




# Project Proposal

Xployt: The bug bounty platform

## Details of Project Supervisor, Co-supervisor, Advisors, and Clients

Proposed Project Supervisor (Academic Staff of UCSC):	
Name of the supervisor:	Dr. Kasun Gunawardana
Signature of the supervisor:	For Official Signature: kgg@ucsc.cmb.ac.lk
Date:	08/07/2024

Proposed Project Co-Supervisor (Assigned by Course Coordinator):	
Name of the co-supervisor:	Chathumini Thathsarani
Signature of the co-supervisor:	
Date:	08/07/2024

# Content

<b>Details of Project Supervisor, Co-supervisor, Advisors, and Clients.....</b>	<b>0</b>
<b>Content.....</b>	<b>1</b>
<b>1.0 Introduction to Project.....</b>	<b>3</b>
1.1 Problem Statement.....	3
1.2 Our Solution.....	3
1.3 Project Overview.....	4
<b>2.0 Project Goals.....</b>	<b>5</b>
<b>3.0 Scope of the project.....</b>	<b>6</b>
3.1 In-Scope.....	6
3.2 Out Scope.....	8
<b>4.0 Objectives of the Project.....</b>	<b>9</b>
<b>5.0 Project Feasibility.....</b>	<b>10</b>
5.1 Technological Feasibility.....	10
5.2 Operational Feasibility.....	10
5.3 Schedule Feasibility.....	11
5.4 Resource Feasibility.....	11
5.5 Legal Feasibility.....	11
5.6 Ethical Feasibility.....	12
5.7 Financial Feasibility.....	12
<b>6.0 Deliverables of the project.....</b>	<b>13</b>
<b>7.0 Project Constraints and Assumptions.....</b>	<b>14</b>
7.1 Constraints.....	14
7.2 Assumptions.....	14
<b>8.0 Requirements.....</b>	<b>15</b>
8.1 Functional Requirements.....	15
8.1.1 Client User Role.....	15
8.1.2 Project Lead Role.....	16
8.1.3 Hacker User Role.....	16
8.1.4 Validator User Role.....	17
8.1.5 Admin Role.....	17
8.2 Use Case Diagram.....	18
8.3 Use Case Narratives.....	18
8.4 Quality Attributes Requirements.....	19
<b>9.0 System Walkthrough.....</b>	<b>20</b>

9.1 Main workflow.....	20
9.2 Activity Diagram.....	22
10.0 Technologies to be used.....	23
11.0 Project Timeline.....	24
12.0 Declaration.....	25
Appendices.....	26
Appendix A - Use Case Narratives.....	26

## Table of figures

Figure 1 - Use-case diagram.....	18
Figure 2 - Activity diagram.....	22
Figure 3 - Timeline.....	24

# 1.0 Introduction to Project

## 1.1 Problem Statement: The Cybersecurity Challenge

In today's digital landscape, organizations face escalating cybersecurity threats characterized by increasing scale, frequency, and sophistication. This situation presents several key challenges:

1. Identifying Reliable Security Professionals
  - Difficulty in verifying skills and establishing trust
  - Ensuring confidentiality of sensitive information
2. Need for Specialized Expertise
  - Requirement for diverse, up-to-date skills and ethical conduct
3. Lack of Structured Engagement Processes
  - Prevalence of ad-hoc assessments and inconsistent reporting
  - Fragmented communication between parties

Consequences of inadequate security measures include data breaches, financial losses, reputational damage, and legal penalties.

The market lacks a comprehensive solution addressing these challenges, creating a need for:

- A centralized platform connecting organizations with verified professionals
- Standardized processes for vulnerability management
- Trust-building mechanisms and continuous improvement framework

Without such a solution, organizations remain vulnerable to breaches, financial and reputational risks, and regulatory non-compliance.

Addressing these issues can significantly mitigate cybersecurity risks, highlighting the necessity for a robust bug bounty platform to bridge the gap between organizations and security professionals, ultimately creating a more secure digital ecosystem.

## 1.2 Proposed Solution

The project addresses these challenges by developing a comprehensive bug bounty platform that facilitates the identification and resolution of security vulnerabilities in web applications.

Here's how our solution works:

1. **Client Engagement:** Clients submit their websites or applications to our platform for security testing. They work with an assigned project lead to define the scope and parameters of the project.
2. **Hacker Involvement:** Clients invite skilled and vetted hackers to participate in the bug bounty program. These ethical hackers, motivated by rewards, rigorously test the applications for vulnerabilities.
3. **Reporting and Validation:** Hackers submit detailed reports of any vulnerabilities they discover. Our team of validators reviews and validates these findings to ensure their accuracy and relevance, maintaining high standards of reliability and confidentiality.
4. **Compensation and Reporting:** Validated vulnerabilities are compiled into a comprehensive report by the project lead. Hackers are rewarded based on the severity and significance of their findings. Clients receive a final report detailing all identified vulnerabilities and recommended fixes.
5. **Outcome:** By leveraging the collective expertise of ethical hackers, our platform helps organizations fortify their web applications against potential threats. This collaboration enhances their overall security posture and reduces the risk of cyber attacks.

The proposed bug bounty platform not only provides a robust mechanism for identifying security flaws but also ensures that participating professionals are reliable, trustworthy, and adhere to strict confidentiality and ethical standards. This approach fosters a secure and collaborative environment where organizations and ethical hackers work together to create safer digital spaces.

## 2.0 Project Goals

### 1. Enhance Web Application Security

- The primary goal is to enforce the security of web applications by identifying and addressing vulnerabilities. This involves developing a platform that connects organizations with skilled hackers who can effectively pinpoint and resolve security flaws. By leveraging the expertise of these security professionals, the platform aims to reduce the risk of security breaches.

### 2. Develop a Platform to Bridge the Gap between Seekers and Providers of Security Solutions

- To achieve enhanced security, the project will create a comprehensive platform that facilitates seamless interaction between organizations seeking to secure their applications and hackers. This platform will serve as a marketplace where organizations can post their security needs and security professionals can offer their services.

### 3. Foster Collaboration in Cybersecurity

- This platform focuses on creating an environment where clients (organizations) and hackers can work together efficiently. The platform will include features that enable real-time communication, feedback, and collaboration among all parties involved. By fostering a collaborative environment, the platform aims to enhance the quality of security vulnerability identification and resolution.

### 4. Streamline the Bug Reporting Process

- Our platform simplifies the process of reporting, validating, and resolving security issues. The platform will provide intuitive tools for hackers to submit bug reports, for validators to assess and verify these reports, and for organizations to manage and respond to them. By streamlining this process, the platform aims to minimize the time between the identification and resolution of security vulnerabilities.

### 5. Build a Trusted and Reliable Platform

- We aim to establish a reputation for the platform as a secure, dependable, and effective tool for managing bug bounty programs. This will be achieved by implementing security measures, maintaining transparency in operations, and consistently delivering high-quality services.

## 3.0 Scope of the project

### 3.1 In-Scope

#### Actors

1. **Client Organization:** The entity that submits a website or app for security testing.
2. **System Admin:** The administrator responsible for assigning project leads and validators, and overseeing the project.
3. **Project Lead:** The individual who manages the project, communicates with the client, and compiles the final report.
4. **Hacker:** The security expert who tests the website/app for vulnerabilities and submits reports.
5. **Validator:** The professional who validates the vulnerability reports submitted by hackers.

**Platform Development:** Creating user registration and profile management functionalities for Clients, Hackers, Validators, Project Leads, and Admins. Access permissions will be controlled according to user roles across the platform.

**Bug Submission and Validation:** Creating detailed vulnerability submission forms and project report submission forms for Hackers and Validators. This will ensure an accurate assessment of submitted vulnerabilities.

**Communication Tools:** Establishing basic channels for communication between Clients, Hackers, and Validators during bug reporting and the validation process to streamline coordination and minimize conflicts.

**Conflict Resolution:** Facilitating tools and media to resolve disputes that can arise related to compensations.

**Database Integration:** Implementing a backend system to store and retrieve information securely and efficiently.

**Customized Dashboards:** Develop dashboards that allow users to view project data, reports, and other relevant information in an organized manner.

**Email Notifications:** Implementing notifications via emails to deliver time-sensitive information.



## 3.2 Out Scope

### Payment Portal

- **Functional Payment Portal:** The project will not include the development of a functional payment portal. Instead, a mock-up version will be integrated for demonstration purposes only. This decision is made to focus resources on core functionalities.

### Security Emphasis

- **Comprehensive Security Features:** While basic security measures will be implemented to protect user data and ensure platform stability, comprehensive security features will not be developed. The primary goal is to ensure that the platform is complete and functional, with an emphasis on usability and core functionalities.

### Dynamic Project Initialization

- **Static Project Initialization and Deadlines:** Projects will have fixed initiation dates and application deadlines to ensure all hackers within a project work in parallel. This approach avoids the complexity of dynamically handling project initialization and makes the implementation more straightforward and manageable.

### Multiple Platform Development

- **Exclusive Web Platform Development:** We will not develop any applications other than the web platform. Our focus is to create a user-friendly web interface, ensuring that core functionalities are well-implemented and stable.

### Team Participation

- **Independent Work Requirement:** Hackers will be required to work individually and independently.

### Project Submission

- **Web Platform Projects Only:** Only web platforms will be accepted as project submissions.

## 4.0 Objectives of the Project

The ultimate objective of this project is to deliver a fully functional, user-friendly, and secure bug bounty platform that caters to the needs of Clients, Hackers, Validators, Project Leads, and Admins. This platform will streamline the process of reporting, validating, and resolving software bugs, fostering a collaborative and efficient environment for all users.

By the end of the project, we aim to achieve the following specific objectives:

1. **Develop a Secure and Scalable Platform:** Create a platform that can handle multiple concurrent users, ensuring data security, scalability, and reliability.
2. **Enable Comprehensive User Management:** Implement functionalities for user registration, profile management, and role-based access control for Clients, Hackers, Validators, Project Leads and Admins.
3. **Facilitate Bug Submission and Validation:** Provide secure submission forms for hackers to report bugs and a workflow for validators to review, reproduce, and validate reported bugs.
4. **Implement Severity Assessment and Reward Calculation:** Develop tools for collaborative severity assessment and automated reward calculation based on the assigned severity class.
5. **Create an Admin Dashboard:** Provide monitoring tools for admins to oversee platform operations, resolve conflicts, and generate analytics reports.
6. **Establish a Comprehensive Rating System:** Implement a rating system to ensure accountability and trust among clients and hackers, including reviews and calculated scores based on various performance metrics.
7. **Ensure Continuous Integration and Deployment:** Set up CI/CD pipelines to automate testing, deployment, and monitoring, ensuring a seamless development process.

By accomplishing these objectives, we aim to create a secure, collaborative, and efficient bug bounty platform that meets the needs of all stakeholders involved.

## 5.0 Project Feasibility

### 5.1 Technological Feasibility

The proposed platform is technologically feasible for the following reasons:

- **Technology Suitability:** Java, MySQL, and TypeScript have been selected due to their robustness, scalability, widespread use, and suitability to maintain a large project. These languages/ technologies are well-supported and have well-defined documentation and community support.
- **Team Expertise:** Our development team has experience contributing to moderate-sized projects and working with web front ends without third-party frameworks.
- **Testing plan:** A very lightweight testing library is proposed to be developed to be used throughout the implementation phase.

### 5.2 Operational Feasibility

- **User Adoption:** The platform addresses a critical need in the cybersecurity industry, which should encourage adoption by both organizations and ethical hackers. The user-friendly interface and clear workflows will facilitate easy onboarding and use.
- **Scalability:** The platform is designed to handle multiple concurrent projects and users. As the user base grows, the AWS infrastructure can be scaled accordingly to maintain performance.
- **Training and Documentation:** Comprehensive documentation and user guides will be provided for all user roles (clients, hackers, validators, project leads, and admins). This will ensure smooth operation and minimize the learning curve for new users.
- **Monitoring and Reporting:** Built-in analytics and reporting tools will allow administrators to monitor platform performance, user engagement, and project outcomes. This data will inform ongoing operational improvements and strategic decision-making.

## 5.3 Schedule Feasibility

### Project Timeline and Team Structure

Our project follows the Verification and Validation (V Model) for software development, emphasizing test-driven development to ensure a smooth process with minimal bugs. The project spans 11 months, from June to April, with clear milestones and regular progress reviews.

### Team Composition and Commitment:

- 4 team members, each dedicating 20 hours per week
- Total project duration: 48 weeks
- Individual contribution: 960 hours per member
- Total project effort: 3,840 man-hours

The development team utilizes collaboration tools for efficient task allocation, progress supervision, and time management, ensuring clear timelines and responsibilities.

### Project Phases:

1. Requirements & Design (June-September): 1,760 hours
  - Focus: Requirements gathering, project proposal, system architecture, UI design
  - Key Deliverables:
    - Project proposal (August)
    - Software Requirements Specification (SRS) document (September)
2. Development & Testing (September-March): 2,640 hours
  - Core Activities: Development, unit testing, integration testing
  - Continuous deployment and system testing
  - Milestone: Interim Presentation in December (progress report and demo)
3. Finalization (March-April): 320 hours
  - Priorities: Final testing, documentation, launch preparations
  - Culmination: Final Presentation and viva in April

This structured approach, combined with our dedicated team and clear timeline, positions us to deliver a high-quality bug bounty platform efficiently and effectively. Regular reviews and our test-driven methodology will help us maintain course and adapt to challenges throughout the development lifecycle.

## 5.4 Resource Feasibility

- Human Resources: The development team's skills are assumed to be adequate. As noted above, clear role and responsibility definitions will maximize their productivity..
- Financial Resources: Financial resources are required only for the AWS server costs. For the first few months, the free tier will be enough. After that, the team members will pitch in for the server costs.

## 5.5 Legal Feasibility

- Compliance with Data Protection: Organizations will submit their sites for testing. Any information that they need to keep hidden will stay undisclosed. Compliance with data protection regulations GDPR (General Data Protection Regulation) and CCPA(California Consumer Privacy Act) will be mandatory.
- There will be policies clearly stating the payable amounts for each kind of service provided by hackers. The project lead will work with the client organization to clear out any ambiguities before the start of the project.
- Dispute Resolution: Hackers are allowed to submit appeals through the platform in case they feel like the compensated amount is unfair. Organizations are also allowed to submit appeals in case they feel like the payable amount is unfair for the received work. Project leads will investigate any issues before escalating the issue to more severe legal complaints.

## 5.6 Ethical Feasibility

- Ethical Reporting and Conduct: A significant risk is that hackers who find vulnerabilities and instead of reporting them, they exploit them for their own gain. Hackers must adhere to ethical guidelines. Those who are found to be exploiters will be banned from the platform.
- Client and Hacker Relationships: Maintaining transparent and respectful relationships between organizations (clients) and hackers are essential. The platform will require all participants to agree to ethical guidelines. Project Leads and validators will contribute to enforce these guidelines.

## 5.7 Financial Feasibility

- Revenue Model: The platform will generate revenue by deducting a commission from the payments made to hackers. This commission will cover the maintenance costs of the platform, including salaries for project leads and validators, and the remainder will be taken as profit.
- Cost Analysis: Initial costs are minimal as the free tier of AWS server services will be used. After the free tier period, team members will cover the server costs. The commission deducted from hacker payments will cover maintenance costs that will arise as the platform scales.
- Profitability: The commission-based revenue model ensures that as more organizations use the platform and more hackers are paid for their work, the platform's revenue will increase proportionally.

## 6.0 Deliverables of the project

- A functional, reliable web frontend and a backend including proposed features deployed in a server accessible through a URL.
- Well-documented and clean code base
- User Guide
- Administrator Manual Including deployment instruction
- Developer Manual Including instructions to set up the developer environment

## 7.0 Project Constraints and Assumptions

### 7.1 Constraints

- Limited budget, primarily allocated for server configuration.
- Development is limited due to only having four preselected members
- Compliance with data protection regulations such as GDPR and CCPA is mandatory.
- Since it is web-based, a constant internet connection is required.
- Having to develop the system on a strict deadline.
- The amount of projects a single validator can manage at a given time is limited.
- A sandbox is implemented as a payment gateway since an active one would be impractical.

### 7.2 Assumptions

- Validators and Project leads will be well-versed in the field of cybersecurity.
- Expert / intermediate hackers will be enrolled to the platform.
- An adequate literacy of the English language by all users.
- During a dispute, project lead will be unbiased
- A stable internet connection.
- All projects accepted are accomplishable with the available resources in the system.
- Any and all complaints can be resolved using the system itself.



## 8.0 Requirements

### 8.1 Functional Requirements

#### 8.1.1 Client User Role

1. **Request Security Checkup**
  - Clients can request a security checkup by submitting their project details and requirements on the platform.
2. **Consult with Project Lead**
  - After initial submission, the Project Lead reviews the request and provides recommendations.
3. **Select Payment and Visibility Options**
  - Clients can choose minimum payment amounts for each severity class.
  - Clients can select to list the project as public or private.
4. **Deposit Funds**
  - Clients make an initial deposit to cover the agreed payment amount for hackers, held in escrow.
5. **Review Reports**
  - Clients can access and review reports submitted by hackers.
  - Clients can track the progress of the project through the platform.
6. **End of Project Report**
  - Clients receive a comprehensive report from the Project Lead at the conclusion of the project.
7. **Manage Funds**
  - Clients can choose to keep the remaining amount for the next project or retrieve unused funds.
8. **File Complaints**
  - If dissatisfied with the work or if issues arise with validators, clients can file complaints through the platform.

#### 8.1.2 Project Lead Role

1. **Receive and Review Requests**
  - Project Leads get notified of incoming client requests.
  - Review submitted information to ensure legitimacy and project feasibility.
2. **Assist with Project Setup**
  - Help clients decide between public or private listing based on project sensitivity.
  - Configure project parameters such as the number of required hackers.

3. **Initiate Project**
  - Coordinate arrangements between clients and validators.
  - Officially start the project upon confirmation of all parties.
4. **Review and Compile Reports**
  - Review all reports submitted by hackers.
  - Compile reports into a single comprehensive document for client presentation.
5. **Dispute Mediation**
  - Mediate disputes between clients and validators.
  - Document resolutions and update the platform accordingly.
  - In financial disputes, the project lead's decision will be final.

### 8.1.3 Hacker User Role

1. **Registration and Profile Setup**
  - Register on the platform providing email, name, and contact details.
  - Add previous work experience and education.
  - Complete specialties and interests profile.
  - Add payment methods and define a minimum payment amount per report.
2. **Receiving Project Invitations**
  - Receive project invitations through email or platform notifications based on matching algorithm results.
  - Review project details and decide to accept or decline invitations.
3. **Project Participation**
  - Review detailed project descriptions from the project dashboard.
  - Accept and sign project rules and ethics agreements.
4. **Reporting Vulnerabilities**
  - Use the platform's form to submit vulnerability reports.
  - Include detailed steps, impact analysis, and necessary evidence.
  - Submit reports for validation.
5. **Validation and Payment**
  - Receive validation status notifications.
  - Address feedback or requests for more information.
  - Receive payments for validated reports.
6. **Post-Project Review**
  - Rate the project and validator based on experience.
  - Provide comments or suggestions for improvement.
7. **Complaints Handling**
  - Submit complaints through a platform form with relevant details and evidence.
  - Receive acknowledgment and follow-up via email (communication excluded from scope).

## 8. Profile Update

- Update profile with new work experience, education, certificates, and payment preferences.

### 8.1.4 Validator User Role

#### 1. Registration and Profile Setup

- Apply for a vacancy via the platform providing the necessary details.
- Verify email and complete profile setup, including skills and experience.
- Approved by Admins to meet platform standards.

#### 2. Reviewing and Validating Bug Reports

- Receive notifications about new bug reports.
- Review reports for completeness and clarity.
- Reproduce and validate bugs, and assess impact.
- Document findings and severity classification.

#### 3. Conflict Resolution and Dispute Management

- Receive notifications about disputes.
- Review relevant information and assist in resolution with Project Lead.

### 8.1.5 Admin Role

#### 1. Validator Approval

- Interview and approve Validators to ensure they meet qualifications.

#### 2. Assigning Validators

- Recommend validators based on the nature of the project and required cybersecurity expertise.

#### 3. Access Control

- Access and manage any user information as necessary.

#### 4. Vacancy Posting

- Create posts to notify of validator vacancies.

## 8.2 Use Case Diagram



## 8.3 Use Case Narratives

*See Appendix A*

## 8.4 Quality Attributes Requirements

### Performance

- Ensure the platform can handle a high number of concurrent users and large volumes of data efficiently.
- **Approach:** Implement efficient database queries, optimize code, and use caching mechanisms.

### Security

- Protect the platform and user data from unauthorized access and breaches.
- **Approach:** Use encryption for data, and secure user authentication methods.

### Usability

- Create an intuitive and accessible user interface.
- **Approach:** Design a user-friendly interface with clear navigation, provide comprehensive help resources, and ensure the platform is responsive across various devices.

### Maintainability

- Ensure the platform can be easily maintained and updated.
- **Approach:** Adopt modular design principles, and maintain clean and well-documented code.

### Extensibility

- Allow for easy addition of new features and functionalities.
- **Approach:** Design the system architecture to support extensions and modular components, making it easy to integrate new features.

### Testability

- Facilitate comprehensive testing of the platform.
- **Approach:** Implement testing tools and frameworks, and ensure that all components are designed to be easily testable.

## 9.0 System Walkthrough

### 9.1 Main workflow

1. **Client Submits Project:**
  - The Client submits their website or app along with project details to the system.
  - The Client selects the project type (public or private) and invites hackers.
2. **Admin Assigns Project Lead:**
  - The System Admin reviews the submission and assigns a Project Lead to the project.
3. **Initial Project Setup:**
  - The Project Lead communicates with the Client to understand their requirements and help initialize the project.
  - Any ambiguities related to severity classifications considering the specific project will be cleared out. And the mandatory test routine will be discussed and finalized.
  - The Client makes an initial deposit to cover the agreed payment amount for hackers, held in escrow.
4. **Project Visibility and Hacker Invitations:**
  - If the project is public, the Client can both invite hackers and additional hackers can apply to participate.
  - If the project is private client can only invite hackers
  - Invited hackers must accept the invitation to participate.
5. **Assign Validators:**
  - Based on the project size and the number of participating hackers, the System Admin assigns the necessary number of Validators to the project.
6. **Project Start:**
  - The Project Lead officially starts the project once all parties are confirmed.
7. **Vulnerability Testing:**
  - Hackers begin testing the website/app for vulnerabilities.
  - Hackers must complete the mandatory test routine and submit their reports before the deadline.
8. **Validation of Reports:**
  - Validators review, reproduce, and validate the vulnerability reports submitted by hackers.
  - If there are any issues or the reports need clarification, Validators communicate with the Hackers to resolve them.
  - Validators then provide their validated reports to the Project Lead.

**9. Final Report Compilation:**

- The Project Lead compiles all the validated reports and creates a comprehensive final report.
- The Client can review the individual reports from Hackers and Validators before receiving the final report.

**10. Payment Processing:**

- Hackers who have completed the testing receive the predefined payment.
- If additional issues are found by Hackers, they are paid according to the severity of the issues, as determined at the start of the project.

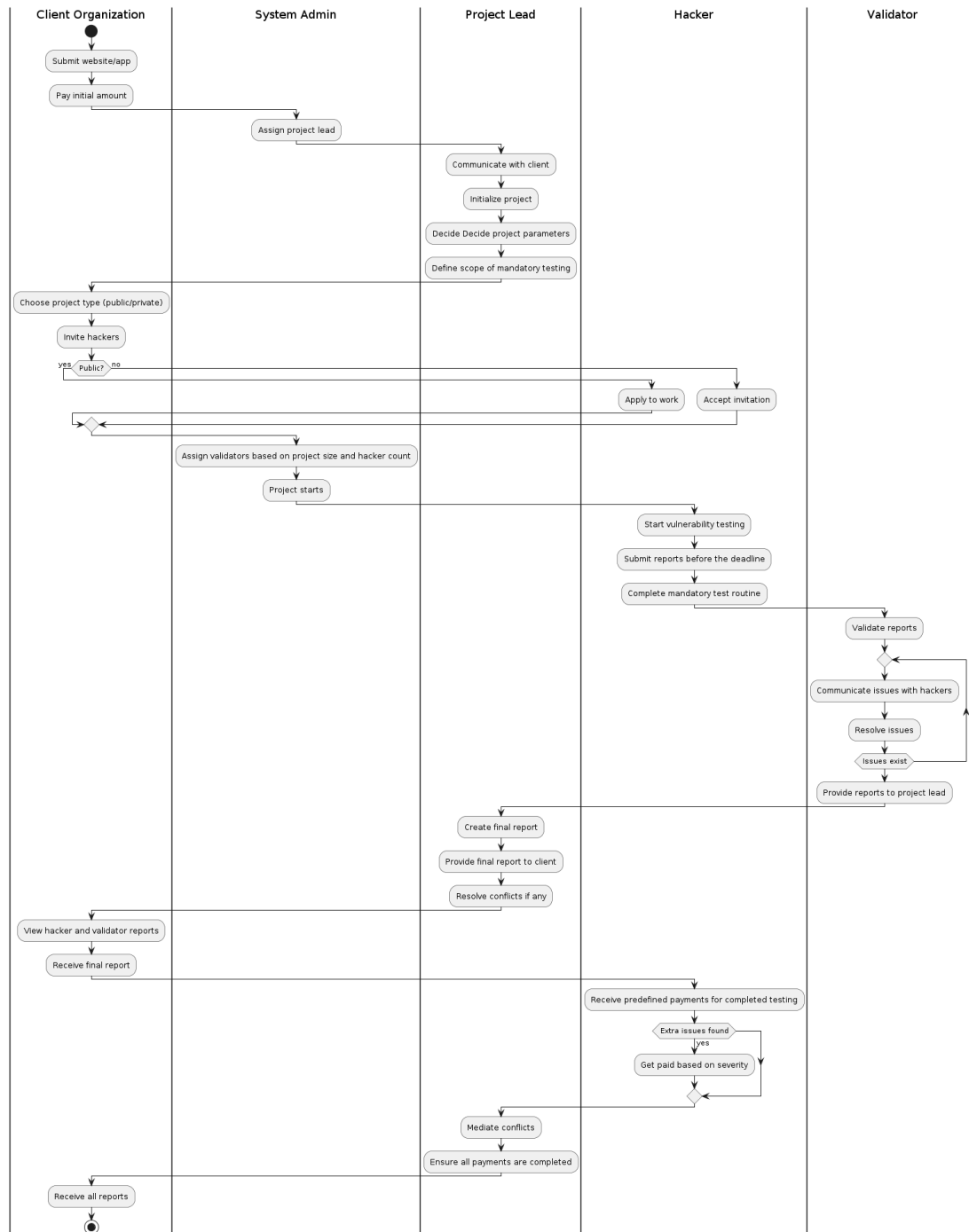
**11. Conflict Resolution:**

- If any conflicts arise during the project, the Project Lead acts as an unbiased mediator to resolve them.

**12. Project Completion:**

- Once all payments are processed and conflicts resolved, the project is considered complete.
- The Client receives all the reports, marking the end of the project.

## 9.2 Activity Diagram





## 10.0 Technologies to be used

- Frontend: HTML, CSS, Typescript  
Typescript was picked instead of JavaScript to utilize the type system. Because in a large project, types will allow us to write code faster and reduce debugging time.
- Backend: Java  
Java was selected for its robustness/ scalability and concrete object-oriented nature. Also To fully utilize OOP concepts and design patterns we learn in SE modules.
- Database: MySQL  
Mysql is a widely used database relational database management system. It offers high performance and ease of use.
- UI Design: Figma
- Testing: Manual/ Internal lightweight testing framework
- Deployment: Amazon Lightsail  
“Amazon Lightsail is a cloud platform that's cost-effective, fast, & reliable with an easy-to-use interface. It's ideal for simpler workloads, quick deployments, and getting started on AWS.” - aws
- Version Control: Git
- Git hosting: Github
- IDEs: VS Code/ Jetbrain IDEs
- Development Tools: XAMPP
- Collaboration Tools: Atlassian Trello, Figjam

## 11.0 Project Timeline


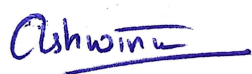


Given below is a Gannt chart of the proposed timeline of the project.

Task	June	July	August	September	October	November	December	January	February	March	Apr
requirements gathering											
Finalize requirements											
Project proposal submission											
System and Architecture Design											
UI Design											
SRS submission											
Development											
Unit Testing											
Interim Presentation											
Integration Testing											
System Testing											
Deployment											
Maintenance											
Final Presentation and viva											

As we're following the Verification and Validation model, there will be recurring testing and deploying phases. A major part of the time will be dedicated to development. We aim to develop a fully functional platform by the end of March 2025.

## 12.0 Declaration

*We as members of the project titled **Xploit**, Certify that we will carry out this project according to the guidelines provided by the coordinators and supervisors of the course as well as we will not incorporate, without acknowledgment, any material previously submitted for a degree or diploma in any university. To the best of our knowledge and brief, the project work will not contain any material previously published or written by another person or ourselves except where due reference is made in the text of appropriate places.*

Index No	Name	Signature
22001385	Lakshith Nishshanke	
22000747	Ashwinie Jayamanna	
22002146	Maahela Weerasinghe	
22001451	Savith Panamgama	

# Appendices

## Appendix A - Use Case Narratives

Use Case ID	1
Use Case Name	Project Request by Client
Primary Actor	Client
Participating Actors	Project Lead
Description	The process of requesting a project lead to create a project.
Preconditions	<ul style="list-style-type: none"><li>• Client having an account registered in the platform.</li><li>• Client understands the exact role of the platform.</li><li>• Project leads are available to be assigned to new projects.</li></ul>
Main Flow	<ol style="list-style-type: none"><li>1. Client posts a project request on the platform.</li><li>2. A project lead gets assigned to the said request.</li><li>3. Project leads accept said project request.</li><li>4. Project lead and client, review and update the project requirements.</li><li>5. Client decides to post the job publicly.</li></ol>
Alternate Flows / Exceptions	<ul style="list-style-type: none"><li>• Client decides to list privately. Invitations are sent via email and through the platform to the chosen hackers.</li><li>• Project request is declined.<ul style="list-style-type: none"><li>○ Project lead finds that the project is suspicious and declines the request.</li><li>○ Ratings of said client is reduced</li></ul></li></ul>
Postconditions	<ul style="list-style-type: none"><li>• Adequate hackers have agreed to work on the project.</li><li>• Project is initiated by the project lead.</li></ul>

Use Case ID	2
Use Case Name	Project Initiation
Primary Actor	Project Lead
Participating Actors	Client
Description	Project Lead manages the initiation of a project by reviewing the client's request, configuring the project, assigning validators, and recommending suitable hackers.
Preconditions	<ul style="list-style-type: none"> <li>Client requests to create a project and submits required information.</li> </ul>
Main Flow	<ol style="list-style-type: none"> <li>1. An available project lead is assigned to the project.</li> <li>2. Project Lead reviews the submitted details and accepts the request if all requirements are fulfilled.</li> <li>3. Project Lead assists the client in making decisions regarding project configuration.</li> <li>4. Evaluates whether the project should be listed publicly or conducted privately, considering the client's budget and platform nature, and provides recommendations.</li> <li>5. Recommends the number of hackers to be hired for the project based on the project timeline and deadlines.</li> <li>6. Suggests suitable hackers if the project is done privately.</li> <li>7. Assigns validators appropriate for the project, determining the number based on the project's scale and the number of hackers requested.</li> <li>8. Initiates contact between validators and the client to finalize the project setup.</li> <li>9. Initiates the project.</li> </ol>
Alternate Flows / Exceptions	<ul style="list-style-type: none"> <li>If the request is deemed suspicious due to the nature of the client's platform or the client's background, the Project Lead rejects the request.</li> <li>Client details are stored, and a lower rating with negative remarks is assigned.</li> </ul>
Postconditions	<ul style="list-style-type: none"> <li>Project is configured and initiated successfully.</li> <li>Validators and hackers are assigned and informed of their roles.</li> </ul>

Use Case ID	3
Use Case Name	Private Project invitations
Primary Actor	Hacker
Participating Actors	Project lead, Client
Description	The Hackers are invited to a project based on their interests. The hackers can view the project details, payments, etc, and accept or decline the invitation.
Preconditions	<ul style="list-style-type: none"> <li>• The project lead has set up the project and assigned one or more validators to it.</li> <li>• The project lead has defined project requirements, recruitment deadline, and the number of hackers</li> <li>• Sufficient hackers are registered</li> </ul>
Main Flow	<ol style="list-style-type: none"> <li>1. The system matches hackers using the defined project requirements by the project lead and hackers' specialized areas</li> <li>2. The system sends invitations to the matched hackers</li> <li>3. Hackers are notified via email and platform notifications</li> <li>4. Hacker reviews project details provided in the invitation, including payment amounts</li> <li>5. The Hacker accepts the invitation</li> <li>6. The Hacker is recruited for the project</li> <li>7. The invitation expires after the deadline</li> </ol>
Alternate Flows / Exceptions	<p>If The Hacker declines an invitation, the system records the response and does not recruit them.</p> <p>If a Hacker accepts the invitation but the required number of hackers for a project has been filled, The Hacker will be notified, and they will not be recruited</p>
Postconditions	<ul style="list-style-type: none"> <li>• Hacker views the project dashboard</li> </ul>

Use Case ID	4
Use Case Name	Report Submission
Primary Actor	Hacker
Participating Actors	Validator
Description	The Hackers perform vulnerability tests and submit comprehensive reports stating their findings, steps of recreation, and severity
Preconditions	Hackers are recruited to the project Validators are assigned to one or more hackers
Main Flow	<ol style="list-style-type: none"> <li>1. The Hacker views details from the project dashboard.</li> <li>2. The Hacker accesses the vulnerability reporting form from the dashboard.</li> <li>3. The Hacker completes the form, providing detailed steps to reproduce the vulnerability, impact analysis, and necessary evidence.</li> <li>4. The Hacker submits the completed report for validation.</li> </ol>
Alternate Flows / Exceptions	<ul style="list-style-type: none"> <li>● Incomplete Report <ul style="list-style-type: none"> <li>○ The system prompts The Hacker to provide the missing information before submission.</li> <li>○ Continue from step 3</li> </ul> </li> <li>● Submission Failure due to technical issues: <ul style="list-style-type: none"> <li>○ The Hacker is notified</li> <li>○ The Hacker retries the submission.</li> </ul> </li> </ul>
Postconditions	<ul style="list-style-type: none"> <li>● Assigned validators are notified regarding the report submission</li> <li>● The submitted report is available for validation and further action.</li> </ul>

Use Case ID	5
Use Case Name	Validator Reviewing Bug Reports
Primary Actor	Validator
Participating Actors	Client, Project Lead
Description	Validators review and validate vulnerability reports from hackers, ensuring accuracy and completeness. They provide feedback for improvements, accept satisfactory reports for hacker compensation, and compile their findings into a final report for the project lead. Unsatisfactory reports are rejected, with reasons documented, possibly affecting The Hacker's rating.
Preconditions	<ul style="list-style-type: none"> <li>• A project has been requested by a client.</li> <li>• The project lead has set up the project and assigned one or more validators to it.</li> <li>• The project has a maximum number of hackers, as requested, working on verifying the security of the web application or server.</li> <li>• Each validator has been assigned one or more hackers.</li> <li>• Hackers have had sufficient time to submit vulnerability reports, confirming the security of the site or reporting reproducible bugs before the project deadline.</li> </ul>
Main Flow	<ol style="list-style-type: none"> <li>1. All reports submitted by hackers involved in the project are made visible to the validators.</li> <li>2. Each validator reviews the reports submitted by The Hackers assigned to them.</li> <li>3. The validator provides feedback and suggestions to improve the completeness and accuracy of the report.</li> <li>4. The Hacker updates the report, incorporating the validator's feedback.</li> <li>5. This review and feedback process may repeat a few times until the validator deems the report satisfactory.</li> <li>6. Once the report is satisfactory, the validator accepts it.</li> <li>7. The Hacker is paid the agreed amount from the deposit held in escrow.</li> <li>8. The validator compiles a report that includes details from The Hacker reports assigned to them.</li> <li>9. This report also includes information about the confirmation process for any reported bugs and additional feedback.</li> </ol>



	10. The validator submits their report through the platform for review by the project lead.
Alternate Flows / Exceptions	<ul style="list-style-type: none"> <li>● Rejection of Reports: <ul style="list-style-type: none"> <li>○ The validator reviews a report and determines it does not meet the minimum criteria.</li> <li>○ The validator rejects the report.</li> <li>○ Information about rejected reports is documented in the validator's report.</li> <li>○ Hackers with rejected reports may have their ratings reduced if the reports are considered intentionally misleading or of very poor quality.</li> </ul> </li> </ul>
Postconditions	<ul style="list-style-type: none"> <li>● The project lead reviews the reports submitted by each validator and prepares the final project report.</li> <li>● The final project report is sent to the client.</li> <li>● The work done by each hacker may be publicly disclosed if the client grants permission, considering the nature of the report.</li> </ul>

Use Case ID	6
Use Case Name	Compensating Hackers for reports
Primary Actor	Hacker
Participating Actors	Client, Validator, Project Lead
Description	The process of compensating The Hacker based on their report
Preconditions	<ul style="list-style-type: none"> <li>• A hacker has submitted a report</li> <li>• The validator has reviewed and accepted the report</li> <li>• The funds have been allocated.</li> </ul>
Main Flow	<ol style="list-style-type: none"> <li>1. The system triggers a payment process through the payment gateway using the initial project fund.</li> <li>2. The payment is made to the hacker</li> <li>3. The Hacker is notified that the payment has been made via email and platform notifications</li> <li>4. The hacker logs into the platform and marks the payment as acknowledged.</li> </ol>
Alternate Flows / Exceptions	<ul style="list-style-type: none"> <li>• Payment Not Received <ul style="list-style-type: none"> <li>○ If the payment is not received, the hacker is prompted to file a complaint through the platform.</li> <li>○ The system logs the complaint.</li> </ul> </li> <li>• Payment Gateway Failure <ul style="list-style-type: none"> <li>○ If the payment gateway fails during the transaction, the system retries the payment process and notifies the hacker and the validator of the delay.</li> </ul> </li> </ul>
Postconditions	The payment status is updated in the system.