Curriculum for
Bachelor of Computer Engineering Technology

Contents

.... 1

Chapter 01: Background of Training Organization 4

Organization Overview 4

Training Context 4

Chapter 02: Schedule of Training and Duties as Trainee 5

Weekly Training Summary 5

Chapter 03: Working Experience 9

Projects Carried Out 9

Key Learnings: 13

Hands-On Skills Acquired 14

Problems and Challenges Encountered 17

Problem-Solving Process/Approach 19

Supervisory Tasks 20

Suggestions for Enhancing Productivity 21

Quality Management Systems in Place 22

Safety Features at Workplace 24

Lessons Learned for Future Work 24

Chapter 04: Conclusion 25

References 27

Appendices 27

Supervised Industrial Training Report

Declaration Page

I hereby declare that this report is my original work and reflects my supervised industrial training experience. It has been prepared in compliance with the guidelines provided by my institution. All the information presented is accurate to the best of my knowledge.

Signature of Student:

Signature of Trainee Supervisor:

Chapter 01: Background of Training Organization

Organization Overview

MaxRemind Inc., as one of the leading providers of medical billing services, is not just another name in the healthcare industry. It is the result of 20 years of extremely hard work and dedication. With a bright vision & clear focus in mind, we embark on a meticulous journey of forging innovation & convenience in the medical industry.

It all started with the collaboration of two extremely brilliant & visionary minds who wanted to change medical care for everyone. Be it medical practitioners, patients, or even hospital IT experts – the idea was to bring an amazing experience for all.

Their driving force came from a rough experience where they first-handedly saw all the ill practices and negligence that were being carried out by different practices. They decided to change the industry once and for all.

Training Context

The purpose of the training program was to enhance technical skills in software development, focusing on both backend and frontend technologies. The program was designed to provide hands-on experience in building and deploying web and mobile applications, integrating APIs, managing databases, and working with real-time communication protocols. By involving the development of various projects, including Django API integration, React Native app development, real-time chat applications, and machine learning-based solutions, the training aimed to improve proficiency in key areas such as backend development, API design, security protocols, and user interface design.

The training program aligns with the organization's operations by addressing the need for skilled developers capable of delivering end-to-end solutions. The knowledge gained during the training will support the organization in building more efficient, scalable, and user-friendly applications, directly contributing to the organization's growth and technical capabilities. Additionally, the program prepares the team to work on advanced technologies like AI and real-time communication, which are crucial for staying competitive in the rapidly evolving tech landscape.

Chapter 02: Schedule of Training and Duties as Trainee

Weekly Training Summary

Week 1: Django setup, API development, React Native environment setup, and CRUD operations.

- Gained an understanding of Django's file structure and its integration with SQL Server.
- Developed APIs (PUT, DELETE, GET, POST) in Django.
- Set up the React Native environment and assisted team members with debugging.
- Created a basic React Native app to retrieve data via APIs.
- Applied CRUD operations in a React Native app and resolved API data-fetching errors.

Week 2: Advanced API development and debugging.

- Created APIs for integrating ChatGPT Vision and Text in Django.
- Debugged and resolved errors in APIs, improving functionality.
- Built and implemented a React.js project with CRUD operations.

Week 3: Image processing and user authentication.

- Implemented image upload and viewing features in React Native apps.
- Integrated JWT authentication for securing API endpoints.
- Developed APIs and completed full-stack integration for Books, Library, and Members entities.

Week 4: React Native data fetching and Excel file formatting.

- Explored and utilized useQuery for efficient data fetching in React Native.
- Developed and debugged an API for formatting Excel files in Django.

Week 5: Excel formatting API development.

- Progressed on creating and refining an API for Excel data extraction and formatting.
- Addressed specific validation issues for data accuracy.

Week 6: Enhancing Excel formatting API and user authentication.

- Resolved formatting issues and supported multiple Excel formats.
- Developed user authentication API endpoints for signup and login.

Week 7: Frontend-backend integration for authentication.

- Designed authentication screens in React Native for signup, signin, and password recovery.
- Successfully integrated frontend and backend for seamless user interaction.

Week 8: React.js development and GPT-powered Python code generation.

- Built a React.js web page for Excel formatting and integrated it with the Django API.
- Collected and cleaned Python code data from GitHub for training GPT.

Week 9: Python Code Generation and Real-Time Chat Initiation

- Preprocessed and tokenized a dataset of 12,805 Python files for GPT-based model training.

- Initiated work on a Django WebSocket-based real-time chat application, resolving initial database chat history issues.

Week 10: OCR API QA and Real-Time Chat Progress

- Conducted rigorous testing of the OCR API for PDF and image data extraction.
- Progressed with WebSocket functionality for real-time chat.

Week 11: Chat App API Development and Secure Authentication

- Developed WebSocket APIs for one-on-one chat in Django.
- Implemented secure sign-in, sign-up, room creation, and file upload features.

Week 12: Mobile App Design and Theming

- Set up navigation patterns (drawer, bottom, and stack) using React Native Paper.
- Implemented dynamic light/dark themes.

Week 13: Debugging and Attendance Module Development

- Resolved errors during frontend-backend integration.
- Developed APIs for attendance management and designed search-enabled screens.

Week 14: Standalone App Debugging and Code Optimization

- Resolved build errors for a React Native standalone application.
- Optimized the frontend architecture of the real-time chat app.

Week 15: Media Sharing and AI Chatbot Integration

- Implemented file, image, and video sharing in the chat app.
- Began integrating the Llama 3.2 model for real-time chatbot functionality.

Week 16: Dataset Preparation and Final Testing

- Cleaned, preprocessed, and finalized datasets for fine-tuning the Llama 3.2 model.
- Conducted UI testing and resolved integration issues.

Key Takeaways

- Advanced skills in real-time communication, backend development, API integration, and debugging.
- Hands-on experience in machine learning dataset preparation and model fine-tuning.
- Improved understanding of mobile app design, theming, and cross-platform compatibility.
- Strengthened problem-solving abilities by tackling integration challenges and optimizing codebases.

Chapter 03: Working Experience

Projects Carried Out

1. Django API Development and Integration with React Native

- **Objective:** Build RESTful APIs to handle CRUD operations, user authentication, and data management, integrated into a React Native app for a seamless user experience.
- **Key Achievements:**
  - Created APIs with Django for CRUD operations and JWT-based authentication.
  - Connected APIs to a SQL Server database and integrated them with a React Native frontend.
  - Enhanced backend and frontend communication proficiency.
- **Tools:** Django, React Native, SQL Server, JWT Authentication
- **Outcome:** Strengthened skills in backend development and API integration.

2. React Native App Development

- **Objective:** Develop a React Native application to interact with Django APIs for CRUD operations and additional functionalities.
- **Key Achievements:**
  - Built a fully functional React Native app, including data fetching and display.
  - Added features like image processing and JWT-based authentication.
- **Tools:** React Native, Django APIs

- **Outcome:** Improved React Native development skills, focusing on UI/UX and backend connectivity.

3. Excel Formatter API Development

- **Objective:** Develop a Django-based API to process and format data from Excel files.
- **Key Achievements:**
    - Designed and implemented an API for Excel data extraction and formatting.
    - Supported multiple Excel formats and resolved formatting issues with logic improvements.
- **Tools:** Django, OpenPyXL, Pandas
- **Outcome:** Enhanced proficiency in handling and processing Excel data programmatically.

4. React.js Web Application for Excel Formatting

- **Objective:** Create a React.js frontend to interact with the Excel Formatter API.
- **Key Achievements:**
    - Developed a user-friendly interface for uploading and processing Excel files.
    - Streamlined document formatting tasks with API integration.
- **Tools:** React.js, Django APIs
- **Outcome:** Improved web development skills with a focus on frontend-backend integration.

5. User Authentication System

- **Objective:** Build a secure authentication system with JWT-based APIs.
- **Key Achievements:**
  - Created signup, login, and password recovery endpoints in Django.
  - Designed authentication screens in React Native and integrated them with the backend.
- **Tools:** Django, React Native, JWT
- **Outcome:** Gained expertise in designing and implementing secure authentication systems.

6. GPT-Powered Python Code Generation

- **Objective:** Train GPT models to generate Python code using a custom dataset.
- **Key Achievements:**
  - Preprocessed a dataset of 12,805 Python files and trained a GPT model.
  - Analyzed results, identifying the need for larger datasets for better functionality.
- **Tools:** Python, GPT API
- **Outcome:** Acquired foundational knowledge in machine learning and GPT-based code generation.

7. Real-Time Chat Application Development

- **Objective:** Build a chat application with real-time messaging, file sharing, and AI chatbot integration.
- **Key Achievements:**
  - Implemented WebSocket APIs in Django for real-time communication.

    - Added features like file sharing, theme switching, and AI chatbot integration with Llama 3.2.
    - Developed and cleaned datasets to fine-tune the chatbot's responses.
- **Tools:** Django, React Native, WebSockets, LangChain, Llama 3.2
- **Outcome:** Strengthened skills in real-time communication, WebSocket debugging, and AI chatbot integration.

8. Attendance Management Module (Company Task)

- **Objective:** Create an API and mobile interface for real-time attendance tracking.
- **Key Achievements:**
    - Developed APIs for attendance marking and member management.
    - Built a React Native interface for displaying employee data with search functionality.
- **Tools:** Node.js, React Native
- **Outcome:** Improved backend API design and mobile UI development skills.

9. React Native Standalone Application

- **Objective:** Build and deploy a standalone React Native application.
- **Key Achievements:**
    - Resolved build errors and successfully deployed the app.
- **Tools:** React Native, Node.js
- **Outcome:** Gained expertise in troubleshooting and deploying mobile applications.

10. OCR API Quality Assurance (Company Task)

- **Objective:** Test and enhance an OCR API for extracting data from medical billing documents.
- **Key Achievements:**
    - Identified edge cases and suggested improvements for accuracy and reliability.
- **Tools:** OCR API, PDFs, Images
- **Outcome:** Improved API testing and quality assurance skills.

11. AI Model Fine-Tuning for Chatbot

- **Objective:** Fine-tune the Llama 3.2 model for integration into a chat application.
- **Key Achievements:**
    - Prepared datasets for fine-tuning the AI model.
    - Integrated the chatbot into the application for personalized communication.
- **Tools:** Llama 3.2, Python, LangChain
- **Outcome:** Enhanced understanding of fine-tuning AI models for practical applications.

Key Learnings:

- Mastered **Django** and its integration with frontend frameworks like React.js and React Native.
- Developed proficiency in **React Native** for building user-friendly mobile applications.
- Gained foundational experience in **machine learning** and **GPT model training**.
- Strengthened debugging skills and real-time communication expertise using **WebSockets**.
- Enhanced problem-solving abilities by addressing complex integration issues.

Hands-On Skills Acquired

Backend Development

- Proficient in building RESTful APIs with Django for CRUD operations and authentication.
- Hands-on experience in securing APIs using **JWT Authentication**.
- Expertise in **SQL Server** integration and database management.
- Skilled in handling file operations, including data extraction and processing from Excel files using libraries like **OpenPyXL** and **Pandas**.
- Real-time communication using **WebSockets** for chat applications and real-time notifications.

Frontend Development

- Skilled in developing mobile applications using **React Native**, with features such as:
  - CRUD operations.
  - Dynamic theming (light/dark mode).
  - File upload and viewing.
- Experienced in building web interfaces using **React.js**, focusing on API integration and data visualization.

- Proficient in implementing intuitive UI/UX designs with **React Native Paper** and responsive web frameworks.

Authentication and Security

- Expertise in designing and implementing secure user authentication systems with **JWT-based security**.
- Developed APIs for signup, login, and password recovery.
- Integrated authentication systems seamlessly with frontend applications.

Real-Time Applications

- Built real-time chat applications with Django WebSockets and React Native, incorporating:
    - Messaging, file, and media sharing.
    - AI chatbot integration (Llama 3.2).
    - Real-time notifications.

Data Processing and Automation

- Extracted, cleaned, and processed datasets from multiple sources like Excel and PDFs.
- Automated data formatting and validation tasks for organizational needs.
- Hands-on experience in working with OCR APIs for text extraction and analysis.

AI and Machine Learning

- Experience in preparing datasets for AI model training and fine-tuning.

- Fine-tuned the **Llama 3.2** model for chatbot integration, improving contextual responses.
- Explored GPT-based Python code generation by training models on custom datasets.

Mobile Application Development

- Set up React Native environments, resolved build errors, and successfully deployed mobile applications.
- Designed and implemented navigation patterns (stack, drawer, and bottom navigation).
- Enhanced apps with dynamic theme switching and cross-platform compatibility.

Debugging and Problem-Solving

- Resolved complex integration issues between frontend and backend systems.
- Debugged API connectivity, build errors, and performance issues in both web and mobile applications.

Team Collaboration

- Assisted team members in setting up development environments and resolving setup issues.
- Collaborated on projects requiring backend, frontend, and AI model integration.

Tools and Technologies

- **Languages:** Python, JavaScript
- **Frameworks/Libraries:** Django, React Native, React.js
- **Databases:** SQL Server
- **APIs:** WebSocket, JWT, OCR APIs
- **AI/ML:** GPT API, Llama 3.2, LangChain, ChromaDB

- **Utilities:** OpenPyXL, Pandas
- **Version Control:** GitHub

These skills collectively highlight strong technical proficiency in full-stack development, real-time communication, and AI-driven applications

Problems and Challenges Encountered

Backend Challenges

1. **API Performance Bottlenecks**:
   - **Problem**: APIs faced performance issues while handling large datasets during data extraction and formatting.
   - **Solution**: Optimized database queries, used bulk operations, and implemented caching mechanisms.
2. **Data Validation Complexities**:
   - **Problem**: Validating and processing inconsistent Excel data with varying formats.
   - **Solution**: Developed dynamic logic using **Pandas** to handle multiple formats and implemented automated data validation checks.
3. **Real-Time Communication**:
   - **Problem**: Establishing and maintaining WebSocket connections for real-time messaging in Django.
   - **Solution**: Implemented Django Channels for WebSocket management, tested thoroughly, and optimized connection stability.

Frontend Challenges

4. **Debugging React Native Setup**:

   - **Problem**: Faced build errors during the setup of React Native environments.
   - **Solution**: Researched error logs, identified dependency conflicts, and collaborated with team members to resolve issues.

5. **Dynamic Theme Management**:
   - **Problem**: Adapting light/dark themes based on mobile system settings.
   - **Solution**: Used **React Native Paper** and context-based state management to implement dynamic theme switching.

6. **Responsive UI Design**:
   - **Problem**: Ensuring consistent UI performance across devices with different screen sizes.
   - **Solution**: Adopted responsive design techniques and thoroughly tested the UI on various devices.

AI and Machine Learning Challenges

7. **Limited Dataset for GPT Training**:
   - **Problem**: Training a GPT model on a small dataset led to suboptimal results in code generation.
   - **Solution**: Preprocessed and tokenized the dataset to maximize its utility, while identifying the need for larger datasets for future improvement.

8. **AI Chatbot Fine-Tuning**:
   - **Problem**: Llama 3.2's initial responses were generic and lacked context-awareness.

    - **Solution**: Fine-tuned the model using custom datasets extracted and cleaned from domain-specific PDFs.

Integration and Deployment Challenges

9. **Frontend-Backend Synchronization**:
    - **Problem**: Encountered API response delays and mismatches during frontend-backend integration.
    - **Solution**: Debugged API calls, streamlined response formats, and implemented asynchronous data handling.

10. **Real-Time Notifications**:

- **Problem**: Ensuring reliable real-time notifications with WebSockets.
- **Solution**: Integrated WebSocket-based push notifications and tested extensively for edge cases.

11. **Mobile Deployment Issues**:

- **Problem**: Deployment of React Native apps failed due to environment misconfigurations.
- **Solution**: Reviewed build configurations, updated dependencies, and debugged Gradle and Xcode errors.

Problem-Solving Process/Approach

1. **Understanding the Problem**:
    - Analyzed the root cause through debugging tools, logs, and testing scenarios.

- Researched similar issues in official documentation, forums, and developer communities.

2. **Collaborative Problem-Solving**:
   - Engaged with team members for brainstorming and knowledge-sharing.
   - Adopted version control practices (GitHub) to track and review code changes during troubleshooting.
3. **Iterative Development**:
   - Broke down complex problems into smaller, manageable tasks.
   - Tested solutions incrementally to identify and resolve errors quickly.
4. **Optimization and Refinement**:
   - Focused on improving performance by optimizing code, database queries, and server configurations.
   - Incorporated feedback from testing phases to refine the final solution.
5. **Learning from Failures**:
   - Documented lessons learned from unresolved issues or partial successes for future reference.
   - Improved error handling mechanisms to prevent recurring problems.

Supervisory Tasks

During the training period, I actively collaborated with my supervisor and team members to ensure the successful execution of various tasks. My supervisory tasks included:

- **API Design and Debugging**: I assisted the team in designing APIs for multiple features and ensured smooth integration with both frontend and backend systems. I also helped debug API-related issues to improve functionality and optimize performance.
- **Code Reviews**: I participated in reviewing team members' code, providing feedback for improvements, and ensuring adherence to coding best practices.
- **Assistance with Development Environments**: I helped set up development environments for team members, resolved setup issues, and ensured that all developers were using consistent tools and versions.
- **Collaboration on Problem-Solving**: I worked closely with my supervisor to address challenges, particularly in areas related to integrating APIs with the frontend, optimizing code for performance, and implementing new features for real-time communication.

Suggestions for Enhancing Productivity

- **Use Larger Datasets for Training AI Models**: During the process of training AI models, particularly the GPT-based Python code generation, I noticed that the results could be improved with larger and more diverse datasets. Expanding the training dataset would increase the model's ability to generate more accurate and context-aware outputs.
- **Optimize Chat Application Architecture for Scalability**: As the real-time chat application expands, optimizing its architecture to ensure scalability is crucial. Implementing load balancing, data partitioning, and enhancing WebSocket management would help the application handle an increased number of users without performance degradation.

- **Automate Testing and Continuous Integration**: Introducing automated testing frameworks and continuous integration tools would significantly reduce the time spent on manual testing and integration tasks. This would help catch bugs earlier and allow developers to focus on new features and improvements.

- **Code Modularization and Reusability**: To enhance maintainability, breaking down complex code into smaller, reusable modules would help in reducing duplication and make future updates more efficient. Encouraging developers to follow modular design patterns could streamline the development process and improve collaboration.

- **Improved Communication and Knowledge Sharing**: Although teamwork was strong during the training, fostering an even more robust knowledge-sharing culture could further enhance productivity. Regular team meetings, documentations, and shared repositories of solutions to common problems would speed up issue resolution and reduce redundant efforts.

Quality Management Systems in Place

During the industrial training period, several quality management systems (QMS) were implemented to ensure the effectiveness and reliability of the work produced. These systems were designed to streamline processes, improve performance, and ensure high standards of quality across all projects. Key quality management practices included:

- **Regular QA Testing**:
    - QA testing was consistently conducted for critical components of the systems, especially for APIs and AI-driven features. This included rigorous testing of the OCR API for PDF and image data extraction, as well as the real-time chat application and GPT-powered Python code generation.
    - Each release or update to the system underwent extensive QA testing to identify bugs, performance issues, or security vulnerabilities. Feedback from the testing phase was promptly incorporated into the development process to ensure that the final product met the required standards.

- **Code Reviews and Peer Reviews**:
    - Regular code reviews were conducted to ensure that all development followed best practices in terms of coding style, optimization, and security. This helped maintain consistency and encouraged the sharing of knowledge and improvement of skills among team members.
    - Peer reviews not only helped catch issues early but also fostered a collaborative environment where everyone contributed to the quality of the codebase.
- **Documentation of Issues and Resolutions**:
    - A structured approach was adopted to document issues encountered during the development process. Each bug or challenge was logged, along with its root cause, proposed solutions, and resolution steps.
    - This practice ensured that similar issues could be quickly addressed in the future, reducing repetitive troubleshooting efforts and improving long-term efficiency.
- **Version Control with GitHub**:
    - GitHub was used for version control, ensuring that all code changes were tracked and that multiple developers could collaborate on the same codebase without conflict. By maintaining a well-structured Git workflow, issues like merge conflicts and accidental overwrites were minimized.
    - Version control allowed for systematic release management, helping ensure that the most stable versions of the application were deployed.
- **Performance Monitoring and Optimizations**:
    - Key performance indicators (KPIs) were monitored throughout the development process, especially for APIs and real-time applications. Tools for profiling and performance benchmarking helped identify bottlenecks and optimize both frontend and backend components.
    - Load testing and stress testing were also conducted to verify the scalability of applications, especially with regard to the real-time chat feature and the handling of large datasets during API processing.
- **Security Protocols**:

- Security measures, particularly for APIs, were carefully planned and implemented. This included the use of JWT authentication for securing endpoints, implementing encryption for sensitive data, and ensuring that the APIs were protected against common vulnerabilities.
- In addition, guidelines for secure coding practices were followed, and security audits were performed regularly to assess the integrity of the systems in place.

These quality management systems collectively ensured that the work produced was reliable, secure, and high-performance, while also fostering a continuous improvement mindset throughout the training process.

Safety Features at Workplace

MaxRemind guarantees a completely safe work environment by focusing on representative prosperity and hardware security. The office is observed all day, every day with reconnaissance cameras and a committed security monitor. Network and electrical links are safely introduced to lessen dangers, and fundamental security hardware like emergency treatment packs is promptly accessible.

Moreover, fire security measures, including dousers and crisis departure plans, are set up to address unanticipated episodes. The association likewise underscores IT foundation security, giving representatives a protected and solid climate to work gainfully and with certainty. Standard security bores and preparing further build up the organization's obligation to work environment wellbeing.

The company commitment to creating a safe environment stretched out to protecting its IT infrastructure, allowing employees to work confidently and productively.

Lessons Learned for Future Work

1. **Punctuality and Time Management**

- Adhering to schedules and being on time are essential for maintaining professionalism and meeting deadlines.

2. **Professional Conduct During Work Hours**
   - Avoid engaging in unrelated activities, such as personal conversations or games, during working hours to ensure productivity and focus.
3. **Maintaining Professional Behavior**
   - While communicating or collaborating, ensure that discussions are conducted in a professional setting, avoiding casual or distracting behaviors.
4. **Commitment and Attendance**
   - Minimize unnecessary leaves to maintain consistency and reliability in delivering tasks and meeting objectives.
5. **Engagement During Seminars and Meetings**
   - Stay attentive and actively participate in discussions during seminars and meetings to enhance learning and contribute effectively.

Chapter 04: Conclusion

The training period has been a pivotal chapter in my journey toward becoming a well-rounded software developer. It provided not only technical skills but also a deep understanding of how to approach real-world problems, collaborate effectively within teams, and maintain professionalism in a corporate environment. The hands-on experience with various technologies, tools, and frameworks, such as Django, React Native, WebSockets, JWT Authentication, and GPT models, has significantly improved my ability to design and implement complex systems, ensuring seamless integration between backend and frontend components.

Through the development of real-time applications, user authentication systems, AI-powered tools, and Excel data processing APIs, I have strengthened my problem-solving and debugging abilities. These projects helped me refine my coding practices and taught me how to manage large datasets, create user-friendly interfaces, and ensure robust and secure backend architectures.

However, the technical skills were only part of the overall learning experience. The importance of teamwork, clear communication, and adherence to professional ethics became evident as I navigated through group projects and company tasks. The challenges of balancing time between individual tasks and collaboration underscored the value of time management, effective communication, and the ability to adapt to dynamic work environments.

Additionally, this training reinforced the need for continuous learning and improvement, especially in an industry that evolves rapidly. While I have gained proficiency in several areas, I recognize that software development is an ongoing learning process, and staying updated with the latest advancements is key to remaining competitive and innovative.

From a professional standpoint, I have learned valuable lessons in workplace behavior, such as maintaining punctuality, staying focused during working hours, and understanding the significance of active listening during seminars and meetings. These behavioral aspects are essential for fostering a positive and productive work environment, where both personal growth and organizational goals can thrive.

In conclusion, this training has laid a solid foundation for my career in software development. The technical knowledge, problem-solving skills, and professional attitude I have developed will be the guiding principles as I move forward in my career. With a commitment to continuous learning, improving my soft skills, and embracing new challenges, I am confident that I am prepared to make meaningful contributions in any future professional setting

References

Django Documentation. (n.d.). Django Project. Retrieved from https://www.djangoproject.com/

React Native Documentation. (n.d.). Facebook. Retrieved from https://reactnative.dev/

JWT Authentication. (2024). JWT.io. Retrieved from https://jwt.io/

Openpyxl Documentation. (2024). openpyxl. Retrieved from https://openpyxl.readthedocs.io/

Pandas Documentation. (2024). pandas. Retrieved from https://pandas.pydata.org/

WebSockets in Django. (2024). Django Channels. Retrieved from https://channels.readthedocs.io/

Llama 3.2 Model Documentation. (2024). Meta. Retrieved from https://ai.meta.com/

Langchain Documentation. (2024). Langchain. Retrieved from https://www.langchain.com/

GitHub Repository for ReactNative-Django Integration. (2024). Retrieved from https://github.com/your-username/ReactNative-Django

GitHub Repository for Excel Formatter Django. (2024). Retrieved from https://github.com/your-username/Excel-Formatter-Django

GPT-3 Documentation. (2024). OpenAI. Retrieved from https://openai.com/

Appendices

Include additional materials such as screenshots, sample code snippets, or datasets prepared during training.