Chapter– 3

INTRODUCTION TO METABONOMICS

Ch.Id:-ASU/GRF/EB/RAPOET/2022/Ch-03
DOI: https://doi.org/10.52458/9789391842765.2022.eb.grf.asu.ch-03

[1]ANUJ KAUSHIK
[1]Apeejay Stya University, Sohna, Gurugram

[2]Mr. JITENDER SINGH
[2]Industry Mentor

[3]Dr. GARIMA SHARMA
[3]Apeejay Stya University, Sohna, Gurugram

Some of the features of Express are as follows:

- *Rapid Development: Express was designed with the intention to make a framework that takes less time to build web applications The project implementation phase is very time taken but Express creates it rapidly.*
- *Secure: Express takes security seriously and helps developers to avoid many common security mistakes, such as SQL injection, cross-site scripting, cross-site request forgery etc. Its user authentication system provides a secure way to manage user accounts and passwords.*
- *Scalable: Express is scalable in nature and has the ability to quickly and flexibly switch from small to large-scale application projects.*
- *Fully Loaded: Express includes various helping task modules and libraries which can be used to handle common Web development tasks. Express takes care of user authentication, content administration, site maps, RSS feeds etc.*
- *Versatile: Express is versatile in nature which allows it to build applications for different-different domains. Nowadays, Companies are using Express to build various types of applications like: content management systems, social network sites or scientific computing platforms etc.*

EXPRESS REST FRAMEWORK

Some reasons you might want to use the REST framework:

- The Web browsable API is a huge usability win for your developers.
- Authentication policies include packages for OAuth1a and OAuth2.
- Serialization that supports both ORM and non-ORM data sources.
- Customizable all the way down - just use regular function-based views if you don't need the more powerful features.
- Extensive documentation and great community support.
- Used and trusted by internationally recognized companies including Mozilla, Red Hat, Heroku, and Eventbrite.

CONCLUSION

- This Web app was successfully created within the month and all the documentation and videos were also created.
- A complete and total independent UI service and back-end services were deployed
- I improved my full-stack web developer skills after working on this project, looking back I could have done lot of things in a better way.
- I got a good grasp of Express as a Backend framewok.
- I learned to work in a team and also learned to work in pressure situations.
- I also learned how to create documentation & also improved my communication skills by creating the videos in English.

Developer Bootcamp Application

Scope of the Project

It was the application build using NodeJS, Express and MongoDB. It is used to manage bootcamps, courses, reviews, users & authentication. The project was focused on the parts of Node and Express to build real life REST API's with authentication, database integration and so on.

Methodology

I worked closely with the design, testing and content development team to deliver the project on time. Agile and fast sprints were used to deliver projects on time. Regular sprints and meetings were conducted between design, developers and Quality Assurance team so that people are on same page. Its was a SCRUM environment so

every team be it mobile, design or backend team they have common scrum with CTO which varied from 20 to 40 minutes.

Modules of the application

The app is divides into 6 modules which includes Bootcamps, Courses, Reviews, Users, Authentication & Security.

Design Programming and Implementation

About Express

Some of the features of Express are as follows:

- Rapid Development: Express was designed with the intention to make a framework that takes less time to build web applications The project implementation phase is very time taken but Express creates it rapidly.
- Secure: Express takes security seriously and helps developers to avoid many common security mistakes, such as SQL injection, cross-site scripting, cross-site request forgery etc. Its user authentication system provides a secure way to manage user accounts and passwords.
- Scalable: Express is scalable in nature and has the ability to quickly and flexibly switch from small to large-scale application projects.
- Fully Loaded: Express includes various helping task modules and libraries which can be used to handle common Web development tasks. Express takes care of user authentication, content administration, site maps, RSS feeds etc.
- Versatile: Express is versatile in nature which allows it to build applications for different- different domains. Nowadays, Companies are using Express to build various types of applications like: content management systems, social network sites or scientific computing platforms etc.

Express REST framework

Some reasons you might want to use the REST framework:

- The Web browsable API is a huge usability win for your developers.
- Authentication policies include packages for OAuth1a and OAuth2.
- Serialization that supports both ORM and non-ORM data sources.
- Customizable all the way down - just use regular function-based views if you don't need the more powerful features.
- Extensive documentation, and great community support.
- Used and trusted by internationally recognized companies including Mozilla, Red Hat, Heroku, and Eventbrite.

MVC Architecture

The Model- View- Controller (MVC) is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller. Each of these components are built to handle specific development aspects of an application. MVC is one of the most frequently used industry-standard web development framework to create scalable and extensible projects.

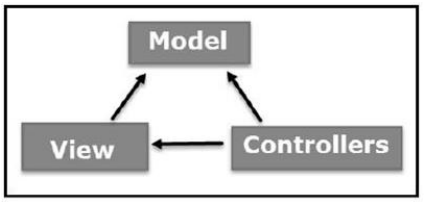


Figure 4.1 for MVC

The Model

The model is responsible for managing the data of the application. It responds to the request from the view and it also responds to instructions from the controller to update itself.

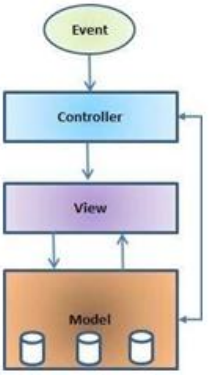


Figure 4.2 for MVC

The View

It means presentation of data in a particular format, triggered by a controller's decision to present the data. They are script-based templating systems like JSP, ASP, PHP and very easy to integrate with AJAX technology.

The Controller

The controller is responsible for responding to the user input and perform interactions on the data model objects. The controller receives the input, it validates the input and then performs the business operation that modifies the state of the data model.

Postman

Postman is an API(application programming interface) development tool which helps to build, test and modify APIs. Almost any functionality that could be needed by any developer is encapsulated in this tool. It is used by over 5 million developers every month to make their API development easy and simple. It has the ability to make various types of HTTP requests(GET, POST, PUT, PATCH), saving environments for later use, converting the API to code for various languages(like JavaScript, Python).

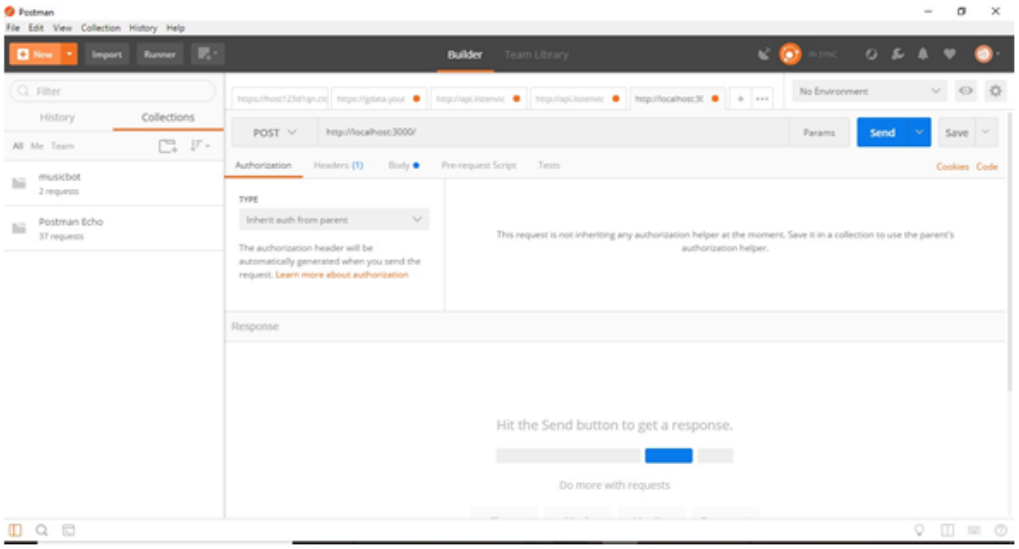


Figure 4.3 for Postman

Explaining the Interface

- The longest middle input field that looks something like a search bar is where the URL that we want to GET or POST or DELETE,etc. is fed.
- Just to the left of it, is a drop down button which has all the various HTTP methods as options. If you want to POST to the URL that you have specified, select POST.
- To the right of it is the params button. If you click on it, a new interface will appear.

  Params are basically the data that we want to send to the server with our request. We will use this params interface to POST to put app a new User.
- To the left of this button is the Send button which is used in sending the request to the server or the app in this case.

Sending and receiving requests through Postman

- Enter the url that you want to hit in the URL bar that i described above. I will put http:localhost:3000 in my case.
- Lets select our HTTP method to send the request as GET in the left button. Now click on the Send button.
- You will be returned HTML of the URL that you GET. I have selected the Preview to have a browser-like look.

- As you can see in the snap below that with the response from the server or the app, various headers are returned too with the main response.

MongoDB

MongoDB is an open-source document-oriented database that is designed to store a large scale of data and also allows you to work with that data very efficiently. It is categorized under the NoSQL (Not only SQL) database because the storage and retrieval of data in the MongoDB are not in the form of tables.

The MongoDB database is developed and managed by MongoDB.Inc under SSPL(Server Side Public License) and initially released in February 2009. It also provides official driver support for all the popular languages like C, C++, C#, and .Net, Go, Java, Node.js, Perl, PHP, Python, Motor, Ruby, Scala, Swift, Mongoid. So, that you can create an application using any of these languages. Nowadays there are so many companies that used MongoDB like Facebook, Nokia, eBay, Adobe, Google, etc. to store their large amount of data**.**

How it works ?

Now, we will see how actually thing happens behind the scene. As we know that MongoDB is a database server and the data is stored in these databases. Or in other words, MongoDB environment gives you a server that you can start and then create multiple databases on it using MongoDB. Because of its NoSQL database, the data is stored in the collections and documents. Hence the database, collection, and documents are related to each other as shown below:

- The MongoDB database contains collections just like the MYSQL database contains tables. You are allowed to create multiple databases and multiple collections.
- Now inside of the collection we have documents. These documents contain the data we want to store in the MongoDB database and a single collection can contain multiple documents and you are schema-less means it is not necessary that one document is similar to another.
- The documents are created using the fields. Fields are key-value pairs in the documents, it is just like columns in the relation database. The value of the fields can be of any BSON data types like double, string, boolean, etc.
- The data stored in the MongoDB is in the format of BSON documents. Here, BSON stands for Binary representation of JSON documents. Or in other words, in the backend, the MongoDB server converts the JSON data into a binary form that is known as BSON and this BSON is stored and queried more efficiently.
- In MongoDB documents, you are allowed to store nested data. This nesting of data allows you to create complex relations between data and store them in the

same document which makes the working and fetching of data extremely efficient as compared to SQL. In SQL, you need to write complex joins to get the data from table 1 and table 2. The maximum size of the BSON document is 16MB.

Features of MongoDB –

- **Schema-less Database:** It is the great feature provided by the MongoDB. A Schema- less database means one collection can hold different types of documents in it. Or in other words, in the MongoDB database, a single collection can hold multiple documents and these documents may consist of the different numbers of fields, content, and size. It is not necessary that the one document is similar to another document like in the relational databases. Due to this cool feature, MongoDB provides great flexibility to databases.
- **Document Oriented:** In MongoDB, all the data stored in the documents instead of tables like in RDBMS. In these documents, the data is stored in fields(key-value pair) instead of rows and columns which make the data much more flexible in comparison to RDBMS. And each document contains its unique object id.
- Indexing: In MongoDB database, every field in the documents is indexed with primary and secondary indices this makes easier and takes less time to get or search data from the pool of the data. If the data is not indexed, then database search each document with the specified query which takes lots of time and not so efficient.
- **Scalability:** MongoDB provides horizontal scalability with the help of sharding.

Sharding means to distribute data on multiple servers, here a large amount of data is partitioned into data chunks using the shard key, and these data chunks are evenly distributed across shards that reside across many physical servers. It will also add new machines to a running database.

- Replication: MongoDB provides high availability and redundancy with the help of replication, it creates multiple copies of the data and sends these copies to a different server so that if one server fails, then the data is retrieved from another server.
- Aggregation: It allows to perform operations on the grouped data and get a single result or computed result. It is similar to the SQL GROUPBY clause. It provides three different aggregations i.e, aggregation pipeline, map-reduce function, and single- purpose aggregation methods
- High Performance: The performance of MongoDB is very high and data persistence as compared to another database due to its features like scalability, indexing, replication, etc

CONCLUSION

With the help of this project I learned how to write the clean code and to follow some principles like DRY (Don't Repeat Yourself), KISS (Keep It Simple !) and many other to write clean code. So that the code can be maintainable, its reusability can be increased, its maintainability and it can be extensible. I also got to know to set up the environments in postman, published documentations and many other things related to postman. I also got a good grasp of Express as a Backend framework. Along with it I learned to work in a team and also learned to work in pressure situations.

YouTube Transcript Summarizer Objective

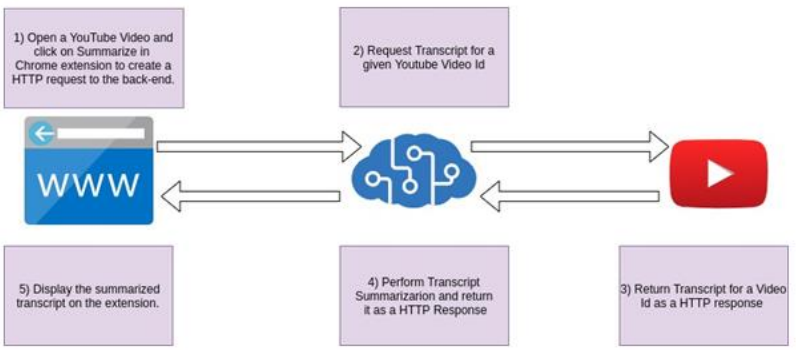
In the project, I have created a Chrome Extension which will make a request to a backend REST API where it will perform NLP and respond with a summarized version of a YouTube transcript.

Project Context

Enormous number of video recordings are being created and shared on the Internet through out the day. It has become really difficult to spend time in watching such videos which may have a longer duration than expected and sometimes our efforts may become futile if we couldn't find relevant information out of it. Summarizing transcripts of such videos automatically allows us to quickly look out for the important patterns in the video and helps us to save time and efforts to go through the whole content of the video This project will give us an opportunity to have hands on experience with state of the art NLP technique for abstractive text summarization and implement an interesting idea suitable for intermediates and a refreshing hobby project for professionals.

Project Stages

The project consists of the following stages:



High-Level Approach

- Get transcripts/subtitles for a given YouTube video Id using a Python API.
- Perform text summarization on obtained transcripts using HuggingFace transformers.
- Build a Flask backend REST API to expose the summarization service to the client.
- Develop a chrome extension which will utilize the backend API to display summarized text to the user.

Applications that can further be created using this idea

- Meetings and video-conferencing - A system that could turn voice to text and generate summaries from your team meetings.
- Patent research - A summarizer to extract the most salient claims across patents.

Weekly work breakdown

WEEK 1

Getting started with the back-end

APIs changed the way we build applications, there are countless examples of APIs in the world, and many ways to structure or set up your APIs. In this milestone, we are going to see how to create a back-end application directory and structure it to work with the required files. We are going to isolate the back-end of the application to avoid the conflicting dependencies from other parts of the project.

- Created a back-end application directory containing files named as app.py and requirements.txt.
- Initialized app.py file with basic Flask RESTful BoilerPlate with the tutorial link as mentioned in the Reference Section below.
- Created a new virtual environment with pip installed which will act as an isolated location (a directory) where everything resides.
- Activated the newly formed virtual environment and installed the following dependencies using pip:-
- Learned about Flask
- Implemented youtube_transcript_api
- Implemented and learned about transformers[torch]
- Executed pip freeze and redirect the output to the requirements.txt file. This requirements.txt file is used for specifying what python packages are required to run the project.

Get transcript for a given video

Here I have utilized a python API which allows us to get the transcripts/subtitles for a given YouTube video. It also works for automatically generated subtitles, supports translating subtitles and it does not require a headless browser, like other selenium based solutions do!

WEEK 2

Perform text summarization

Text summarization is the task of shortening long pieces of text into a concise summary that preserves key information content and overall meaning. There are two different approaches that are widely used for text summarization:

- Extractive Summarization: This is where the model identifies the important sentences and phrases from the original text and only outputs those.
- Abstractive Summarization: The model produces a completely different text that is shorter than the original, it generates new sentences in a new form, just like humans do. In this project, we will use transformers for this approach.

In this project, I have used HuggingFace's transformers library in Python to perform abstractive text summarization on the transcript obtained from previous milestone.

Created REST API endpoint

The next step is to define the resources that will be exposed by this backend service. This is an extremely simple application; we only have a single endpoint, so our only resource will be the summarized text.

In app.py,

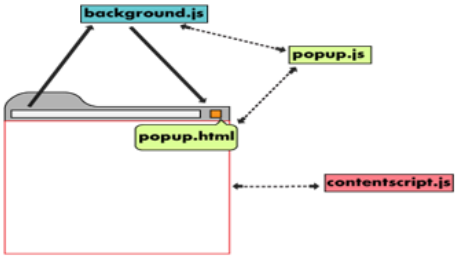
- Created a Flask API Route with GET HTTP Request method with a URI http://[hostname]/api/summarize?youtube_url=<url>.
- Extracted the YouTube video id from the YouTube URL which was obtained from the query params.
- Generated the summarized transcript by executing the transcript generation function following the execution of transcript summarizer function.
- Returned the summarized transcript with HTTP Status OK and handle HTTP exceptions if applicable.

WEEK 3

Getting Started with Chrome Extension

Extensions are small software programs that customize the browsing experience. They enable users to tailor Chrome functionality and behavior to individual preferences. They are built on web technologies such as HTML, CSS and JavaScript. In this milestone, we are going to see how to create a recommended Chrome extension application directory and structure it to work with the required files.

- Created a chrome extension application directory containing essential files required as mentioned below.
- The below diagram indicates the brief role of each of the files for building a chrome extension.



- Paste the following code snippet in the manifest.json.

{"manifest_version": 2, "name": "YSummarize",

"description": "An extension to provide summarized transcript of a YouTube Subtitle eligible Video.",

"version": "1.0", "permissions": ["activeTab"],}

Built a User Interface for Extension Popup

We need a user interface so that the user can interact with the popups which are one of several types of user interface that a Chrome extension can provide. They usually appear upon clicking the extension icon in the browser toolbar.

Display Summarized transcript

We have provided a basic UI to enable users to interact and display the summarized text but there are some missing links which must be addressed. In this milestone, we will add a functionality to allow the extension to interact with the backend server using HTTP REST API Calls.

Design Programming and Implementation

About Flask

Flask is an API of Python that allows us to build up web-applications. It was developed by Armin Ronacher. Flask's framework is more explicit than Django's framework and is also easier to learn because it has less base code to implement a simple web-Application. A Web- Application Framework or Web Framework is the collection of modules and libraries that helps the developer to write applications without writing the low-level codes such as protocols, thread management, etc. Flask is based on WSGI(Web Server Gateway Interface) toolkit and Jinja2 template engine.

Features of Flask

Flask was designed to be easy to use and extend. The idea behind Flask is to build a solid foundation for web applications of different complexity. From then on you are free to plug in any extensions you think you need. Also you are free to build your own modules. Flask is great for all kinds of projects. It's especially good for prototyping. Flask depends on two external libraries: the Jinja2 template engine and the Werkzeug WSGI toolkit.

- built-in development server and fast debugger
- integrated support for unit testing
- RESTful request dispatching
- Jinja2 templating
- support for secure cookies (client side sessions)
- WSGI 1.0 compliant
- Unicode based

Plus Flask gives you so much more CONTROL on the development stage of your project. It follows the principles of minimalism and let's you decide how you will build your application.

- Flask has a lightweight and modular design, so it easy to transform it to the web framework you need with a few extensions without weighing it down
- ORM-agnostic: you can plug in your favourite ORM e.g. SQLAlchemy.
- Basic foundation API is nicely shaped and coherent.
- Flask documentation is comprehensive, full of examples and well structured. You can even try out some sample application to really get a feel of Flask.
- It is super easy to deploy Flask in production (Flask is 100% WSGI 1.0 compliant")
- HTTP request handling functionality

- High Flexibility
- The configuration is even more flexible than that of Django, giving you plenty of solution for every production need.

About Transformers

Transformers provide APIs to easily download and train state-of-the-art pretrained models. Using pretrained models can reduce your compute costs, carbon footprint, and save you time from training a model from scratch. The models can be used across different modalities such as:

Text: text classification, information extraction, question answering, summarization, translation, and text generation in over 100 languages.

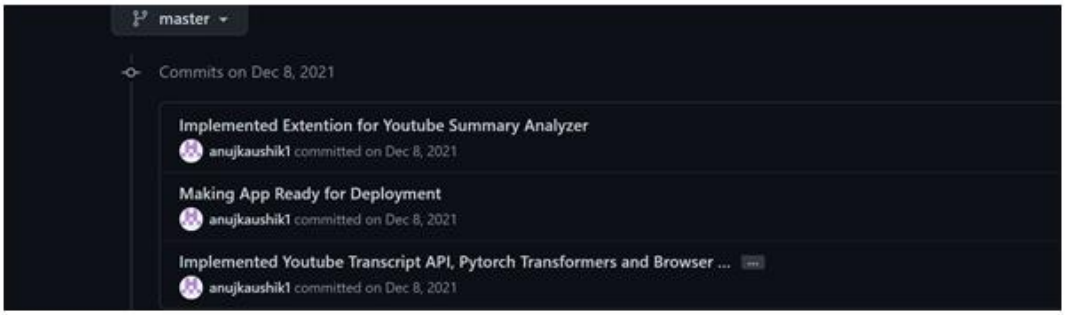
Images: image classification, object detection, and segmentation. Audio: speech recognition and audio classification. Multimodal: table question answering, optical character recognition, information extraction from scanned documents, video classification, and visual question answering.

About Pipelines

The pipeline () makes it simple to use any model from the Model Hub for inference on a variety of tasks such as text generation, image segmentation and audio classification. Even if you don't have experience with a specific modality or understand the code powering the models, you can still use them with the pipeline()! This tutorial will teach you to:

- Use a pipeline () for inference.
- Use a specific tokenizer or model.
- Use a pipeline () for audio and vision tasks.

Github Project Snapshot



CONCLUSION

With the help of the project I learned how to create the backend using the Flask framework. In this I got to know how to create own chrome extentions and integrate it with the flask application. I also got the introductions on the NLP and Tranformers as I haven't ever worked on something like this. In this I also learned about the CORS to connect the Javascript application[Chrome Extention] with the Python Framework[Flask]. I learned to work in a team and also learned to work in pressure situations. Along with this I also learned how to create documentation & also improved my communication skills by creating the videos in

Conclusion of Training

This project has turned out to be a valuable experience as a software developer. The theory of how the development should be carried out and the things that can go wrong in the development process were experienced first time. Many little challenges were encountered and the understanding of why managing, and Web development is not an easy task grew during this project. This internship offered me opportunities to learn and develop in many areas and I gained a lot of experience in many technologies which were new for me such as, NodeJS, Express, MongoDB, React, Flask, NLP and many other things

The valuable experience has transformed me as an individual & taught me how to handle the workload in industry & how to make documentations and prototypes before beginning a project, what are the things you must keep in mind before taking up a project, such as target audiences, user-friendly, attractiveness etc. Finally, this internship allowed me, autonomously, to acquire skills such as:

- Core Backend development
- Database management
- Scalability of the application
- Software Design
- Testing and debugging.
- Frontend web designing.
- React.js Framework.
- Responsive design skills.
- Search engine optimization

BIBLIOGRAPHY

- *https://nodejs.org/en/about/*
- *https://www.postman.com/*
- *https://code.visualstudio.com/*
- *https://expressjs.com/*
- *https://flask.palletsprojects.com/en/2.1.x/*
- *https://www.google.com/search?q=transformrs+hgging+face&rlz=1C1UEAD_en IN976IN976&oq=transformrs+hgging+face&aqs=chrome..69i57j0i13l2j0i22i30j0 i10i22i30l2j0i22i30j0i10i22i30j0i8i10i13i30j0i8i13i30.3291j0j4&sourceid=chrome &ie=UTF-8*
- *https://www.mongodb.com/cloud/atlas/lp/try2?utm_source=google&utm_campai gn=gs_apac_india_search_core_brand_atlas_desktop&utm_term=mongodb&ut m_medium=cpc_paid_search&utm_ad=e&utm_ad_campaign_id=12212624347 &adgroup=115749713423&gclid=Cj0KCQjwtvqVBhCVARIsAFUxcRtPF1QjwJv-Z-hkwdw1bG7Fyk-8YNLIV6U0r6CfXAvXlD7GLvgO93EaAoMrEALw_wcB*
- *https://www.python.org/psf-landing/*
- *https://www.java.com/en/*
- *https://reactjs.org/*