# **Project Overview**

## **OBJECTIVE**

1. Design complete solution to demonstrate end-to-end pipeline development of website development along with its cloud deployment
2. Develop understanding of all stages of an AI product (development & cloud deployment) project lifecycle
3. Demonstrate understanding of challenges encountered during the project development and provide ways to tackle them
4. Showcase understanding of software engineering best practices while developing the project
5. Deliverables:
   1. Content website with recommendation system, text verification, searching with sync functionality with github, dropbox, googledrive, arxiv, linkedIn, facebook, instagram, twitter etc
      1. App development for the above content website
   2. Online latex documentation with text verification and searching with sync functionality with github, dropbox, googledrive, arxiv, linkedIn, facebook, instagram, twitter etc
      1. App development for the above online documentation
   3. Online python IDE development (similar to google colab jupyter notebook + spyder) with sync functionality with github, dropbox, googledrive, arxiv, linkedIn etc
   4. Android Mobile App Development for projects 1.a, 1.b and 1.c (all in one App)

**References**:

1. https://business.adobe.com/in/products/magento/open-source.html
2. Content website sample: <https://medium.com/>
3. Online document development sample: <https://www.overleaf.com/project>
4. <https://www.kaggle.com/saurav9786/recommender-system-using-amazon-reviews>
5. <https://www.kdnuggets.com/2017/08/recommendation-system-algorithms-overview.html>

## **PROBLEM STATEMENT**

Develop the following applications with relevant security features:

* 1. Content website with recommendation system, text verification, searching with sync functionality with github, dropbox, googledrive, arxiv, linkedIn, facebook, instagram, twitter etc
     1. App development for the above content website
  2. Online latex documentation with text verification and searching with sync functionality with github, dropbox, googledrive, arxiv, linkedIn, facebook, instagram, twitter etc
  3. Online python development IDE (similar to google colab, jupyter notebook + spyder) with sync functionality with github, dropbox, googledrive, arxiv, linkedIn etc
  4. App development for the above online documentation
  5. Android Mobile App Development for projects 1.a, 1.b and 1.c (all in one App)

These applications needs to have the following features:

* Admin login
  + Add new functionalities like trained APIs
  + Modify existing functionalities like updated APIs
* Customer login
  + Submit new article
  + Remove/modify already submitted article
  + Develop article using online word tool or latex tool
  + Sync documents/articles with github, dropbox, googledrive, arxiv, linkedIn, facebook, instagram, twitter etc
  + Search through existing documents
  + Get relevant article recommendations
  + Payment gateway

## **Modules Details**

Different technical modules are elaborated as follows –

1. Project Management
2. Project Requirement Documentation
3. Data
   1. Data Preparation:
   2. Data sources: medium.com
   3. Database: MySQL / MongoDB
4. AIML Module
   1. Models
      1. Collaborative filtering
      2. Clustering
      3. Content based recommendations
      4. Ensemble methods
      5. Technologies: Python, MySQL
   2. Model Training
   3. Model Deployment
   4. Model Prediction
5. Front-End & Website
   1. Technologies: Java script, html, CSS
   2. Sample website: <https://www.dunzo.com/order>
   3. Open Source ecommerce build tools
      1. <https://business.adobe.com/in/products/magento/open-source.html>
      2. https://kinsta.com/blog/shopify-alternatives/
6. Back-end
   1. Technologies: Python, React JS, Java, MySQL and MongoDB
7. Recommendation service
   1. Fast API
8. Search Service
   1. Elastic search, Faiss similarity search tools
      1. <https://engineering.fb.com/2017/03/29/data-infrastructure/faiss-a-library-for-efficient-similarity-search/>
      2. <https://towardsdatascience.com/understanding-faiss-619bb6db2d1a>
      3. https://faiss.ai/
9. Integration and Development
   1. Technologies: Docker, Nginx, AWS EC2 small instance
10. Payment Gateway
11. Testing
    1. Write test cases
12. Documentation
    1. Project documentation - detailed
    2. Concise technical documentation (Sample: <https://arxiv.org/abs/1901.11210> )

## **MILESTONES**

There would be 14 milestones –

| WeekNo | 1.1 Content | 1.2 Latex | 1.3 APP | 2. DS APIs |
| --- | --- | --- | --- | --- |
| 1  1-7May | Project Requirement Doc | Project Requirement Doc | Project Requirement Doc | Project Requirement Doc |
| 2  8-14May | 1. SignOff Project Req Document 2. Literature Survey 3. Technology Identification 4. Effort Estimation 5. Knowledge Sharing | 1. SignOff Project Req Document 2. Literature Survey 3. Technology Identification 4. Effort Estimation 5. Knowledge Sharing | 1. SignOff Project Req Document 2. Literature Survey 3. Technology Identification 4. Effort Estimation 5. Knowledge Sharing | 1. SignOff Project Req Document 2. Literature Survey 3. Technology Identification 4. Effort Estimation 5. Knowledge Sharing |
| 3  15-21May | 1. High level design 2. Low level design 3. Open Sources Identification 4. Tech set-up ready 5. Product documentation version 0 | 1. High level design 2. Low level design 3. Open Sources Identification 4. Tech set-up ready 5. Product documentation version 0 | 1. High level design 2. Low level design 3. Open Sources Identification 4. Tech set-up ready 5. Screen designs on paper 6. Product documentation version 0 | 1. High level design 2. Low level design 3. Open Sources Identification 4. Tech set-up ready 5. Product documentation version 0 |
| 4  22-28May | 1. MVP ready | 1. MVP ready | 1. MVP ready | 1. MVP ready |
| 5 |  |  |  |  |
| 6 |  |  |  |  |
| 7 |  |  |  |  |
| 8 |  |  |  |  |
| 9 |  |  |  |  |
| 10 |  |  |  |  |
| 11 |  |  |  |  |
| 12 |  |  |  |  |
| 13 |  |  |  |  |
| 14 | UAT & Signoff | UAT & Signoff | UAT & Signoff |  |

Technology Requirement:

* Front-End: HTML, CSS, Bootstrap, Java Script, React
* Environment: Python 3
* Backend: Jupyter Notebook, Flask, FastAPI, Streamlit, Papermill, Python
* ML Libraries: Pandas, Numpy, Sklearn, Rake,Pivottable,nltk,truncatedSVD
* DataBase: MySQL, MongoDB
* AWS Instance: EC2
* Local infrastructure: Vagrant
* Virtualization tool: Docker
* Testing: SQLMap(SQL Injection), OWASP ZAP(Overall Testing)
* Source code Management: GitHub
* Security: Codacy Security Scan, Dependabot alerts (Plugins in Github)
* Documentation: Overleaf, Lucidcharts

***Integration & Deployment :-***

**Devops Literature Survey: -**

In this project, we will learn how to set up a continuous integration and continuous delivery (CI/CD) pipeline on AWS. A pipeline helps us to automate steps in your software delivery process, such as introducing automatic builds and then deploying to Amazon EC2 instances. We will use Jenkins, a service that builds, tests, and deploys the code every time there is a code change, based on the release process models defined. As part of our setup, we will plug other devops tools into jenkins to complete your software delivery pipeline. The other tools which are used are github and Maven and others based on requirement. Github acts as source code repository and Maven acts as Building tool. The detailed diagram will be created based on project requirement and this guide will show you how to create a very simple pipeline that pulls code from a source and automatically deploys it an amazon ec2 instances

<https://www.lambdatest.com/blog/cicd-pipeline-challenges/>

<https://www.altexsoft.com/blog/ci-cd-pipeline-problems/>

<https://harness.io/blog/continuous-delivery/ci-cd-challenges/>

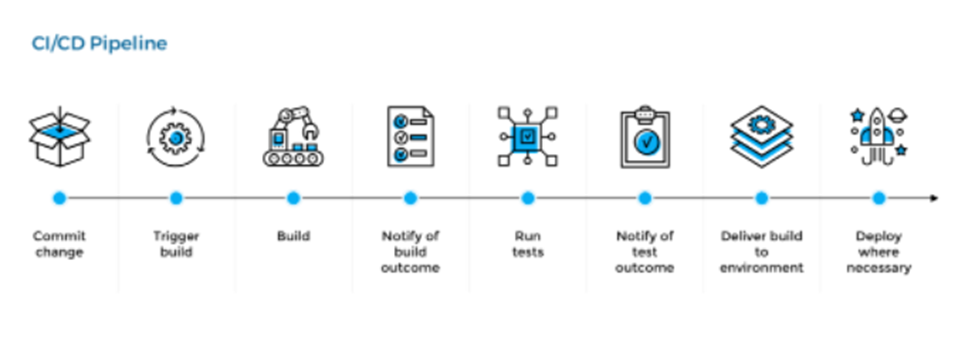
https://betterprogramming.pub/ci-cd-ingredients-for-success-2031cf7ad152

**Technology**

Top choice for integration processes in a lot of organizations is **Jenkins**. Why?

1. Cross platform (Windows, Linux, Unix-like) - Can be installed on almost all operating systems.
2. Open source - Huge community contribution for extending functionality using plugins.
3. Simple interface.
4. Supports master/slave architecture - A team can have distributed builds to manage load and provide the desired environment based on requests sent by the master.
5. Has a CLI (command line interface).
6. The software itself does not have charges although since it will be deployed on a machine, we need to bear the charges of infrastructure.

Devops- Docker, ngnix, AWS(EC2), Kubernetes (EKS)



The goal of DevOps is to

· Accelerate Flow - faster delivery

· Amplify Feedback - high quality

· Continuous improvement

**Devops Tools:-**

*Integration tool*:Jenkins, Gitlab or Gitbhub, Docker Hub

*Deployment*:Nginx,terraform,ansible,docker,kubernetes(eks)

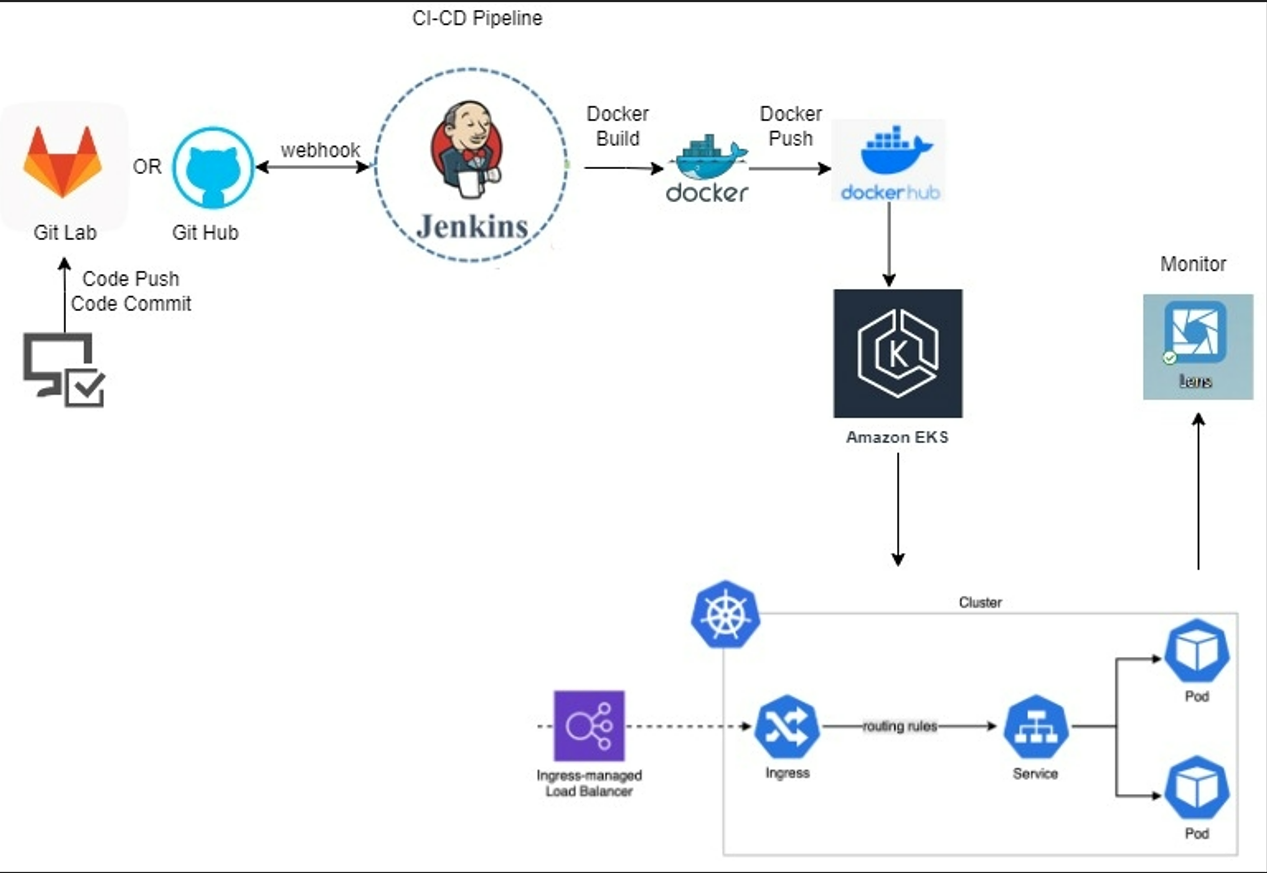
*Monitor*:grafana,lens,promotheus, AWS cloudwatch

**IDE:-**

Intellej

visual studio

**CI-CD HLD**:-



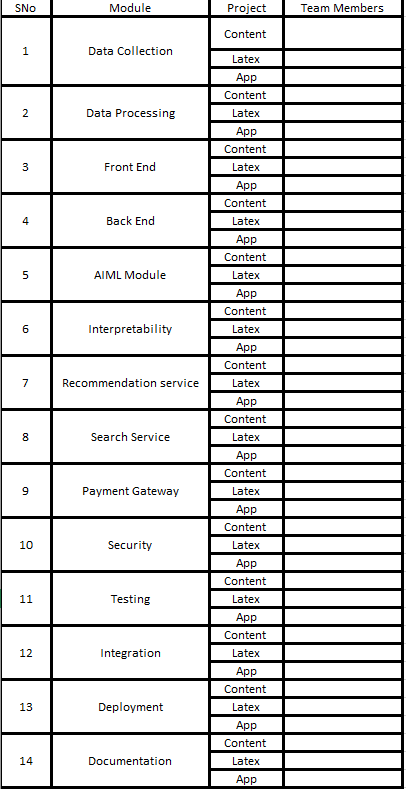
**\*Effort Estimation**

* It is dependent on Content and API requirements. Once it receives the architecture of content and API and Project requirement Document, then it will be estimated.

**Team:**

Team Leads:

1. Data Engineering - [Monideepa Roy](https://ninja.mygreatlearning.com/student_dashboard/student_index?custom_program_group_id=8786&student_id=2125407)
   1. Data Collection
   2. Data Preprocessing
   3. Data Bases maintenance
2. Front end - [Balasubramonian Kumaresan](https://ninja.mygreatlearning.com/student_dashboard/student_index?custom_program_group_id=8786&student_id=2152649)
3. Backend - [Manukumar Bettaswamaiah](https://ninja.mygreatlearning.com/student_dashboard/student_index?custom_program_group_id=8786&student_id=1987374)
   1. Development
   2. Data Bases design
4. AIML - [Bharat Vasireddy](https://ninja.mygreatlearning.com/student_dashboard/student_index?custom_program_group_id=8786&student_id=1620555)
   1. Data Pattern
   2. Recommendation
   3. Search
   4. Sentiment identification
5. Integration & Deployment - [Narayanan Kesavan](https://ninja.mygreatlearning.com/student_dashboard/student_index?custom_program_group_id=8786&student_id=2045962)
6. Mobile App - [Rajeev](https://ninja.mygreatlearning.com/student_dashboard/student_index?custom_program_group_id=8786&student_id=2281638)
7. Documentation - [Sunil Kumar K K](https://ninja.mygreatlearning.com/student_dashboard/student_index?custom_program_group_id=8786&student_id=2135180)
8. Testing - [Venkatesh Rapolu](https://ninja.mygreatlearning.com/student_dashboard/student_index?custom_program_group_id=8786&student_id=2125072)



**Mobile App:**

Important layers in mobile app architecture:

* Presentation layer
* Business layer
* Data access layer

Sample list of features:

| SNo | Feature | Primary | Secondary | Not Needed |  |
| --- | --- | --- | --- | --- | --- |
| 1 | Sign-up and login |  |  |  |  |
| 2 | Onboarding |  |  |  |  |
| 3 | Splash screen |  |  |  |  |
| 4 | Navigation |  |  |  |  |
| 5 | Image galleries |  |  |  |  |
| 6 | Forms |  |  |  |  |
| 7 | Social media integration |  |  |  |  |
| 8 | Social feeds |  | Yes |  |  |
| 9 | Product menus / Article Categories | Yes |  |  |  |
| 10 | Shopping carts and payments |  |  | Yes |  |
| 11 | Loyalty cards |  |  | Yes |  |
| 12 | Calendar integration |  |  |  |  |
| 13 | Push notifications |  |  |  |  |
| 14 | Device hardware access |  |  |  |  |
| 15 | App analytics |  |  |  |  |
| 16 |  |  |  |  |  |
| 17 |  |  |  |  |  |
| 18 |  |  |  |  |  |
| 19 |  |  |  |  |  |
| 20 |  |  |  |  |  |

Mobile App Team:

App Lead: Rajeev

Team:

| **Member** | **Exp** | **Present Industry Role** | **Company** | **Project Role** |
| --- | --- | --- | --- | --- |
| Rajeev | 8-12 Years | Senior Software Engineer (Android) | Intuit Inc | Team Lead |
| [Akhil Singla](https://ninja.mygreatlearning.com/student_dashboard/student_index?custom_program_group_id=8786&student_id=2219412) | 12-15 Years | Senior Architect | Persistent System | App Architecture and Design |
| Mayuri Musalgaonkar | 2-3 Years | Android application developer | Plug2field solution Pvt ltd |  |
| Rohina Naaz | 3-5 Years | Software Engineer - Frontend Developer | Society for Innovation and Development (SID) |  |
| Mangesh Kore | 3-5 Years | Sr. Software Engineer | Cybage Software Pvt. Ltd. |  |
| Nandhini J | 2-3 Years | Software engineer | Juhomi |  |
| Soumalya Khan | <1 Year | OE (Operation Executive) | TcsIon |  |
| Sasidaran J A | 1-2 Years | Associate Software Engineer | Macroid Apps Services Pvt Ltd, Navalur, Chennai. |  |
| Ayush Mer | 3-5 Years | Senior Software engineer | Samsung |  |
| N A Amogh | 1-2 Years | Software Engineer | Mindtree |  |
| Piyush Sahani | 3-5 Years | Software Engineer | Taskymonk |  |
| Rachit Kumar Rastogi | >15 Years | SDET | TD Ameritrade, Jersey City | App Integration Lead |
| Harshita Kurmala | 3-5 Years | ITANALYST | Tata Consultancy Services |  |
| Kanchana K | 0 Years | NA | NA | App Testing |

# **Data Engineering API Requirement Specifications**

# **Sub-Modules**

**1. Data Collection**

**2. Data Pre-processing**

**3. Database Maintenance**

**Data Engineering API Team Members**

PM: Binny Verghese

TL: Monideepa Roy

1.Satyam Shukla

2.Shivam Sangal

3.Soham Chakraborty

4.Ramesh Mantripagada

**Objective**

To extract, transform and load the data from various healthcare data sources for fetch by APIs

# **Tech Stack**

Python 3.10.4 can be used for data extraction, pre-processing, module creation,API endpoints with flask framework

* MongoDB can be used for storing unstructured data,image data
* Cloud resources as Azure CosmosDB on free subscription
* Amazon S3 can be used to store huge sized files in the order of GBs
* MySQL can be used for storing structured data, user information.

Code repository: GitHub

# **Open-Source Data Links for Data Collection**:

1. <https://data.mendeley.com/datasets/rscbjbr9sj/2>

Chest X-Ray Images (used for pneumonia detection, lung cancer detection as example use cases)

Suggested DB: NoSQL DB: MongoDB

Or Amazon S3 can be used as a file store.

2. <https://github.com/wooque/openpacs>

3. <https://archive.ics.uci.edu/ml/index.php>

# **Python Libraries:**

Pandas for tabular data

Nltk, spacy for text processing

Pillow for image processing

Beautiful Soup for web scraping

Flask framework for APIs

# **Literature Survey**:

<https://towardsdatascience.com/python-etl-tools-best-8-options-5ef731e70b49>

<https://ieeexplore.ieee.org/abstract/document/1347934>

<https://www.researchgate.net/profile/Maria-Nancy-3/publication/343655656_A_REVIEW_ON_UNSTRUCTURED_DATA_IN_MEDICAL_DATA/links/5f36a848a6fdcccc43c6b2d5/A-REVIEW-ON-UNSTRUCTURED-DATA-IN-MEDICAL-DATA.pdf>

<https://www.analyticsvidhya.com/blog/2022/03/a-quick-overview-of-data-engineering/>

https://docs.microsoft.com/en-us/azure/cosmos-db/

# **Reference sites (how they are hosted on cloud):**

Generalized API platform: <https://h2o.ai/>

APIs for healthcare: <https://monai.io/>

APIs for finance :<https://www.capfront.in/>

APIs for logistics, Education and Retail : <https://www.tensorbrew.com/>

# **Knowledge Sources**:

<https://feedly.com/i/top/data-engineering-blogs>

<https://www.youtube.com/watch?v=0buKQHokLK8> -- relational db vs nosql db

<https://www.youtube.com/watch?v=FW1LOP09RM8> --- Flask vs Django

<https://www.guru99.com/flask-vs-django.html#:~:text=Flask%20provides%20support%20for%20API,for%20easy%20and%20simple%20projects.>

<https://www.lastweekinaws.com/blog/10-free-cloud-databases-you-should-consider-and-1-you-shouldnt/>

—-----------------------------------------------------------------------------------------------------------------------

# 

# **Data Engineering ( Content ) Requirement Specifications**

# **Sub-Modules**

1. Data Collection

2. Data Pre-processing

3. Database Maintenance

# **Data Engineering Content Team Members**

PM: Shyam Nair

TL: Monideepa Roy

1.Nikhil Thorawade

2.Digvijay Kumar

3.Narendra Kumar

4.Pasupuleti Venkateswara Rao

# **Objective**

To extract, transform and load articles/documents for content website

# **Tech Stack**

Python 3.10.4 can be used for data extraction, pre-processing

MongoDB can be used as the nosql DB

Cloud resources such as MongoDB Atlas /Azure CosmosDB/ AWS DynamoDB can be used with the free subscription.

Code repository: GitHub

# **Open-Source Data Links**:

<https://medium.com/>

# **Python Libraries:**

Beautiful Soup for web scraping

# **Literature Survey**:

<https://callygood.medium.com/6-methods-of-data-collection-e946e993b930>

# **Knowledge Sources**:

## Free MongoDB hosting

Different options compared below

<https://studio3t.com/knowledge-base/articles/cheap-free-mongodb-hosting/>

### MongoDB Atlas

<https://www.mongodb.com/cloud/atlas/register>

* Free Registration, No Credit Card required
* 512 MB Free Cluster

### MongoDB tutorials

* <https://www.youtube.com/watch?v=ofme2o29ngU>
* <https://www.youtube.com/watch?v=DZBGEVgL2eE>

## WebSite Bot for Content

* Need to decide the technology for scraping through the website downloading the articles to store in DB.
* Need to decide indexing logic for searching functionality on the articles.

**Front End Project Requirement**

**Front End Content**

**High level flow:** User lands on the Home Page after successful login. Users can see all the articles posted in the Blog. Users can search articles by topic/author etc. Users also can login to the blog using credentials of the site also using facebook/ google credentials. Users can post an article. Users can also edit an article already posted. Users can edit only their articles. Users can subscribe to a payplan to access premium articles.

**Technology** : HTML, CSS, Bootstrap, Javascript, React JS

**Reference Site**: <https://medium.com>

**Literature survey:**

Ease of Navigation

Displaying all the available features/functionality

Search bar for topic title, author etc

Grouping the articles by topics like sports, science etc.

Filters to select articles of specific topic

Titles with icon for each subject

Payment gateway

Easy login experience

Tool tips

**Knowledge Source**

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4974011/>

<https://multichannelmerchant.com/blog/payment-gateways-101-examples-benefits-and-drawbacks/>

<https://github.com/ravigithub19/payment-gateway-integration>

<https://www.c-sharpcorner.com/article/login-with-facebook-using-reactjs/#:~:text=npx%20create-react-app%20sociallogin%20Open%20the%20newly-created%20project%20in,using%20the%20following%20command%20npm%20install%20react-facebook-login%20--save>

https://developers.google.com/identity/sign-in/web/sign-in

**Front End API**

**a)API Website**

**High level flow:** User lands on the Home page. Users can see the list of Models available for training grouped subject wise. Users also can login to the site using credentials of the site also using facebook/ google credentials. Users need to enroll in a payplan to select and run the model. When the user selects and submits a model for training, the UI will display the progress bar until the process is running and display the result after the process is completed.

**Technology** : HTML, CSS, Bootstrap, Javascript, React JS

**Reference Site**: <https://h2o.ai/>

Ease Navigation

Displaying all the available features/functionality

Search bar to find a specific model/ subject wise etc.

Filter the models by subject wise Like Medical, Hotel etc

Predefined filters for Models with icon/subject for each group/ subject

Payment gateway

Easy login experience

Tool tips

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4974011/>

<https://multichannelmerchant.com/blog/payment-gateways-101-examples-benefits-and-drawbacks/>

<https://github.com/ravigithub19/payment-gateway-integration>

<https://www.c-sharpcorner.com/article/login-with-facebook-using-reactjs/#:~:text=npx%20create-react-app%20sociallogin%20Open%20the%20newly-created%20project%20in,using%20the%20following%20command%20npm%20install%20react-facebook-login%20--save>

<https://developers.google.com/identity/sign-in/web/sign-in>

**b) IDE for Python**

Developers can view and edit the workspace files.

**Technology** : HTML, CSS, Bootstrap, Javascript, React JS

**Reference Site**: https://www.spyder-ide.org/

**Documentation :**

Google Docs

OpenAPIEditor

<https://opensource.com/resources/python/ides>

<https://www.spyder-ide.org/>