**Toxic Comment Classification**

***Western Governors University***

***Antonio B Jenkins***

Table of Contents

[**Prompt A** **4**](#_Toc25684)

[Letter of Transmittal 4](#_Toc25685)

[Project Recommendation 7](#_Toc25686)

[Problem Summary 7](#_Toc25687)

[Application Benefits 7](#_Toc25688)

[Application Description 8](#_Toc25689)

[Data Description 8](#_Toc25690)

[Objective and Hypothesis 9](#_Toc25691)

[Methodology 9](#_Toc25692)

[Funding Requirements 10](#_Toc25693)

[Stakeholders Impact 10](#_Toc25694)

[Data Precautions 11](#_Toc25695)

[Developer Expertise 11](#_Toc25696)

[**Prompt B** **13**](#_Toc25697)

[Project Proposal 13](#_Toc25698)

[PROBLEM STATEMENT 13](#_Toc25699)

[CUSTOMER SUMMARY 13](#_Toc25700)

[EXISTING SYSTEM ANALYSIS 14](#_Toc25701)

[DATA 14](#_Toc25702)

[PROJECT METHODOLOGY 15](#_Toc25703)

[PROJECT OUTCOMES 16](#_Toc25704)

[IMPLEMENTATION PLAN 16](#_Toc25705)

[EVALUATION PLAN 18](#_Toc25706)

[RESOURCES AND COSTS 19](#_Toc25707)

[TIMELINE AND MILESTONES 19](#_Toc25708)

[**Prompt C** **21**](#_Toc25709)

[Application Files 21](#_Toc25710)

[**Prompt D** **22**](#_Toc25711)

[Post-implementation Report 22](#_Toc25712)

[Project purpose 22](#_Toc25713)

[Datasets 23](#_Toc25714)

[Data product code 24](#_Toc25715)

[Hypothesis verification 25](#_Toc25716)

[Effective visualizations and reporting 25](#_Toc25717)

[Accuracy analysis 26](#_Toc25718)

[Application testing 26](#_Toc25719)

[**Appendices** **27**](#_Toc25720)

[Installation Guide 27](#_Toc25721)

[User Guide 29](#_Toc25722)

[Summation of Learning Experience 30](#_Toc25723)

[References 31](#_Toc25724)

# Prompt A

## Letter of Transmittal

Dear Stakeholder,

In recent times, the proliferation of online platforms has led to an increase in the exchange of user-generated content. This content, while fostering communication and community building, also includes a significant portion of toxic comments that can be harmful and disruptive. The presence of such comments can degrade the quality of discourse, discourage participation, and negatively impact the reputation of the platform. As a result, there is a pressing need for effective moderation tools that can identify and mitigate the spread of toxic content.

Our team at Reign Toxicity (RT) has developed a solution that leverages the latest advancements in natural language processing and machine learning. Our system uses BERT (Bidirectional Encoder Representations *from* Transformers) in conjunction with a Convolutional Neural Network (*CNN*) to accurately classify comments as toxic or non-toxic. BERT's deep learning capabilities, pre-trained on a vast corpus of text, enable it to understand the nuances of language, while the CNN layer helps in capturing the local patterns within the text, making our classification both robust and sensitive to context.

The objective of our application is to provide an automated, scalable, and accurate tool for content moderation. By integrating our system into existing platforms, we can significantly reduce the manual effort required for moderation, while maintaining a high standard of discourse. Our initial tests have shown promising results, with high accuracy in detecting various forms of toxicity such as insults, threats, and hate speech.

We understand the importance of maintaining user privacy and ensuring the ethical use of our technology. Our system is designed to process data in a way that respects user anonymity and complies with all relevant data protection regulations. We are committed to continuous improvement and are seeking partnerships to further refine and adapt our solution to meet the specific needs of different online communities.

In terms of funding, we are seeking an initial investment to cover the costs of further development, testing, and deployment. Our projected timeline for the initial rollout is 8 weeks, with subsequent updates and feature enhancements planned based on user feedback and technological advancements.

We are excited about the potential of our system to create a safer and more welcoming online environment, and we look forward to the opportunity to discuss this further with you.

Sincerely,

Antonio, Lead Developer RC

**Project Recommendation**

### Problem Summary

### In today's digital landscape, online platforms frequently grapple with the pervasive issue of toxic commentary. This type of content spans a broad spectrum, from blatant hate speech to more insidious forms of verbal aggression. Such toxicity not only degrades the overall user experience but also harbors significant legal and reputational ramifications for the companies that host these platforms.

### Our endeavor confronts this challenge head-on. We are in the process of developing and deploying an advanced Natural Language Processing (NLP) system, tailored to effectively identify and address these toxic elements. This system leverages cutting-edge machine learning algorithms and natural language understanding techniques. It is designed to parse through vast quantities of text data, accurately identifying patterns and nuances that signify toxic content.

### The core of our system lies in its sophisticated algorithmic framework, which has been meticulously trained on diverse datasets. This training enables our model to distinguish between harmful and harmless content with remarkable precision. Moreover, our approach is not static. We continually refine our model, incorporating feedback and adapting to evolving linguistic trends, ensuring that our system remains effective against an ever-changing digital backdrop.

### Our project aims not just at mitigating the immediate risks associated with online toxicity but also at fostering a more respectful and constructive online discourse. By implementing this NLP system, we strive to create a safer digital environment, one where interactions are rooted in respect and understanding, rather than hostility and aggression. This initiative represents a significant step forward in our commitment to leveraging the power of machine learning for the greater good, ensuring a healthier digital ecosystem for users and a more secure operational framework for companies.

### Application Benefits

### The application uses BERT for toxic comment classification. BERT's architecture understands language context, improving over traditional models. It accurately identifies toxic comments, crucial for maintaining a respectful online environment. The use of BERT enhances user experience, leading to better engagement and retention. It also reduces manual moderation costs, providing a scalable solution. By using this application, platforms can promote safety and inclusivity, attracting more users and potentially increasing revenue. In essence, BERT's NLP technology is key to advancing online communication and fostering a positive digital community.

### Application Description

### The Toxic Comment Classification application employs BERT's advanced NLP to ensure top-tier content moderation. It recognizes toxic comments, which can be subtle, and moderates’ content in real-time. The app's sensitivity settings are adjustable to match community guidelines, emphasizing data security and user privacy. Its scalable design integrates well with existing platforms, handling large data volumes efficiently. The dashboard provides insights into user behavior and moderation trends, aiding continuous system refinement. Developed with Python and TensorFlow, the application leverages high-performance machine learning and cloud infrastructure. It goes beyond simple moderation, aiming to enhance digital interactions and ensure safe, inclusive spaces across online platforms.

### Data Description

### The foundation of our Toxic Comment Classification application is solidly built on the extensive dataset from Kaggle's Toxic Comment Classification Challenge. This dataset is a rich compilation of user-generated comments, drawn from a wide online community spectrum. Each comment in this collection has been carefully labeled to reflect varying degrees of toxicity. This labeling covers a broad range of toxic behaviors, including but not limited to threats, obscenity, insults, and identity-based hate. This categorization offers a detailed view of the negative interactions prevalent online.

### Stored in a CSV file format for practicality, this dataset serves as a crucial tool for both training and testing our machine learning model. It immerses the model in a realistic linguistic landscape, brimming with the complexities of everyday language, including colloquialisms, slang, and dialects. Such a diverse linguistic environment is essential for the model to accurately grasp and adapt to the intricacies of human communication.

### The depth and variety of this dataset are key in enhancing the model's ability to classify comments correctly. It plays a significant role in reducing bias, ensuring that the application functions effectively across various platforms and user groups. Moreover, the structured nature of the dataset simplifies data handling and analysis, facilitating its seamless integration into our machine learning workflow.

### In essence, this comprehensive and carefully annotated dataset is a cornerstone in the development of an application that is not only accurate in identifying toxic comments but also astute in understanding the context of such interactions.

### Objective and Hypothesis

### The goal of our project is to create a sophisticated NLP BERT-based system decreasing the level of toxic commenting on online platforms by 50% within half a year. We strongly believe that the BERT model's deep language context understanding will greatly enhance the level of accuracy and efficiency of existing content moderation tools.

### We hypothesize that by leveraging BERT's advanced NLP capabilities, detecting different toxic behaviors, compromising a wide range of explicit aggression through to more implicit negative interactions, will see meaningful improvement. This arises from BERT's special capability of comprehending the language context aside from its subtleties, which offers analysis beyond the reach of standard NLP methodologies.

### Ultimately, the project aims to revolutionize the process of online content moderation with this technological advancement by improving the quality of interactions and discourses within digital communities. Our goal is helping make safer, more respectful online spaces where everyone can be part of a better experience.

### Methodology

### Agile Framework

### Our project employs the Agile methodology, a strategy focused on adaptability, efficiency, and continual improvement. This approach is particularly effective for dynamic projects like ours, as it allows us to swiftly respond to changes and integrate feedback. Agile emphasizes collaboration between teams and stakeholders, ensuring that the development process is aligned with the project's goals and adapts to evolving needs.

### Phased Development Approach

### The development of the Toxic Comment Classification application is executed in distinct phases. Initially, our focus is on delivering a core version of the application that establishes the fundamental functionalities. This phase is crucial for laying down a strong foundation for the project. Subsequently, we enter a period of refinement and enhancement, where feedback from initial testing is used to fine-tune the application. This phased approach not only facilitates a faster initial rollout but also ensures continuous improvement based on practical insights.

### Regular Updates and Feedback Inclusion

### A cornerstone of our methodology is the regular provision of updates and the active inclusion of stakeholder feedback. This process ensures that the development remains in line with your strategic objectives and that any concerns or required changes are addressed efficiently. Regular updates provide transparency and facilitate open communication, key elements for successful project management.

### User Experience and Privacy Consideration

### In addition to technical development, we place significant emphasis on user experience and data privacy. The application is designed to be intuitive and easy to use, ensuring user satisfaction. Simultaneously, we prioritize data privacy and adhere to all relevant data protection regulations, recognizing the importance of user trust and security in today’s digital landscape.

### Funding Requirements

### The financial plan for the development and implementation of our application is outlined in the Letter of Transmittal, and we reiterate the key points here for clarity. While the core technologies we are utilizing, including Python and its extensive range of libraries, are open-source and incur no direct costs, there are significant other expenses that must be considered. These include the remuneration for the development team, the cost of server space for hosting the application, necessary hardware for development and testing, and the salaries of all employees involved in the project.

### To support these requirements, we are proposing a funding model that consists of two separate installments of $150,000 each. These installments are strategically timed to coincide with the commencement of the project's two main phases. The first installment will facilitate the initial development and pilot testing, while the second will cover the full-scale deployment and further refinement of the system.

### In addition to the funding installments, we project that the initial phase of the project will generate a net income of $200,000. This revenue is expected to be reinvested into the project, offsetting part of the costs for the second phase. Therefore, the total funding requirement for the project is projected to be $500,000. It is important to note that if the income generated in the first phase falls short of our projections, the shortfall will need to be added to the second funding installment to ensure the project's continuity and success.

### Stakeholders Impact

### The success of our Toxic Comment Classification system hinges on meeting the expectations and needs of all stakeholders involved. This alignment is crucial, especially when integrating our BERT/NLP-driven system into various platforms. Clear communication and regular updates will play a key role in ensuring transparency and building trust throughout the project's life cycle.

### For our stakeholders to consider this project a success, it needs to hit several important targets:

### Operational Excellence: The application needs to run smoothly with minimal glitches. Keeping technical issues to a bare minimum is essential to maintain user trust and the integrity of the platforms it's used on.

### Ethical Adherence: Our work is defined by a strong commitment to ethics. This means our application must stick to these ethical standards, including user privacy, data security, and transparent content moderation practices.

### Privacy and Noise Control: It's vital that the application balances filtering out unwanted content with respecting user privacy. The system should effectively sift through toxic comments while keeping user data secure and anonymous.

### Financial Impact: A key measure of success will be the financial benefit for the stakeholders. We're expecting the application to generate significant new income once fully operational on our existing customer base. Also, marketing this application as a standalone product could open further revenue opportunities.

### Customer Satisfaction and Retention: Beyond just the financials’, how satisfied customers are and whether they stick around will be a big indicator of success. We're aiming for an increase in user retention and a reduction in moderation costs, all thanks to the application’s ability to improve online environments.

### Scalability and Portability: The design of the system should allow it to grow and be used on different platforms, increasing the potential for more impact and revenue.

### Hitting these objectives is not just about advancing our technology. It's about creating safer, more respectful digital spaces and delivering real value to Secure Enterprise Solutions and all other stakeholders involved.

### Data Precautions

### In the development and deployment of the Toxic Comment Classification application, paramount importance is placed on data security and ethical handling, especially considering the sensitive nature of the user-generated content being analyzed. The utmost care is taken to ensure that all data processing complies with relevant data protection regulations, including GDPR and other privacy standards. The application is designed to operate on an anonymization principle, meaning that while comments are analyzed for toxicity, the identities of the commenters are neither required nor retained, thus preserving user anonymity. This approach is crucial in maintaining user trust and safeguarding against potential data misuse. Also, strict controls are in place to prevent the storage of user data beyond what is necessary for real-time analysis, ensuring that data is not susceptible to unauthorized access or breaches. The application also adheres to a strict ethical code to prevent any potential bias in content moderation, and regular audits are conducted to ensure that the machine learning algorithms remain fair and unbiased. By prioritizing these data precautions, the project not only upholds high ethical standards but also reinforces its commitment to responsible and respectful handling of user data, ensuring the application's credibility and trustworthiness in the eyes of both platform owners and their users.

### Developer Expertise

### The development team for this project will consist predominantly of junior software developers, carefully selected for their potential and foundational skills. To ensure the project's success and adherence to our ambitious timeline, we will be joined by two seasoned software experts who will oversee the development process and serve as scrum masters. This strategic decision is driven by our commitment to expanding our workforce, equipping us to scale our operations in line with the growing demand for our application. By integrating these junior developers into the project, we aim to foster a learning environment that accelerates their professional growth. This hands-on experience will be invaluable, not only in delivering a robust product but also in preparing them for future challenges, enabling them to contribute to more complex projects as they advance in their careers.

# Prompt B

## Project Proposal

### PROBLEM STATEMENT

### Our project addresses the widespread issue of toxic comments online, which negatively affects user experience and can harm the reputation of digital services, with possible legal consequences. We've built an automated system for moderating content that uses a BERT-based NLP model capable of understanding subtle differences in language. This is key for accurately detecting and classifying various types of toxic content, including insults, threats, and hate speech. The system includes a continuous data ingestion pipeline to adapt to changes in online communication, maintaining its moderation effectiveness. This solution is designed to improve the quality of online interactions and reduce the presence and impact of damaging content on digital platforms.

### CUSTOMER SUMMARY

### The application targets admins and moderators dealing with content moderation challenges. It leverages a BERT-based NLP model to automate the detection and categorization of toxic comments such as insults, threats, and hate speech. This reduces manual effort and increases moderation accuracy. A data ingestion pipeline keeps the system updated with new communication patterns, maintaining its effectiveness. The goal is to create a safer online environment, protect digital service reputations, and enhance user experience.

### EXISTING SYSTEM ANALYSIS

### The current content moderation system relies on basic rule-based algorithms and simple machine learning models. These methods fail to capture the complexities and context of human language, resulting in a high number of false positives and inefficiency in moderating content. Our upgraded system will incorporate advanced Natural Language Processing (NLP) techniques, with a focus on the BERT (Bidirectional Encoder Representations from Transformers) model. This will improve the system's ability to understand context and nuances in text. We will develop the system in Python 3.7, using libraries such as Transformers for the BERT model, and TensorFlow or PyTorch for training and running the model, along with Scikit-learn for other machine learning tasks. The goal is to decrease false positives and make the moderation process more efficient, enhancing the accuracy and efficiency of content moderation.

### DATA

### Our content moderation system leverages a BERT-based NLP model trained on a dataset from Kaggle's Toxic Comment Classification Challenge. This dataset includes user comments labeled for toxicity levels, such as insults, threats, and hate speech. The detailed labels enable the model to identify subtle language differences, reducing false positives and increasing moderation accuracy. We will implement a data pipeline for continuous live data ingestion, allowing the model to adapt to new patterns in online communication. This keeps the model up to date with current linguistic trends, maintaining its content moderation effectiveness. The dataset is structured for efficient processing, with normalized and referenced repeating elements to decrease redundancy and speed up data retrieval. Our system is built to integrate updates easily, including adding new content categories or updating annotations to match evolving language use.

### PROJECT METHODOLOGY

### Agile Development with Technical Specificity

### Adopting an Agile methodology, the project is tailored to meet the specific technical challenges presented by advanced machine learning technologies like BERT, CNN, and NLP. Agile's flexibility is crucial for iterative development and integration of these complex components, allowing for rapid adaptation and continuous enhancement.

### Phase-wise Technical Implementation

### The development process is strategically divided into phases, each focusing on a specific technical aspect of the project. Starting with data modeling in Python, we progress to the integration of BERT NLP advanced text analysis. This structured approach ensures logical and efficient progression in developing the application, with each phase building upon the successes of the previous one.

### Feedback Incorporation and Rigorous Testing

### Incorporating feedback is especially critical from a technical perspective. Each development cycle includes thorough testing of new functionalities, ensuring they meet our rigorous technical standards and requirements. This process is crucial for maintaining the quality and reliability of the application.

### Focus on Scalability and Security

### Given the application's technical complexity, special attention is paid to scalability and security. We ensure that the system is capable of handling extensive data and user loads without compromising performance. Security and data privacy are paramount, with stringent measures in place to protect user data and comply with data protection laws.

### Continuous Integration and Deployment (CI/CD)

### Employing CI/CD practices allows for frequent and efficient updates, automated testing, and consistent deployment of the application. This practice ensures that the application remains current with the latest technological advancements and maintains high security and performance standards.

### PROJECT OUTCOMES

### The completion of the Toxic Comment Classification project marks a significant milestone in the field of content moderation. Central to this achievement is the delivery of an AI-powered application, specifically engineered for efficiency and precision. This application harnesses the power of BERT and NLP, forefront technologies in machine learning, to ensure effective management of online interactions.

### Key to this application is its graphical user interface, distinct for its standalone design. This interface is crafted to provide effortless and intuitive access for both users and administrators. It's been engineered with the user in mind, simplifying interactions with the system's advanced features, such as real-time moderation and comprehensive reporting capabilities.

### Accompanying the technical components is a thorough user guide. This manual is a crucial resource, providing clear installation instructions and offering a detailed explanation of the application's functionalities. It's designed to assist users in effectively navigating and utilizing the system, ensuring they can make the most of its advanced BERT and NLP capabilities.

### Also included in the deliverables is a detailed project schedule. This document outlines both the projected and actual dates for key milestones, providing a clear view of the project's progress. This level of transparency is vital for understanding the development process and highlights the meticulous planning and execution behind this sophisticated application.

### In summary, the successful completion of the Toxic Comment Classification project delivers an advanced, AI-powered application. It's not just a technical solution but a strategic tool that leverages the latest advancements in BERT and NLP for improved content moderation.

### IMPLEMENTATION PLAN

### Phase 1: Data Model and Machine Learning Pipeline Setup

### The first phase focuses on data preparation and model framework development. This involves sourcing and preprocessing the dataset from Kaggle's Toxic Comment Classification Challenge, which includes cleaning, tokenizing, and structuring the data for machine learning. Using Python, we will define and implement data models tailored for processing toxic comments. Simultaneously, we will establish a foundational machine learning pipeline integrating libraries like TensorFlow or PyTorch. This pipeline will be the basis for the subsequent integration of the BERT NLP models, setting the stage for advanced text analysis.

### Phase 2: Development of Core Machine Learning Components

### In this critical phase, we will integrate the BERT model to understand the contextual nuances in user comments. Fine-tuning BERT with our specific dataset is essential for its effectiveness in identifying various forms of toxic content. Concurrently, we will develop a CNN layer that complements BERT, focusing on detecting local patterns within text data. This synergy between BERT and CNN is pivotal in enhancing classification accuracy. Also, we will incorporate other NLP features and algorithms to augment the system's capability in understanding and classifying text accurately.

### Phase 3: User Interface and System Integration

### The development of a user-friendly interface for the stand-alone application is the focus of this phase. We will employ Python-based frameworks like Django or Flask, not for web development, but to create a robust and intuitive GUI for desktop environments. This interface will allow users to interact seamlessly with the machine learning models, facilitating easy submission of text for classification and viewing the results. Ensuring smooth integration of the machine learning components with the GUI is crucial for real-time data processing and user experience.

### Phase 4: Testing and Refinement

### Comprehensive testing forms the core of this phase. Each module, including BERT, CNN, and other NLP features, will undergo rigorous testing to ensure optimal functionality. This will be followed by end-to-end system testing to validate the overall performance of the application, including the GUI. Based on the feedback from these tests, we will refine and optimize the application. This iterative process aims to improve accuracy, reduce response time, and enhance the overall user experience.

### Phase 5: Deployment and Documentation

### Preparation for deployment involves setting up the environment with all necessary Python dependencies and machine learning libraries. The application will then be deployed as a stand-alone product, ensuring it is scalable, secure, and robust. Comprehensive documentation covering the codebase, machine learning models, GUI operations, and system maintenance will be prepared, providing users and maintainers with a detailed guide for navigating and managing the application.

### Phase 6: Post-Deployment Support and Maintenance

### After deployment, ongoing monitoring and maintenance will be crucial to ensure the application’s consistent performance. Regular updates and improvements will be made based on user feedback and emerging requirements. This phase is dedicated to the continuous evolution of the application, adapting to new challenges and advancements in the field of machine learning and NLP.

### EVALUATION PLAN

### Our comprehensive evaluation plan for the Toxic Comment Classification project integrates several critical phases to ensure the effectiveness and reliability of the system, particularly focusing on the advanced machine learning and AI technologies employed.

### In the initial phase, our primary focus is on evaluating the performance of the BERT NLP models. This evaluation involves measuring key metrics such as precision, recall, and F1-score against a carefully curated test dataset. The aim is to validate the accuracy of the models in identifying various forms of toxic comments. Concurrently, AI Engineers will undertake an in-depth analysis for model optimization, fine-tuning hyperparameters to balance speed and accuracy effectively. A crucial aspect of this phase is the assessment of data bias and fairness, where our Data Scientists will scrutinize the models against diverse datasets to ensure unbiased and fair predictions.

### Proceeding to the second phase, the evaluation extends to the user interface and its integration with the backend machine learning models. The UI/UX Designer spearheads the user experience testing, emphasizing the interface's usability, responsiveness, and overall user satisfaction. In parallel, Software Engineers conduct rigorous integration tests to ensure seamless interaction between the machine learning components and the user interface, verifying efficient data flow and real-time processing capabilities.

### The third phase encompasses end-to-end system testing, spearheaded by the Quality Assurance team. This comprehensive testing covers all application aspects, including stress testing under various load conditions to ensure the system's stability and scalability. The DevOps Engineer plays a pivotal role in validating the application's deployment readiness, ensuring that the transition to a production environment is smooth and devoid of any performance or availability issues.

### Finally, in the post-deployment phase, IT Support takes the lead in continuously monitoring the application, tracking its performance in a real-time operating environment, and identifying any potential issues. An integral part of this phase is establishing a feedback loop, wherein regular user and stakeholder feedback is collected and analyzed. This ongoing process enables the incorporation of practical insights into future updates and improvements, ensuring that the application remains effective and relevant in real-world scenarios.

### RESOURCES AND COSTS

### Personnel Costs

### The project's success hinges on the expertise and coordination of our specialized team, comprising a Project Manager, Data Scientist, AI Engineer, Software Engineer, Quality Assurance personnel, UI/UX Designer, DevOps Engineer, and IT Support. The Project Manager, at $100 per hour, will require $1,000 for the initial project planning. The Data Scientist, pivotal in data preparation, is budgeted at $120 per hour, totaling $3,600. AI Engineers, central to model development and refinement, will cost $140 per hour, amounting to $15,400. Software Engineers, tasked with development and deployment, are estimated at $110 per hour, summing up to $10,450. Quality Assurance experts, crucial for testing, are budgeted at $90 per hour, amounting to $4,950. The UI/UX Designer, responsible for the user interface, will cost $100 per hour, totaling $4,000. For deployment preparation, the DevOps Engineer is estimated at $120 per hour, totaling $2,400. Lastly, IT Support for post-deployment monitoring is estimated at $80 per hour, amounting to $2,400.

### Technology and Infrastructure Costs

### Our technology and infrastructure costs encompass cloud computing resources, essential for model training and deployment, estimated at $1,000 per month, totaling $4,000 for the project duration. Software licensing for development tools and machine learning libraries is projected at $2,500. Also, the setup for development and testing environments requires a budget of $5,000.

### Miscellaneous Costs

### To maintain our team's edge in technology, we've earmarked $3,000 specifically for training and workshops. This investment is crucial for staying updated with the latest advancements and enhancing our skills. In addition to this, recognizing the unpredictability of project developments, we've also allocated a contingency fund. This fund, amounting to about 10% of our total expenses, roughly calculates to $6,170. It's a strategic reserve to address any unexpected costs that might emerge as we progress with our work. This financial planning underscores our commitment to both preparedness and continuous learning.

### Total Estimated Costs

### The total personnel costs for the project stand at $44,200. Combined with technology and infrastructure costs of $11,500 and miscellaneous costs of $6,000, the overall estimated budget for the Toxic Comment Classification project reaches approximately $67,870. This comprehensive budget plan ensures that all aspects of the project, from human resources to technological needs, are adequately funded to achieve the project's goals effectively.

### TIMELINE AND MILESTONES

### The timeline and milestones for the Toxic Comment Classification project have been meticulously planned to ensure a structured and efficient development process. Spanning over a period of approximately three months, the project is segmented into twelve distinct milestones, each tailored to a specific set of activities essential for the progression of the project. The timeline commences with the project kickoff and planning, led by the Project Manager, and progressively moves through crucial stages such as data acquisition, model development, prototype creation, and comprehensive testing. Key roles like AI Engineers, Data Scientists, and Software Engineers are strategically assigned to respective milestones, ensuring that each phase benefits from specialized expertise. This structured approach allows for focused development and testing phases, ensuring that every component of the application, from the machine learning algorithms to the user interface, is rigorously developed and evaluated. The latter part of the timeline is dedicated to final testing, deployment preparation, and post-deployment monitoring, ensuring that the application is not only functional but also robust and user-friendly upon launch. This careful planning and allocation of resources aim to ensure that the project adheres to its schedule, meets its objectives, and successfully achieves a high standard of quality and performance upon completion.

| **Milestone** | **Prerequisites** | **Activity** | **Resource Assigned** | **Hours** | **Start Date** | **End Date** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | - | Project kickoff and planning | Project Manager | 10 | 01/15/2024 | 01/16/2024 |
| 2 | 1 | Data acquisition and preprocessing | Data Scientist | 30 | 01/17/2024 | 01/20/2024 |
| 3 | 2 | Initial model development (BERT, NLP) | AI Engineer | 60 | 01/21/2024 | 01/28/2024 |
| 4 | 3 | Prototype development | Software Engineer | 40 | 01/29/2024 | 02/03/2024 |
| 5 | 4 | Prototype testing and feedback | Quality Assurance | 30 | 02/04/2024 | 02/07/2024 |
| 6 | 5 | Model refinement and optimization | AI Engineer | 50 | 02/08/2024 | 02/14/2024 |
| 7 | 6 | User Interface development | UI/UX Designer | 40 | 02/15/2024 | 02/20/2024 |
| 8 | 7 | Integration testing | Software Engineer | 35 | 02/21/2024 | 02/25/2024 |
| 9 | 8 | Final user acceptance testing | Quality Assurance | 25 | 02/26/2024 | 02/29/2024 |
| 10 | 9 | Deployment preparation | DevOps Engineer | 20 | 03/01/2024 | 03/03/2024 |
| 11 | 10 | Final deployment and launch | Software Engineer | 15 | 03/04/2024 | 03/05/2024 |
| 12 | 11 | Post-deployment monitoring and support | IT Support | 30 | 03/06/2024 | 03/09/2024 |

# Prompt C

## Application Files

\Jenkins\_Antonio\_BERTNLP\_Capstone

\data

sample\_submission.csv

test\_labels.csv

test.csv

train.csv

\bert\_toxic\_model.pth

\bert\_train.py

\bertnlp\_model.ipynb

\predict.py

\README

\requirements.txt

- bertnlp\_model.ipynb:  contains the code for setting up, training, and evaluating a BERT model for the task of classifying text into multiple categories of toxicity. It includes code for data preparation, model definition, training loop, and evaluation metrics. The notebook also contains widget state information for interactive elements used during the training process.  
  
- README: This file provides an installation guide, steps for model training, a user guide for making predictions, and notes on visualization and metrics.  
  
- bert\_train.py: A Python script that contain functions and classes for training a BERT model. It includes imports, data preprocessing, and dataset class definition.  
  
- requirements.txt: A list of Python packages required for the project, specifying the versions of transformers, torch, pandas, scikit-learn, matplotlib, nltk, numpy, tensorboard, and PyQt5.

# Prompt D

## Post-implementation Report

### Project purpose

### The BERT Toxic Comment Classification system apparently goes a long way in adding politeness and safety within online conversations. It reviews and classifies comments that appear on various online platforms, cataloging toxic ones that may limit positive interactions between people. It flags harmful or inappropriate content with advanced natural language processing ability using the BERT model. With this early detection, moderators and algorithms are allowed to act, such as removing content or warning about conduct breaches. What the system's intended to bring is a respectful, inclusive online space making users free from harmful speech. It raises the quality of digital communication and integrity regarding online community life.

### Datasets

### The dataset is vital to the BERT Toxic Comment Classification system working out since it plays a key role both in the training as well as in the validation process carried out on it. It is from the 'Jigsaw Toxic Comment Classification Challenge' on Kaggle, featuring many user comments labeled for toxicity types like severe, obscenity, threats, insults, and identity hate. It reflects the complexities leveled during real online x-user happenings.

### Before being used for model training, the data undergoes vigorous preprocessing. This includes tokenizing it (breaking text into tokens), removal of stop words (common but uninformative words), and lemmatization (simplifying words to their base form). This is done to simplify the text so that the learning will be improved from its semantic value by the BERT model.

### The chosen data structure helps to tune the training, making the text easy for BERT to process. Cleaning and preparation of the dataset, this way, keeping its essentials while taking away all unnecessary best prepares it. This model will accurately identify and sort toxic comments, aiding in creating healthier online spaces.

A screen shot of a computer program

Description automatically generated

A screen shot of a white sheet

Description automatically generated

**Data product code**

### We developed the Toxic Comment Classification system by dividing this development process into well-articulated, modular components with an objective to facilitate future modifications and lower down systemic mistakes. Various components of the system include text processing, model training as well as evaluation which work together to enhance the classification effectiveness obliged for comments.

### The starting point is preprocessing of the text that prepares and cleanses the data for the model. The main techniques used at this stage are tokenization, stopping, and lemmatization used to reduce or transform the text in the most informative form. The code snippet that shows this preprocessing function kind of cleanly delineated steps, shows how raw text is transformed into a processed format altogether suitable for model consumption.

### Following this, the BertToxicModel class essentially implements the model architecture by incorporating the use of a transformer library's pre-trained BERT model to take advantage of cutting-edge natural language processing techniques. The training process including data flow through the system is driven in the train\_and\_evaluate functions. This function harmonizes the training epochs, processes batches of data and issues backward and optimization steps as displayed in the snippet provided.

### On each training epoch, loss and accuracy metrics are computed and then logged out for purposes of tracking and evaluation. The model after training should be able to input comments and classify their toxicity as well as being evaluated with performance metrics. This pipeline makes it possible to perform continuous fine-tuning of the model, in which at every step, the preprocessing, the BertToxicModel, and the train\_and\_evaluate function are all inter-dependent to make the overall accuracy and specificity of the toxicity comment classifier better.

A computer screen shot of text

Description automatically generated

A screen shot of a computer program

Description automatically generated

### Hypothesis verification

### The Toxic Comment Classification hypothesis stated that a BERT-based model would perform with optimum efficiency and accuracy in the detection and classification of toxic comments provided in a corpus for digital communication. This is checked in efficiency of the scrutinized model through accuracy, precision, recall as well as F1 score – all standard well-established measuring metrics. Computation of these metrics was thoroughly scrutinized for both training and validation steps. Performance was encouraging, and with the BERT-based model having a good accuracy and balanced precision-recall trade-off looking at the F1 score. But the complexity of human language and the subtlety of toxic behavior means that not only is the initial data supporting it, but also that such a hypothesis does not serve as definitive confirmation. This seems to indicate that better refinement of the model, which can be done through a more representative training dataset or using improved algorithms for linguistic processing, may yet confirm this hypothesis with higher certainty. These kinds of findings thus make one believe that the BERT-based model is positively moving in the right direction, with significant promise reflected within automated toxic comments classification.

### Effective visualizations and reporting

### The Toxic Comment Classification application uses bar chart visualizations to effectively showcase the model's ability to predict various categories of toxicity. These graphs are created using the add\_plot method in the App class of the predict.py code. They display the rate of comments predicted to belong to each toxic category. The dashboard concisely presents these visualizations, highlighting the proportion of comments categorized as insults, threats, or other toxic behaviors. This visual breakdown not only affirms the model's capacity to identify and categorize toxic comments but also provides an intuitive and quick insight into how the model predicts different toxic traits. As a result, the bar charts clearly depict the model's classification landscape. Such visual reporting is essential for evaluating the model's prediction distribution and identifying possible biases or areas for improvement in the model's classification, ensuring more balanced and accurate outcomes in real-world applications.

### Accuracy analysis

### The Toxic Comment Classification model was evaluated with a strong suite of metrics from precision, recall, and F1 score to training and validation loss. The model was then trained intensively where the loss during training was observed and made to decrease as the ability of the classifier learned to identify toxic comments increased. Precision, recall, and F1 scores were noted to give a wholesome picture of model's predictive accuracy centered around recall of actual positives (recall) and the accuracy of positive identifications (precision). Using F1 scoring is very crucial in cases where avoiding false positives and negatives is very important, acting as a balance between precision and recall. The model performed relatively well in validation due to the low value of the validation loss which indicated good generalization. In this phase, metrics were monitored through TensorBoard which offered a view of the model's progress quite in detail. Although the F1 score for the model was high, ensuring strong prediction skills, the validation loss indicated possible overfitting aspects, help of which more room for improvement can be tapped upon. This deep insight was achieved because of the thorough tracking of these metrics through TensorBoard, setting in stage the refinement of the model towards top-performing toxic comment classification systems.

A screenshot of a graph

Description automatically generated

A computer screen shot of a program code

Description automatically generated

### Application testing

During the development of the application, a comprehensive and systematic testing methodology that focused on individual components and the integrated system was applied for the Toxic Comment Classification. The team conducted extensive manual and automated tests, primarily to evaluate the core functionality related to the accurate classification of user-input comments. Special attention was given to the user interface, particularly in managing the App class, to ensure it was both intuitive and responsive. This included facilitating the input of different types of comments and effectively displaying classification results. Each module, with a specific focus on comment classification, underwent individual testing using various test cases and sample comments to ensure functionality and accuracy. After module-level testing, the application's overall functioning was assessed to check the seamless integration of all components. The iterative testing process led to continuous adjustments based on user feedback, notably refining the classification algorithms, and enhancing the user interface. As a result of this elaborate testing regime, the application emerged as highly functional, providing accurate comment classification and a user-friendly interface.

A screenshot of a computer screen

Description automatically generated

Appendices

**IMPORTANT**

**The prediction cell is also the last cell inside the same file “bertnlp\_model.ipynb” for simplicity sake to see everything in on file.**

**Since the original trained model is too big to download through WGU, within the first cells, the file will be downloaded here (Make sure to use Google Colab) or the file will have to be downloaded from the link manually:**

**A screenshot of a computer

Description automatically generated**

**If you use anything other than Google Colab, you’ll have to use this cell and follow the link and manually download since gdown is down at the momentA screenshot of a computer

Description automatically generated**

***All the files needed to run this application are in the compressed folder “Jenkins\_Antonio\_BERTNLP\_Capstone”***

# Github: <https://github.com/tonioreign/Jenkins_Antonio_BERT_Capstone>

**Installation Guide**

**Prerequisites**

- Python 3.7 or higher

- Jupyter Notebook or JupyterLab

- Supporting libraries as specified in requirements.txt

**Steps**

1. Ensure Python 3.7 or higher is installed.
2. Install Jupyter Notebook or JupyterLab if not already installed:
3. Install required Python libraries using pip and requirements.txt:
4. Open the bertnlp\_model.ipynb notebook in Jupyter Notebook or JupyterLab.

**Model Training**

Place the training data in the specified directory, typically /content/drive/MyDrive/train.csv.

Execute the cells in bertnlp\_model.ipynb sequentially to train the model.

The trained model will be saved as bert\_toxic\_model.pth.

**User Guide for Toxicity Prediction and Model Training with BERT**

Welcome to the user guide for making predictions with a pre-trained BERT model and training the model using the bertnlp\_model.ipynb notebook. This guide will walk you through the process of using the provided tools and understanding the functionality of TensorBoard for visualization.

**Setting Up the Program:**

1. Clone the repository or download the source code to your local machine.
2. Navigate to the project directory in your terminal or command prompt.
3. Install the required dependencies by running `pip install -r requirements.txt`.

Training the Model with TensorBoard Visualization:

1. Launch Jupyter Notebook or JupyterLab and open the bertnlp\_model.ipynb file.
2. Follow the instructions within the notebook to train the model on your dataset.
   1. Ensure that TensorBoard is running by executing `%load\_ext tensorboard` and `%tensorboard --logdir runs` in separate cells.
   2. As the model trains, TensorBoard will display real-time metrics such as loss, accuracy, precision, recall, and F1 score.
   3. Use the TensorBoard interface to monitor the model's performance and to analyze the training process.
3. After training, the model will be saved as bert\_toxic\_model.pth in the specified output directory.
   1. The TensorBoard logs will also be saved, allowing for post-training analysis.

**Using the GUI for Prediction and Monitoring Training Metrics:**

**IMPORTANT**

**The prediction cell is also the last cell inside the same file “bertnlp\_model.ipynb” for simplicity sake to see everything in on file.**

**Since the original trained model is too big to download through WGU, within the first cells, the file will be downloaded here (Make sure to use Google Colab) or the file will have to be downloaded from the link manually**

**A screenshot of a computer

Description automatically generated**

**If you use anything other than Google Colab, you’ll have to use this cell and follow the link and manually download since gdown is down at the moment.**

**A screenshot of a computer

Description automatically generated**

1. Ensure that the trained model bert\_toxic\_model.pth is in the correct directory as specified in predict.py.
2. Run predict.py to start the graphical user interface (GUI).
3. Enter the text you wish to analyze in the provided text field within the GUI.
4. Click the 'Predict' button to get the toxicity predictions for the entered text.
5. The GUI will display the prediction results, indicating the likelihood of each type of toxicity.
6. To review the model's performance metrics such as F1 score, precision, and recall over the training period,
7. click the 'Show TensorBoard Metrics' button in the GUI.
8. TensorBoard will provide visualizations of these metrics, allowing for in-depth analysis of the model's training process.

**Summation of Learning Experience**

Working with Natural Language Processing, I tackled the complex task of understanding and implementing BERT (Bidirectional Encoder Representations from Transformers). The intricate details of BERT, like attention and context-aware embeddings, were challenging at first. My goal to improve my NLP skills and the need to process large text volumes kept me going. I learned the theory behind BERT and applied it in a text classification project. The project required extensive hyperparameter tuning and model fine-tuning, which sharpened my problem-solving and coding skills. Even with a difficult learning curve, I managed to enhance the accuracy of my models. This self-driven project, with occasional guidance from my mentor, has boosted my confidence in AI and motivated me to deepen my machine learning knowledge.

## References

Jigsaw Toxic Comment Classification Challenge. (n.d.). Kaggle. <https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge>

Pham, K. (n.d.). Text Classification with BERT. Medium. <https://medium.com/@khang.pham.exxact/text-classification-with-bert-7afaacc5e49b>