Santiago Andrés Jaramillo

[Email address]

Sitecore Hackathon 2021

Website developer with Sitecore 10.1

Wolfensteing team

User Guide

# Introduction

## It is common for many electronic devices such as smartphones and tablets to get a smart keyboard. What does it mean? Devices can suggest words or frasses while users are writing text messages. So, what had happened with the old editors for Content Manager software? Common rich editors do not get smart behavior to help their users.

## This project's main idea is to add simple functionality to suggest content authors text or frasses to add in their content. Since here, new possibilities are opened to add new functionalities to help content authors. For example. Rich text editors can get an option to analyze the text to predict what kind of sentiment their text produces.

## All information related to the text is valuable for content authors to improve their quality content.

## Predictive Text Search

This project built a machine learning model to suggest text based on an initial text. The model was coded in Python. The model is running in a public service. Sitecore consumes those services using javascript.

The main structure is.

|  |
| --- |
| def generateText():  """  Text Recommender API  Call this api to send text and get an array of recommended text  Parameters must be passed as multipart-form  ---  TAGS:  - Text Recommender API  PARAMETERS:  - text: string  in: string  type: string  required: true  description: Text that is going to be used as base for the generated text  - length: string  in: string  type: int  required: true  description: The number of words to be generated  - sequences: string  in: string  type: int  required: true  description: The number of sequencesto generate  RESPONSES:  500:  description: Error with the parameters, text or length!  200:  description: the original text and generated text are going to be responded  schema:  properties:  originalText:  type: string  description: original text recieved  default:  generatedTexts:  type: array  description: array of strings that content the generated text  default: []  items:  type: string  """  print("enter in generation")  if request.method == 'POST':  content = request.json  text = content['text'].strip()  length = int(content['length'])  sequences = int(content['sequences'])  print(text)    generated\_texts = get\_generated\_text(text, length, sequences)  return jsonify({'originalText': text, 'generatedTexts': generated\_texts})  @app.route('/api/sentiment-text', methods=['POST'])  @auto.doc()  @cross\_origin() |