# WBS

## PROJECT MANAGEMENT 101

### Project Charter Completion

- Draft Project Charter
- Define Objectives, Deliverables, and Scope
- Identify Stakeholders and Roles
- Document Risks, Assumptions, and Constraints
- Obtain Sponsor and Faculty Advisor Consent

### Setting Up Resources and Environment

- Provision Local/Cloud Server Development Environment
- Install Development Tools (Python, Node.js, React, PostgreSQL)
- Implement Intern Workstations (e.g., VM, browser plug-in)
- Make OSINT Tools Available (Sherlock, Maltego, Shodan, theHarvester)
- Provide Free Access to Free APIs (WHOIS, HaveIBeenPwned, Hunter.io)

### Tracking Progress

- Develop Progress Checker (e.g., Google Sheet/LMS/Microsoft Planner)
- Track Commits and Pull Requests in GitHub Daily
- Conduct Periodic Sprint Reviews with Interns
- Encourage Learning Logs and Reflection During Internship
- Track Web Application Development Milestones

### Final Dissemination and Deliverables

- Assemble End Project Documentation
- Prepare Capstone Presentation Slides
- Deliverables to Faculty Advisor
- Issue Certificates of Internship to Students
- Conduct Project Close-Up Meeting with Stakeholders

## WEB APPLICATION DEVELOPMENT

### Finalization of Requirements

- Draft Feature Set (authentication, input forms, OSINT modules, reports, dashboard)
- Define Tech Stack (React, Node.js, PostgreSQL, Python)
- Verify Requirements with Faculty Advisor and End-Users (cybersecurity analysts)

### UI/UX Design

- Develop Wireframes (dashboard, input forms, login pages)
- Design UI for Report Viewer (to view PDF previews)
- Establish Brand Imaging Protocol
- Conduct Design Reviews with Interns and Faculty Advisor

### Front-End Development

- Install React Environment (using CDN: cdn.jsdelivr.net)
- Implement Authentication UI (OAuth-based login)
- Build Input Forms (for domain, email, phone number)
- Construct Dashboard Components (report status, history table)
- Style with Tailwind CSS

### Back-End Development

- Install Node.js Server with Express
- Implement Authentication API (OAuth/JWT)
- Design Task Scheduler for OSINT Queries
- Install PostgreSQL Database for Report Storage Table
- Develop API to Generate and Download Reports

**OSINT Modules**

- Domain and Subdomain Module
  - Integrate WHOIS API (Python whois library)
  - Implement Subdomain Enumeration (Sublist3r, crt.sh)
- Neighboring IP Scanner Module
  - Integrate Shodan/Censys APIs
  - Scout IP Neighbors and Open Ports
- Exposed Files Finder Module
  - Implement Google Dorking Queries (GHDB)
  - Scan for .bak, .sql, .cer files
- GitHub Leak Detector Module
  - Integrate Trufflehog/Gitleaks
  - Query GitHub via API to Find Sensitive Data
- Credential Leaks Module
  - Embed HaveIBeenPwned/Dehashed APIs
  - Check Credential Exposures
- Email Harvesting Module
  - Integrate theHarvester/Hunter.io
  - Pull Email Data Using Regex and APIs
- Username Enumeration Module
  - Install Sherlock/WhatsMyName
  - Cross-link Usernames
- Metadata Extraction Module
  - Utilize ExifTool for Images/PDFs
  - Extract EXIF data and Timestamps
- Optional Phone Number Recon Module
  - Research Phone Number APIs (if available)
  - Implement Basic Phone Number Validation with Regex

**Report Generation**

- Develop JSON to HTML Mapping Logic for Report Structure
- Apply HTML-to-PDF Conversion (pdfkit/weasyprint)
- Implement AES-256 Encryption for PDF Reports

- Design Report Download Button with Authentication
- Design Report Template (branded uniform formatting)

**Testing and Debugging**

- Conduct Unit Testing of OSINT Modules
- Perform Integration Testing (frontend-backend)
- Eliminate Bugs and Improve Performance (e.g., report generation under 5 minutes)

**Deployment and Handover**

- Deploy Locally or on Cloud (AWS EC2)
- Produce Deployment Documentation
- Create Project Demo Video
- Commit Final Code to GitHub Repository
- Hand Over Deliverables to Faculty Advisor and End-Users