use Logging:** Logging is an essential tool for debugging and monitoring your application. Use the built-in Python or a third-party logging library such as loguru to log essential events and errors. For example, logging the incoming request and the response can help debug issues related to the input data or the API response.

8. **Use A Database Abstraction Layer:** If your API requires data persistence, use a database abstraction layer such as SQLAlchemy or Tortoise-ORM to make it easier to interact with the database and write maintainable code. For example, using an ORM like SQLAlchemy can abstract away the SQL syntax and allow you to write Python code that interacts with the database.

9. **Use Automated Testing:** Automated testing is essential for ensuring that your API is reliable and that changes don't introduce new bugs. Use pytest or another testing framework to write automated tests for your API. For example, writing tests for each endpoint can ensure that the API responses are correct and that changes to the code don't break existing functionality.

Elevate Your [Data Science Portfolio](#) With [End-to-End Solved Projects By ProjectPro](#)

Build Cutting-Edge FastAPI Projects With ProjectPro

FastAPI's high performance, easy-to-use API design, and support for asynchronous programming make it ideal for building scalable and robust APIs for machine learning models and other data-related projects. By working on the 15 FastAPI project ideas we have explored in this blog, you can gain hands-on experience with this framework and take your [data science skills](#) to another level. And if you want more opportunities to enhance your Python skills and work on real-world projects, check out the [ProjectPro](#) repository. With a wide range of Python projects and expert guidance from industry professionals, ProjectPro can help you build the skills and experience you need to succeed in the data science industry. So, start exploring FastAPI and embark on your journey to becoming a data science pro today!

FAQs on FastAPI Project Ideas

1. **How do I start a FastAPI project?**

You need to install Python on your system to start a FastAPI project. After that, you can install the FastAPI framework using a package manager like pip. After installing FastAPI, you can create your API by specifying endpoints, models, and database connections in a new project. You can simplify the process using tools like Pydantic and SQLAlchemy. Finally, you can test your API using tools like pytest, Swagger UI or Postman and deploy it to a server using platforms like Heroku or AWS.

2. **What tools and technologies are commonly used in FastAPI projects?**

Some of the tools and technologies commonly used in FastAPI projects include Python, Pydantic for data validation and serialization, SQLAlchemy for database management, Docker for containerization, databases like PostgreSQL and MySQL, JWT (JSON Web Tokens) for authentication and authorization, and OpenAPI (formerly known as Swagger) for API documentation. FastAPI also supports async programming, making it compatible with popular async libraries such as asyncio and aiohttp.

3. **What is FastAPI used for?**

FastAPI is used to build APIs (Application Programming Interfaces) using Python programming. FastAPI is a great option for developing asynchronous APIs that can process several requests without blocking due to its support for asynchronous programming. This makes it possible to manage massive volumes of data, create scalable web services, and build machine learning models.

4. **How can I deploy a FastAPI project?**

You can deploy a FastAPI project using any cloud provider or hosting service, such as AWS, Google Cloud, Microsoft Azure, etc., that supports Python and provides a WSGI server such as Gunicorn or Uvicorn. You can also use containers such as Docker for packaging your application and dependencies. Once you have deployed your project, you can use tools like NGINX or Apache to handle incoming requests and route them to your application.

About the Author

Daivi

Daivi is a highly skilled Technical Content Analyst with over a year of experience at ProjectPro. She is passionate about exploring various technology domains and enjoys staying up-to-date with industry trends and developments. Daivi is known for her excellent research skills and ability to distill

Meet The Author
›