TED UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

CMPE 492

WorkerSecure

Final Report

by

Hasret Türkmen

Asaf Emin Gündüz

Selin Sevcan Çakan

Niyazi Ahmet Metin

Table of Contents

1. Introduction ... 3
- 1.1. Project Overview ... 3
- 1.2. Objectives of the Report ... 3
- 1.3. Scope ... 3

2. System Architecture and Design ... 3
- 2.1. Final Architecture ... 3
- 2.2. Design Details ... 4
- 2.3. Diagrams ... 4

3. Deployment Process ... 5
- 3.1. Detailed Workflow and Process ... 5
- 3.2. Technologies Used ... 5
- 3.3. Modules ... 6
  - Front-end: ... 6
  - Back-end: ... 6
- 3.2. Code Snippets ... 7

4. Testing and Evaluation ... 9
- 4.1. Test Plan Summary ... 9
- 4.2. Test Cases & Results ... 9
- 4.3. Assessment Of the Tests ... 10

5. Impact Analysis ... 10
- 5.1. Global Impact ... 10
- 5.2. Economic Impact ... 11
- 5.3. Societal Impact ... 11

6. Contemporary Issues and Use Of Resources ... 11

7. Current Trends and Future Directions ... 12

8. Use of Resources ... 12

9. Compliance and Ethical Considerations ... 13

10. Deployment Process & Manual ... 14

11. Conclusion ... 15
- 11.1. Summary Of Findings ... 15
- 11.2. Achivements ... 15
- 11.3. Future Works ... 15

12. References ... 15

1. Introduction

1.1. Project Overview

With this report we aimed to explain comprehensive details of our hardhat detection system that is designed for increasing safety cautions in construction sites. System identifies workers that aren't wearing their safety equipments utilizing YOLO-based object detection technology.

1.2. Objectives of the Report

This report mainly provides information about how the hardhat detection system was created and executed, how it performed during the test session, retailers it may affect among others. Correspondingly, it seeks to examine current affairs, ways to deploy it and adhere to ethical principles in question.

1.3. Scope

This report describes the system's architectural design; the technologies and tools used in its development; testing approaches and performance assessment results. It also covers the economic consequences of system implementation, taking into account its environmental and social effects. Besides, this paper addresses various compliance matters and suggests possible future enhancements that could be made to ensure continuity in implementing new features.

2. System Architecture and Design

2.1. Final Architecture

WorkerSecure is a project whose aim is to improve the safety of workers in construction areas. The improvement procedure is to use YOLO model to monitor the environment in real time. The goal of the system is to identify the workers who isn't wearing necessary safety equipment. YOLO model enables the detection process to be completed with image processing. The system will be able to detect the anomalies and notify the OH&S specialist who's in charge of ensuring safety by mail.

Presentation Layer:

This layer is the user interface that OH&S personnel can monitor real-time video feeds and receive the instant notifications. The frontend is developed using React.js for a responsive and user-friendly experience.

Business Logic Layer:

The crucial layer of this project. This layer handles data processing the video feeds, warns the specialist when a violation of rule is detected. It uses YOLO image processing module for detecting them in real-time. The business logic layer is implemented in python, using TensorFlow and Numpy for image processing and machine learning tasks.

Data Management Layer:

We store user data in this layer. WorkerSecure uses user data for login processes and sending information via email. To ensure flexibility and scalability to handle large volumes of unstructured data, MongoDB is used as database. The backend code works with MongoDB to manage data tasks well.

2.2. Design Details

User Interface:

The UI provides live streaming from cameras, allows users to switch between different camera feeds (if there's more than one camera in the area), and offers functionalities such as taking screenshots. The UI also displays notifications for detected safety violations (Adding to SMS and email notifications). Only users (OH&S specialsts/personnels) signed up is allowed to watch the video feed and get notification to ensure privacy and security.

2.3. Diagrams

We created some diagrams to provide a visual representation of the WorkerSecure system architecture, components, and data flow. Following diagrams illustrate the structure and interactions between the User Interface, YOLO Detection Model, and MongoDB Data Management layers.

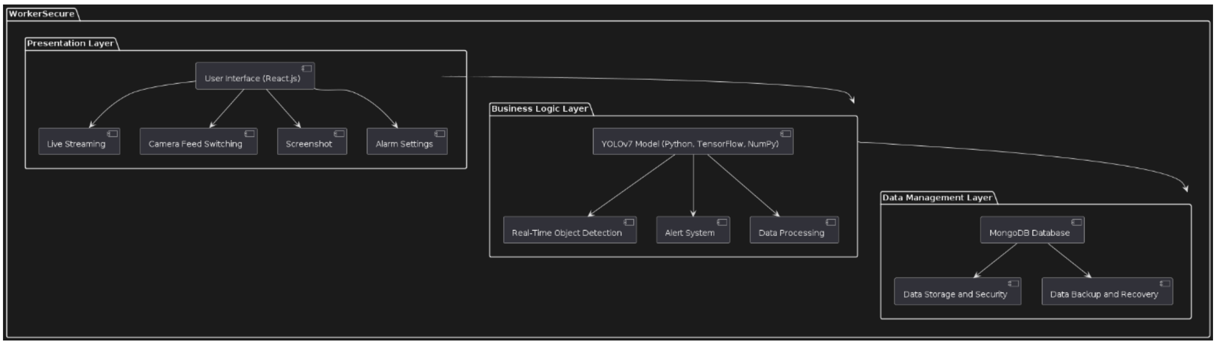

Figure 1: System Architecture Diagram

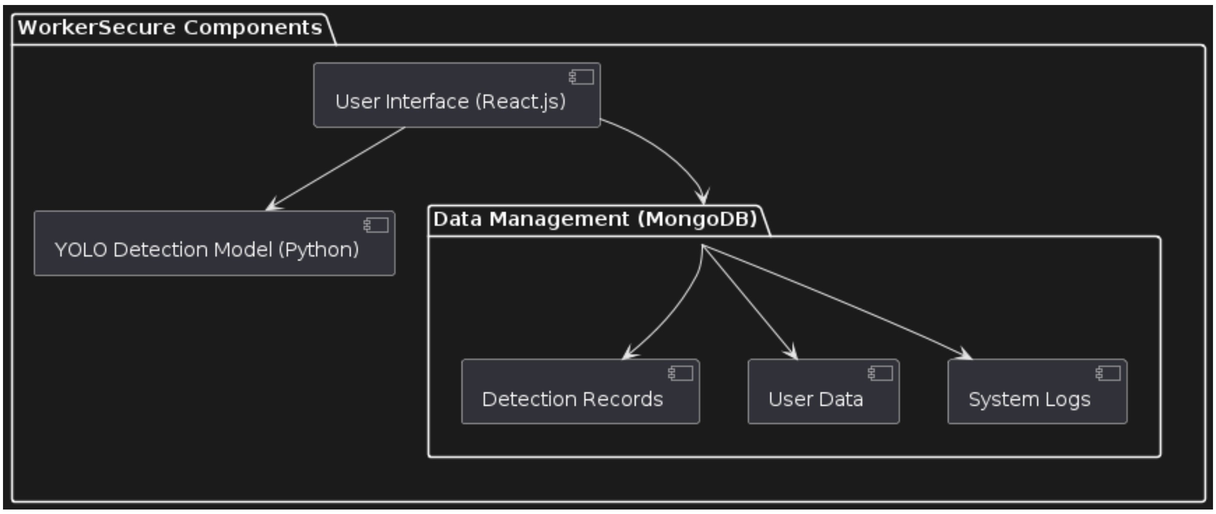

Figure 2: WorkerSecure Components

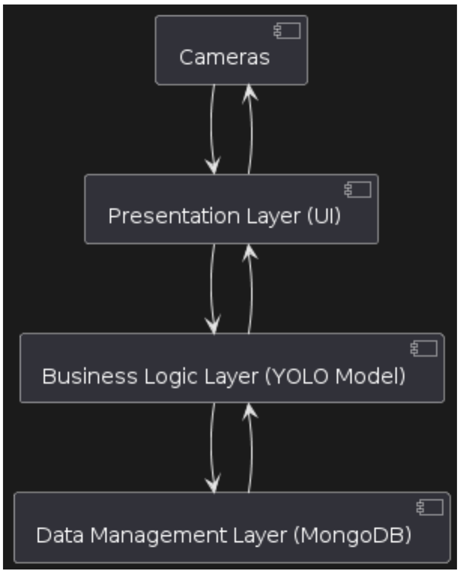

Figure 3: Data Flow Diagram

3. Deployment Process

WorkerSecure is composed mainly of three parts: the user interface, the image processing module, and the alert system. The user interface allows the monitoring of live feeds from cameras and setting of system parameters. The image processing module uses the YOLO architecture for the detection of safety equipment violations. It ensures the alert system alerts personnel whenever there is a breach for immediate action.

3.1. Detailed Workflow and Process

- **Requirement Gathering**: Identification of the safety needs and regulatory requirements of industrial environments.
- **System Design**: Development of a layered architecture with distinct presentation, business logic, and data management layers.
- **Development**: System implementation includes React.js for the frontend, Python for image processing, and MongoDB for data management.
- **Testing**: Carry out extensive testing that would ensure the accuracy and reliability of the system in diverse conditions.
- **Deployment**: The system would be deployed on local machines first, and then it would be integrated with some of the existing monitoring frameworks.

3.2. Technologies Used

- **React.js:** For developing the front-end user interface.
- **Python**: For implementing image processing algorithms.
- **MongoDB**: For managing and storing data securely.
- **Node.js and Express**: For backend server management and API development.

3.3. Modules

Front-end:

CSS Files:

- **About.css**: Styling for the About component.
- **App.css**: Global styling for the application.
- **Developers.css**: Styling for the Developers component.
- **HelmetChecker.css**: Styling for the HelmetChecker component.
- **Home.css**: Styling for the Home component.
- **Navbar.css**: Styling for the Navbar component.
- **Notifications.css**: Styling for the Notifications component.
- **SignUp.css**: Styling for the SignUp component.
- **UserManagement.css**: Styling for the UserManagement component.

JS Files:

- **About.js**: Component for the About page.
- **App.js**: Main application component.
- **Developers.js**: Component for the Developers page.
- **Home.js**: Component for the Home page.
- **Login.js**: Component for the Login page.
- **Navbar.js**: Component for the navigation bar.
- **Notifications.js**: Component for the Notifications page.
- **SignUp.js**: Component for the SignUp page.
- **UserManagement.js**: Component for user management.
- **Video.js**: Component for handling video functionality.
- **Welcome.js**: Component for the welcome page.

Back-end:

Controllers:

- **errorController.js**: Manages error handling and responses.
- **loginController.js**: Handles login-related operations and user authentication.
- **notificationController.js**: Manages notification-related functionality.
- **userController.js**: Handles user-related operations.

Middlewares:

jwtAuth.js: Middleware for handling JWT (JSON Web Token) authentication, ensuring secure access to protected routes.

Models:

- **notification.js**: Defines the schema and model for notifications in the database.
- **user.js**: Defines the schema and model for users in the database.

Routes:

- **index.js**: Main entry point for routing.
- **login.js**: Defines routes related to user login and authentication.
- **notification.js**: Defines routes for notification-related operations.
- **users.js**: Defines routes for user-related operations.

Services**:**

database.js: Manages database connections and configurations.

3.2. Code Snippets

App.js:

```
import React from 'react';
import { Route, BrowserRouter as Router, Routes } from 'react-router-dom';
import '../css/App.css';
import About from './About';
import Developers from './Developers';
import Home from './Home';
import Login from './Login';
import Navbar from './Navbar';
import Notifications from './Notifications';
import SignUp from './SignUp';
import UserManagement from './UserManagement';
import Video from './Video';
import Welcome from './Welcome'; // Welcome bileşenini ekleyin

const App = () => {
    return (
        <Router>
            <div className="app">
                <Navbar />
                <Routes>
                    <Route path="/" element={<Home />} />
                    <Route path="/about" element={<About />} />
                    <Route path="/developers" element={<Developers />} />
                    <Route path="/login" element={<Login />} />
                    <Route path="/user/signup" element={<SignUp />} />
                    <Route path="/video" element={<Video />} />
                    <Route path="/notifications" element={<Notifications />}
/>
                    <Route path="/user-management" element={<UserManagement
/>} />
                    <Route path="/welcome" element={<Welcome />} />
                </Routes>
            </div>
        </Router>
    );
};
export default App;
```

This piece of code is a React component tasked with giving the basic structure to a web application using React Router for navigating between pages. The code imports the necessary libraries and components and applies global styles from 'App.css'. The component App makes use of BrowserRouter, aliased as Router, for client-side routing. Inside the Router, it includes a Navbar component that appears on every page and uses the Routes component to define several Route elements that specify a path and what component should be rendered on that path. This setup then allows users to browse different parts of an application, including Home, About, Developers, Login, SignUp, Video, Notifications, UserManagement, and Welcome pages. Further, the App component exports it so that it can be used elsewhere in the application.

NotificationController.js

```
const Notification = require("../models/notification");
const jwt = require("jsonwebtoken");

exports.getNotifications = async (req, res) => {
    const authHeader  = req.headers.authorization;
    const token = authHeader && authHeader.split(' ')[1];
    const decoded = jwt.verify(token, process.env.ACCESS_TOKEN_SECRET);
    const email = decoded.user.email;
    const page = parseInt(req.query.page, 10) || 1;
    const pageSize = parseInt(req.query.pageSize, 10) || 10;

    try {
        const skip = (page - 1) * pageSize;
        const notifications = await Notification.find({ email: email })
                                                .skip(skip)
                                                .limit(pageSize)
                                                .exec();
        const total = await Notification.countDocuments({ email: email });
        if (!notifications.length) {
            return res.status(404).send({ message: "No notifications found for 
this user." });
        }
        res.status(200).send({
            message: "Notifications retrieved successfully",
            notifications: notifications,
            currentPage: page,
            totalPages: Math.ceil(total / pageSize),
            totalNotifications: total
        });
    } catch (error) {
        console.error("Error retrieving notifications:", error);
        res.status(500).send({ message: "Server error" });
    }
};
exports.postNotification = async (req, res, next) => {
```

```
    Notification.create(req.body)
      .then((Notification) => res.status(200).json(Notification))
      .catch((err) => res.status(500).json({ message: err.message }));
};
```

The code defines two functions: getNotifications and postNotification. The getNotifications function fetches a user's notifications and extracts the authorization header, then verifies it to get the user's email, conducts pagination based on the page and pageSize query parameters, finds the relevant notifications, and responds with a paginated list and pagination details. It returns 404 with a proper message if it doesn't find any notifications and catches any errors that happened during retrieval, logs them, and responds with a status of 500. On the other hand, the postNotification function will create a new notification by getting the data of that notification in its body request, saving it in the database, and answering it with the newly created notification or with an error message in case creation failed.

4. Testing and Evaluation

4.1. Test Plan Summary

The primary goal of the testing phase for our WorkerSecure project was to ensure the accuracy of helmet detection system. We have employed YOLOv7 model optimized for our real-time analysis. Testing was evaluated in a controlled environment. It also simulated various operational conditions. These conditions include different lighting and weather scenarios, to make sure the system is effective in real-world scenarios as well. Our test cases included the helmet detection accuracy, responsiveness of real-time alert system, and adaptability to environmental variations.

4.2. Test Cases & Results

Test ID	Description	Pass/Fail Result	Detail of the Result
TC-HDA-01	Helmet Detection Accuracy	Passed Successfully	The system maintained a precision and recall rate above 95%
TC-HDA-02	Validate Helmet Detection Under Different Lightings	Passed Successfully	Consistent detection across all lighting condisitons
TC-RAS-01	Response Time of The Alert	Passed Successfully	The alert system notified immediately in reaction time
TC-SAD-01	Model Performance in Different Weather Conditions	Passed Successfully*	The system maintained high accuracy under tested conditions
TC-FUC-01	Functionality of UI Controls	Passed Successfully	The system worked without any delay

TC-UIR-01	UI Responsiveness Under Peak Load	Passed Successfully**	The system was able to stay stable under peak load conditions
TC-DHS-01	Data Encryption and Data Transmission Controls	Passed Successfully	The system was able the transmit data securely
TC-DHS-02	Security of the Stored Data Against Unauthorized Access	Passed Successfully	The system blocked unauthorized access successfully

4.3. Assessment Of the Tests

The testing phase of our WorkerSecure system yielded excellent results. The system demonstrated high reliability and robustness in helmet detection accuracy, response time, and adaptability under varied conditions.

*TC-SAD-01 test case passed under the weather conditions faced during the semester. However, it did not involve any snowy wheather. Therefore, after the test cases are provided this could be for the further enhancement of the system.

**Even though all test cases were successfully passed, there were a few minor bugs identified in TC-UIR-01. This could be optimized further.

Our project team will focus on refining these aspects found after the testing phase in future updates.

5. Impact Analysis

5.1. Global Impact

WorkerSecure can significantly help to enhance workplace safety on a global basis. With real-time image processing and machine learning, the system will find application in all kinds of industries located anywhere on the planet. Universal applicability of WorkerSecure will, therefore, suit the application of this system in various countries with their regulations and standards of safety, flexible to improve worker safety.

- **Accidents at Workplaces**: Making sure that workers observe safety protocols is a great way to help WorkerSecure reduce accidents and injuries happening globally and thus make workplaces safer.
- **Standardization of Safety Practices:** The contribution of WorkerSecure would be to standardize safety practices across world industries and ensure one approach towards workplace safety.
- **Increased awareness**: The use of such a system sensitizes on the need to have safety at workplaces and may make other companies invest in such technologies.

5.2. Economic Impact

The economic impacts of WorkerSecure are therefore three-pronged, since it offers benefits to the company, benefits to the economy generally, etc. The more major economic impacts are:

- **Cost savings for organizations**: WorkerSecure can save companies the cost incurred in medicals, legal fees, and compensation claims by preventing accidents and injuries. Besides, it will decrease the downtime and increase productivity by ensuring a safer working environment.
- **Increased Productivity**: A safer work environment can mean higher employee morale and better productivity. The workers will do better once they are assured of their safety and security.
- **Investment in technologies for safety:** Success with WorkerSecure will further drive investment in safety technologies, growing this sector and thereby creating new jobs.

5.3. Societal Impact

The impacts of WorkerSecure go beyond the workplace into communities and larger societies. Other key impacts on society include:

- **Improved Well-being of Workers**: WorkerSecure basically increases the well-being of workers through the creation of an environment where they can work without problems. This is going to improve the mental and physical health of the workers, who feel valued and safer.
- **Ethical and privacy considerations:** WorkerSecure is a system engineered with privacy for workers in mind—all to ensure safety. It has put measures in place to protect personal information and assures an ethical use of surveillance technologies.
- **Benefits to the Community**: The safer workplaces have impacts on the community. When their members are safe at work, then the family and the community benefit; this means that the emotional and financial stresses placed on the families and communities as a consequence of accidents in the workplace are reduced.
- **Promoting a Culture of Safety**: Implementing WorkerSecure in organizations can establish a culture of safety. Such a culture could even diffuse beyond the workplace and encourage safety practices in other aspects of life.

6. Contemporary Issues and Use Of Resources

The development of the WorkerSecure system addresses several issues can be found today’s world. Especially, in the field of workplace safety and surveillance technology. Primary among these is the ongoing challenge of ensuring worker safety in dangerous environments such as construction sites. On the other hand, continous monitoring of the workers also pose an issue about their data privacy. Our

WorkerSecure overcomes these by implementing strict data security measures. WorkerSecure also ensures that the system's design is approvable in terms of ethical standards.

The design and development of WorkerSecure were grounded in basic engineering principles. System design, modularity, and scalability was especially designed according to these principles. Also, the principles ensured that our WorkerSecure stayed consistent to robust engineering standards. By this, the system were allowed for efficient troubleshooting and future enhancements.

7. Current Trends and Future Directions

In engineering, the use of AI and machine learning in real-time monitoring systems is an undeniable trend nowadays. WorkerSecure includes the YOLO model, which is at the forefront of these technologies. It offers fast and accurate object detection capabilities. Thus, this aligns with the current trend mentioned for practical, real-world applications in safety and security sectors.

The further potential for our WorkerSecure may include the integration of IoT (Internet of Things) devices. With AI-based monitoring systems and potential for a more personalized safety measures, this might allow WorkerSecure a promising direction in the sector.Additionally, the future might see the implementation of augmented reality (AR). This might provide real-time safety instructions to workers. Therefore, hopefully a step towards reducing the risk of accidents might be taken.

8. Use of Resources

- **Library Resources**: Throughout our project, several library resources were used for practices. For instance, "Digital Image Processing" by Gonzalez and Woods, provided important insights for each team member. The university's provided digital library resources, including IEEE Xplore and SpringerLink, helped the team to access recent studies and developments in the field of computer vision and AI.

- **Internet Resources**: Internet resources played a crucial role for us to stay updated with the latest advancements to WorkerSecure. Platforms like Kaggle and GitHub provided access to datasets and community insights. These insights helped WorkerSecure team in training and refining our model. Also, websites like Stack Overflow and Medium offered troubleshooting advice and practical implementation tips. Since these tips were coming especially from the global engineering community, we were able to see from a different perspective.

- **Component Resources**: The documentation of YOLOv7 provided detailed component specifications and integration guidance. Additionally, deciding datasheets and technical specifications for hardware, such as NVIDIA GTX 1050TI GPUs, were obtained from manufacturers' websites.

9. Compliance and Ethical Considerations

WorkerSecure is implemented with careful considerations, in terms of data privacy and ethical standards. Therefore, the system is aligned with global General Data Protection Regulation (GDPR). Some key features for achieving this are listed below:

• **Encryption**: WorkerSecure encrypts all data transmitted and stored. This ensures that worker data remains secure, especially from unauthorized access.

• **Access Control**: WorkerSecure employs strict access controls to restrict data access from anauothorized users. Thus, user authentication and authorization are managed.

• **Data Minimization**: WorkerSecure uses minimum data as possible. By collecting only necessary data for detecting safety equipment violations, it reduces the risk of privacy.

• **Regular Audits**: WorkerSecure conducts regular audits. These audits ensure continuous compliance with data privacy laws. Also, they help in identifying any potential security danger in the system.

WorkerSecure is not designed for data privacy but also for ethical standards. System focuses on the welfare and rights of the workers. Key features WorkerSecure must align with these standards are also listed below:

• **Transparency**: WorkerSecure operates with a high degree of transparency. Workers and stakeholders are informed about the data collection processes. As well as the purpose of data collection, and how it is used to enhance safety.

• **Non-Discrimination**: AI models used in WorkerSecure are trained on diverse datasets to ensure that the system does not function bias against any group. WorkerSecure functions for every worker regardless of gender, ethnicity, or other factors.

• **Safety for Workers**: WorkerSecure is designed to alert workers to avoid safety violations. System does this without causing any embarrassment or stigma.

• **Professional Responsibility**: The development team adheres to the ACM Code of Ethics. With an aim of ensuring all decisions made during the project development enhance well-being and avoid harm.

WorkerSecure project ensure that the technology it uses serves to enhance, rather than compromise. Rights and safety of individuals are considered as our primary concern.

10. Deployment Process & Manual

The deployment of WorkerSecure system involves several key steps. These steps are designed to ensure smooth integration and efficiency in real-world. The process is structured to minimize disruptions and maximize the effectiveness.

Pre-Deployment Testing: Before full deployment, WorkerSecure undergoes extensive testing in a simulated environment. The environment must mirror actual working conditions as closely as possible. As mentioned in Testing & Evaluation section, there were some conditions to be further tested. Therefore, stress and weather conditions testing must be conducted.

Installation of Hardware: Necessary hardware components are installed. The components include high-resolution cameras and servers are installed. Also, installation process is carried out by trained technicians. This is to make sure all equipment is placed for maximum performance.

Software Integration: After the hardware is installed, WorkerSecure software is installed on servers. This involves setting up the operating system, deploying the YOLO-based detection algorithms. In addition, all system settings must be considered. This is especially to align with the specific needs of the deployment site.

Data Privacy Configuration: WorkerSecure is configured to encrypt all collected data. To comply with data protection regulations, access control is set up.

Go-Live and Monitoring: WorkerSecure system goes live after a final round of checks. Initial deployment is closely monitored by the development team. This is to ensure that system operates as expected. Any issues detected during this phase are addressed immediately.

Post-Deployment Support: WorkerSecure provides ongoing support and maintenance to handle any issues and to update the system, as needed. Regular updates must already be scheduled to improve functionality.

11. Conclusion

11.1. Summary Of Findings

The hardhat detection system's development was over successfully with a robust system architecture and design. These designs adhere to specific requirements and testing revealed that it has been able to detect hard hat usage easily irrespective of environmental changes. A real time performance evaluation phase proved that the system works well enough—instant safety proto cols can pick up from exactly where they left off.

11.2. Achivements

Some major accomplishments have been the design of a system architecture that can resize and adapt itself easily, using recent machine vision techniques to locate objects within pictures plus creating a thorough test plan which proved how well this new monitoring tool worked; also it was able to deal with privacy concerns about big datasets as well as ethical issues towards animals concerning their use during experiments on humans while confirming entirely that everything was done by law.

11.3. Future Works

In the future, the system's functionality could be broadened to include other types of safety equipment such as vests and goggles, thus advanced level of safety at construction sites is ensured. Hence, the sustained adjustment and scaling of the system is crucial especially when deployed to various regions around the world which have different safety standards and environmental backgrounds. In addition, we will focus on enhancing the system's performance as well as minimizing its environmental impact through further studies.

12. References

- **Object-Oriented Software Engineering, Using UML, Patterns, and Java, 2nd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice-Hall, 2004, ISBN: 0-13-047110-0.**
- **Başkent University. (2020). Bitirme Projesi Yazım Kılavuzu. Retrieved from https://end.baskent.edu.tr/kw/upload/503/dosyalar/Bitirme_Projesi_Yazim_Klavuzu_2020-(1).pdf**