🛡️ Anti-Phishing Platform Hackathon

1-Day Build Challenge

Date: [Insert Date]
Duration: 09:00 - 20:00 (11 hours)
Goal: Build a working MVP demonstrating core anti-phishing capabilities for SMEs

---

🎯 Hackathon Objectives

- Create a functional prototype covering prevention, detection, and response
- Demonstrate at least 3 core features working end-to-end
- Prepare a 10-minute pitch/demo for judges or stakeholders
- Establish technical foundation for further development

👥 Team Structure (Recommended 4-6 people)

- **1 Full-Stack Developer** - Main application & integration
- **1 Frontend Developer** - Dashboard & UI/UX
- **1 Backend/Security Developer** - Detection engine & APIs
- **1 ML/Data Specialist** - Threat detection algorithms
- **1 Designer** - UI/UX, branding, presentation
- **1 Business/Product Person** - Requirements, testing, pitch deck

---

🗓️ Timeline & Milestones

09:00 - 09:30 | Kickoff & Planning (30 min)

- Team introductions and role assignment
- Review objectives and scope
- Set up communication channels (Slack, Discord)

- Define MVP feature set (vote on priorities)
- Create GitHub repository and project board

Deliverable: Project plan, task assignments, repo setup

09:30 - 10:00 | Architecture & Design Sprint (30 min)

- Sketch system architecture diagram
- Define data models and API endpoints
- Choose tech stack (React, Node.js, Python, etc.)
- Design basic wireframes for key screens
- Set up development environments

Deliverable: Architecture diagram, wireframes, boilerplate code

---

10:00 - 13:00 | Sprint 1: Core Development (3 hours)

Focus: Email Analysis & Detection Engine

Backend Team:

- Build email parsing API endpoint
- Implement basic phishing indicators:
    - Suspicious link detection (regex patterns)
    - Sender verification (SPF/DKIM check simulation)
    - Urgency language detection (keyword matching)
    - Attachment analysis (file type checking)
- Create threat scoring algorithm (0-100 scale)
- Set up mock threat intelligence database

Frontend Team:

- Build email submission interface
- Create threat analysis results display
- Design risk score visualization

- Implement responsive layout

ML/Data Team:

- Prepare sample phishing email dataset (50-100 examples)
- Build simple classification model (Naive Bayes or logistic regression)
- Create feature extraction pipeline
- Test model accuracy

Designer:

- Finalize color scheme and branding
- Create icon set for threat types
- Design presentation template
- Prepare demo scenarios

Deliverable: Working email analysis feature with threat scoring

---

13:00 - 13:45 | Lunch Break & Demo Prep (45 min)

- Order food or team lunch
- Quick progress check-in
- Adjust afternoon priorities
- Fix critical bugs

13:45 - 16:30 | Sprint 2: Dashboard & Response (2h 45min)

Focus: Admin Dashboard & Incident Response

Backend Team:

- Build alert system API (email/webhook)
- Create incident logging endpoint
- Implement user management (simple auth)

- Build reporting API (threat statistics)

Frontend Team:

- Build admin dashboard with:
  - Real-time threat feed
  - Statistics overview (charts)
  - Recent incidents list
  - Quick action buttons
- Create incident detail view
- Build one-click response actions UI

ML/Data Team:

- Integrate ML model with backend API
- Create threat trend analysis
- Build anomaly detection for email patterns
- Prepare demo dataset with realistic scenarios

Designer/Business:

- Create demo script and storyline
- Prepare 3-5 realistic phishing scenarios
- Design pitch deck structure (10 slides max)
- Draft value proposition and positioning

Deliverable: Complete dashboard with live threat detection

16:30 - 18:30 | Sprint 3: Training Module & Polish (2 hours)

Focus: Employee Training & Integration

Backend Team:

- Build training content API
- Create quiz/assessment endpoint

- Implement progress tracking
- Set up demo data seeding

Frontend Team:

- Build training portal with:
  - Interactive lessons (3-5 slides)
  - Knowledge check quiz
  - Phishing simulation interface
  - Progress tracker
- Polish overall UI/UX
- Fix responsive design issues

ML/Data Team:

- Fine-tune model parameters
- Create threat intelligence feed simulation
- Build metrics dashboard data
- Prepare performance statistics

Integration:

- Connect all components end-to-end
- Test complete user flows
- Deploy to staging environment
- Set up demo accounts

Designer/Business:

- Finalize pitch deck
- Record demo video (backup plan)
- Prepare handout/one-pager
- Practice pitch

Deliverable: Complete MVP with training module

18:30 - 19:30 | Testing & Demo Prep (1 hour)

All Hands:

- End-to-end testing of all features
- Bug bash session (fix critical issues only)
- Load demo data and test accounts
- Practice complete demo flow (3 run-throughs)
- Finalize pitch presentation
- Prepare Q&A responses
- Create video walkthrough (backup)

Deliverable: Tested, demo-ready application

19:30 - 20:00 | Presentation & Wrap-up (30 min)

- Team presents to judges/stakeholders
- 10-minute demo + 5-minute Q&A
- Feedback collection
- Team retrospective
- Celebrate! 🎉

🛠️ Recommended Tech Stack

Frontend

- **Framework:** React with Vite (fast setup)
- **UI Library:** Tailwind CSS + shadcn/ui
- **Charts:** Recharts or Chart.js
- **Icons:** Lucide React

Backend

- **Runtime:** Node.js + Express or Python + FastAPI
- **Database:** PostgreSQL with Supabase (managed) or SQLite (quick)
- **Auth:** Clerk or Auth0 (quick integration) or JWT
- **APIs:** REST (keep it simple for hackathon)

ML/Detection

- **Language:** Python
- **Libraries:** scikit-learn, pandas, NLTK/spaCy
- **Hosting:** Same server as backend or separate microservice

Deployment

- **Frontend:** Vercel or Netlify (1-click deploy)
- **Backend:** Railway, Render, or Heroku
- **Database:** Supabase, PlanetScale, or Neon

Tools

- **Version Control:** GitHub
- **Project Management:** GitHub Projects or Trello
- **Communication:** Slack or Discord
- **Design:** Figma (collaborative)

🎯 MVP Feature Priority

Must Have (P0)

1. ✅ Email submission and analysis
2. ✅ Threat scoring with explanation
3. ✅ Basic admin dashboard
4. ✅ Simple incident alert system

Should Have (P1)

5. ✅ Training module with quiz
6. ✅ Threat statistics and trends
7. ✅ User authentication
8. ✅ Responsive design

Nice to Have (P2)

9. Real-time notifications
10. Email integration (forwarding)
11. Domain monitoring demo
12. Mobile view optimization

📊 Demo Scenarios

Scenario 1: Phishing Email Detection

- User receives suspicious "password reset" email
- Submits to platform for analysis
- System identifies 5 red flags
- Threat score: 85/100 (High Risk)
- Admin receives alert

Scenario 2: Employee Training

- New employee logs into training portal
- Completes interactive phishing awareness lesson
- Takes quiz (5 questions)
- Receives simulated phishing email
- Reports it correctly → +10 security score

Scenario 3: Incident Response

- Compromised employee credential detected
- Automated alert sent to admin
- Admin reviews incident details
- One-click actions: lock account, notify team, log incident
- View full incident report

🎨 Presentation Structure (10 minutes)

Slide 1: Problem (1 min)

- SMEs lose $X billion annually to phishing
- 90% of breaches start with phishing
- Existing solutions too expensive/complex

Slide 2: Solution (1 min)

- Affordable, all-in-one anti-phishing platform
- Prevention + Detection + Response + Training

Slide 3: Demo - Detection (2 min)

- Live demo: analyze phishing email
- Show threat scoring and red flags

Slide 4: Demo - Dashboard (2 min)

- Show admin view with statistics
- Demonstrate incident response

Slide 5: Demo - Training (2 min)

- Employee training module walkthrough
- Simulated phishing test

Slide 6: Market & Business Model (1 min)

- Target market: 30M SMEs globally
- Pricing: $8-15 per user/month
- Distribution: Direct + MSP partners

Slide 7: Next Steps (1 min)

- Roadmap for next 6 months
- Funding requirements
- Team expansion plans

Q&A: 5 minutes

🏆 Judging Criteria (if applicable)

- **Innovation (25%)** - Unique approach or features
- **Technical Execution (25%)** - Code quality, architecture
- **Completeness (20%)** - Working features, polish
- **Business Viability (15%)** - Market fit, monetization
- **Presentation (15%)** - Clarity, storytelling, demo

📝 Pre-Hackathon Checklist

1 Week Before

- [ ] Confirm team members and roles
- [ ] Set up communication channels
- [ ] Share tech stack documentation
- [ ] Collect sample phishing emails dataset
- [ ] Set up cloud accounts (Vercel, Supabase, etc.)

2 Days Before

- [ ] Create GitHub organization/repository

- [ ] Prepare boilerplate code templates
- [ ] Set up project management board
- [ ] Confirm venue and equipment
- [ ] Order food/snacks

Day Before

- [ ] Test all development environments
- [ ] Share hackathon schedule with team
- [ ] Prepare backup internet connection
- [ ] Charge all devices
- [ ] Print any necessary materials

Morning Of

- [ ] Team arrives 15 min early
- [ ] Test projector/screen sharing
- [ ] Set up workstations
- [ ] Quick team sync
- [ ] Start timer! ⏰

🚨 Contingency Plans

If running behind schedule:

- Cut P2 features immediately
- Focus on polish over new features
- Use mock data instead of real integrations
- Simplify UI to basic functional design

If technical issues:

- Have backup demo video ready
- Use local development instead of cloud
- Switch to simpler tech (SQLite vs PostgreSQL)
- Presenter demos on their local machine

If team member unavailable:

- Reassign critical tasks to other members
- Focus on 2-3 core features only
- Simplify architecture
- Use more pre-built components

🎁 Post-Hackathon Actions

Immediate (Next Day)

- Share code on GitHub (clean up commits)
- Document setup instructions
- Send thank you to team
- Collect feedback survey

Week 1

- Schedule post-mortem meeting
- Prioritize post-hackathon roadmap
- Apply for accelerators/grants
- Reach out to potential beta customers

Month 1

- Refactor code for production
- Implement security best practices
- Add comprehensive testing
- Plan next sprint

💡 Pro Tips

1. **Start with paper** - 15 minutes of planning saves hours of coding

2. **Use existing libraries** - Don't reinvent the wheel
3. **Mobile-first design** - Easier to scale up than down
4. **Mock everything non-critical** - Real integrations can wait
5. **Demo-driven development** - Build what you'll show
6. **Commit often** - Push code every 30 minutes
7. **Test as you go** - Don't wait until the end
8. **Keep it simple** - Working simple > broken complex
9. **Prepare for failure** - Have backup plans
10. **Have fun!** - Best ideas come from relaxed minds

📞 Emergency Contacts

- **Technical Support:** [Name, Phone]
- **Venue Contact:** [Name, Phone]
- **Team Lead:** [Name, Phone]
- **Food Delivery:** [Service, Account]

🎉 Success Metrics

At the end of the day, you should have:

- ✅ Working MVP deployed to the cloud
- ✅ 3+ core features functional
- ✅ 10-minute presentation ready
- ✅ Demo video recorded
- ✅ GitHub repository organized
- ✅ One-pager/pitch deck completed
- ✅ Exhausted but proud team! 💪

Good luck! Let's build something amazing! 🚀