DDSC Website Hackathon Initiatives

Community Vibe Coding Hackathon
📅 **Thursday, November 13, 2025** | ⏰ **16:30 - 20:00** | 📍 **Gærtorvet 1, 5, 1799 København V**

Quick Links: 💬 Slack Channel | 💻 GitHub Repository

🗓️ Schedule

16:30-17:30 | Welcome & Team Formation

16:30-16:40 | Arrivals & Settling In (10 min)

- Get comfortable, grab a drink, find a seat

16:40-16:45 | Welcome by Kasper Junge (5 min)

- Welcome to the hackathon
- Practical details (location, facilities, pizza timing)
- Program overview

16:45-16:50 | Opening Words by Kasper Groes (5 min)

- Welcome from the Chair of DDSC
- Context and importance of today's work

16:50-17:05 | Guest Speaker: Nikolaj Englmayer Münster (15 min)

- Presentation on agentic coding (vibe coding) perspective
- Tools and techniques
- What he's built so far

17:05-17:15 | Q&A with Nikolaj (10 min)

- Questions and discussion

17:15-17:20 | Hackathon Process Brief (5 min)

- How the hackathon will run
- Git workflow, collaboration, demos

17:20-17:30 | Group Formation (10 min)

- Review initiatives
- Form teams of 2-3 people
- Select your initiative

17:30-19:30 | Coding Time 🍕

- We'll be vibin' 🔥
- Pizza will arrive 18:30 - eat while you vibe code!
- Ask for help when needed

19:30-20:00 | Demo Presentations

- Show what you built (even if incomplete!)
- Share learnings and challenges
- Celebrate wins together

Overview

This hackathon is about improving the DDSC website based on community feedback. We'll work in groups of 2-3 people on initiatives identified in our community workshop facilitated by **Anna Jelletoft** during the general assembly 2025 in Aarhus.

🎯 Why We're Building This

Community Motivations:

- Open source community as key driver
- Network, events, and mini-meetups
- Platform for new knowledge and professional insights

Barriers to Remove:

- Technical issues accessing communication platforms
- Geographical distance from Copenhagen

- Lack of awareness of DDSC existence and events
- Finding time to engage

Today's Goal: Build features that strengthen what motivates members while breaking down barriers to engagement.

👥 Team Formation & Getting Started

Forming Your Team (2-3 people)

- **Mix experience levels**: Ideally, each group should have at least one person with a vibe coding tool subscription (Cursor, GitHub Copilot, Claude Code, etc.) who is comfortable using AI coding assistants and has experience with building web apps
- **Choose your initiative**: Browse the initiatives below and select one that excites your team
- **Join Slack**: Stay connected in #hackathon-nov-2025

Development Workflow

1. Create a feature branch

```
git checkout -b feature/your-initiative-name
```

2. Work collaboratively

- Use pair programming with AI tools
- Commit frequently with clear messages

3. Submit a Pull Request

- Use clear PR description
- Reference the initiative number
- Tag @maintainers for review

✅ Quick Reminders

- **Focus on MVP**: Build the core functionality first, polish later
- **Ask for help**: Use Slack, ask other teams, reach out to maintainers

- **Prepare for demo**: Have your feature ready to show (even if partial)
- **Have fun**: This is about learning and community, not perfection!

🚀 Initiatives

⭐ High Priority - Start Here!

These initiatives directly address the biggest community barriers and have the most immediate impact.

1. Newsletter Subscription System

Group Size: 2-3 people
Skills Needed: Django, HTML/CSS, Bootstrap, MailerLite API
Impact: Address barrier of "lack of awareness of events"

Description

Implement a comprehensive newsletter subscription system to keep members informed about community activities.

Tasks

- Implement newsletter signup widget for homepage/footer
- Create standalone newsletter landing page
- Integrate with MailerLite API
- Add subscription preferences (event types, frequency)
- Mobile-responsive design
- Unsubscribe workflow
- Confirmation email flow

Workshop Alignment

Solves "Lack of knowledge of DDSC existence and issues getting information on when events happen"

Technical Notes

- Use existing `news` app in codebase

- Extend `NewsSubscriber` model for preferences
- Create Celery tasks for subscription confirmation
- Follow existing email template patterns in `shared/emails.py`

Success Criteria

- [ ] Newsletter signup form on homepage
- [ ] Standalone landing page created
- [ ] MailerLite integration working
- [ ] Confirmation emails sent
- [ ] Mobile responsive
- [ ] User preferences saved

2. Optimized Landing Page Redesign

Group Size: 2-3 people

Skills Needed: HTML/CSS, Bootstrap, UX/UI design, Django templates

Impact: First impressions, increase engagement

Description

Redesign the homepage to better communicate DDSC's value proposition and increase engagement.

Tasks

- Design hero section with clear value proposition
- Showcase Open Source projects prominently (motivation from workshop)
- Add event calendar preview widget
- Include testimonials from community members
- Create clear CTAs for membership, events, and newsletter
- Mobile-first responsive design
- Accessibility improvements (WCAG 2.1)
- Performance optimization (lazy loading images)

Workshop Alignment

"Strengthen communicative outreach to increase awareness"

Technical Notes

- Edit `home/home.html` template
- Use existing Bootstrap 4.3.1 framework
- Leverage Chart.js for visual elements
- Ensure CSRF protection on forms
- Follow existing template structure in `home/base.html`

Success Criteria

- [ ] Modern, engaging hero section
- [ ] Open source projects highlighted
- [ ] Event calendar preview
- [ ] Mobile responsive
- [ ] Page load time < 2 seconds
- [ ] WCAG 2.1 Level AA compliant

3. Improved Onboarding Flow

Group Size: 2-3 people
Skills Needed: Django, Forms, Email templates, UX design
Impact: Reduce "technical barriers to be active on communication platforms"

Description

Streamline the user registration and onboarding process to reduce friction and improve activation rates.

Tasks

- Streamline user registration flow (reduce steps)
- Create welcome email with getting started guide (User story 5.2)
- Build profile completion wizard
- Integrate Slack invitation link
- Implement first-time user tour/tooltips
- Improve email verification UX
- Auto-send introduction to DDSC resources upon signup

Workshop Alignment

"Look into how to improve technical onboarding to DDSC's communication channels"

Technical Notes

- Modify `users/views.py` registration flow
- Create new email template in `users/templates/users/emails/`
- Use existing Celery task pattern from `users/tasks.py`
- Add wizard steps to `users/forms.py`
- Store onboarding progress in session

Success Criteria

- [ ] Registration reduced to 3 steps or less
- [ ] Welcome email sent automatically
- [ ] Profile completion wizard working
- [ ] Slack invitation integrated
- [ ] First-time user tour implemented
- [ ] User feedback positive

4. Azure Cloud Migration

Group Size: 2-3 people

Skills Needed: Docker, Azure, PostgreSQL, Redis, Nginx, DevOps

Impact: Scalability, reliability, professional hosting (may extend beyond hackathon)

Description

Migrate the Django application from current hosting to Azure cloud infrastructure.

Tasks

- Set up Azure Container Instances or App Service
- Migrate database to Azure PostgreSQL
- Configure Azure Redis Cache
- Set up Azure Blob Storage (replacing DigitalOcean Spaces)
- Configure CI/CD pipeline to Azure (GitHub Actions)
- SSL/domain configuration
- Environment variables and secrets management

- Monitoring and logging setup (Azure Monitor)
- Backup and disaster recovery plan

Technical Notes

- Use existing `Dockerfile` in repository
- Migrate `docker-compose.yml` services to Azure
- Update `ddsc_web/settings/prod.py` for Azure
- Configure Azure Storage for `custom_storages.py`
- Update deployment scripts in `deployment/` directory
- Test with staging environment first

Success Criteria

- [ ] Application running on Azure
- [ ] Database migrated successfully
- [ ] Redis cache working
- [ ] Static files served from Azure Blob
- [ ] CI/CD pipeline operational
- [ ] SSL certificate configured
- [ ] Monitoring enabled
- [ ] Documentation updated

5. Dynamic Page Management System

Group Size: 2-3 people
Skills Needed: Django, HTML/CSS, Bootstrap, Markdown (optional)
Impact: Enable easy creation of new pages without developer intervention
GitHub Issue: #118

Description

Create a flexible system to easily add new pages to the site (like /nominations2025) and establish a reusable pattern for future page additions. This addresses the recurring need to add standalone pages for committee work, announcements, and special initiatives.

Tasks

- Create nominations2025 page immediately (quick win)

- Design flexible URL routing pattern for dynamic pages
- Build admin interface for creating/editing standalone pages
- Implement Markdown or WYSIWYG editor for page content
- Create reusable page templates (basic, form-based, content-rich)
- Add SEO fields (title, description, meta tags)
- Implement page visibility controls (draft/published)
- Document the process for non-technical users

Workshop Alignment

Supports efficient committee operations and reduces technical barriers for content creation

Technical Notes

- Create new Django app `pages` or add to existing `home` app
- Option 1 (Simple): Create view in `home/views.py` for nominations page
- Option 2 (Scalable): Build `pages/models.py` with Page model (title, slug, content, template_type)
- Add URL pattern with slug: `path('<slug:slug>/', views.page_detail)`
- Consider FlatPages app as alternative: `django.contrib.flatpages`
- Support Markdown with `django-markdownx` or rich text with TinyMCE
- Create reusable templates in `pages/templates/pages/`

Success Criteria

- [ ] Nominations2025 page live and functional
- [ ] Admin can create new pages without code changes
- [ ] Multiple page templates available
- [ ] Markdown or WYSIWYG editor working
- [ ] SEO fields configurable
- [ ] Documentation for page creation process
- [ ] Mobile responsive
- [ ] Pattern established for future pages

🎯 Medium Priority

These initiatives add valuable functionality and improve user experience.

6. Event Waitlist/Queue System

Group Size: 2-3 people
Skills Needed: Django models, views, forms, email tasks
Impact: Better event capacity management

Description

Add waitlist functionality when events reach capacity, automatically promoting users when spots become available.

Tasks

- Extend `EventRegistration` model for waitlist status
- Add waitlist signup button on sold-out events
- Implement automatic promotion when spots open
- Create email notifications for waitlist status changes
- Build admin interface for waitlist management
- Implement FIFO promotion logic
- Add waitlist count display on event pages

Workshop Alignment

From Kasper's feedback: "Users should be able to join queue for fully booked events"

Technical Notes

- Modify `events/models.py` EventRegistration model
- Add new status: `('Waitlisted', 'Waitlisted')`
- Create signal in `events/models.py` for cancellation → promotion
- Add Celery task in `events/tasks.py` for notifications
- Update `events/views.py` for waitlist registration
- Add queryset methods in `events/managers.py`

Success Criteria

- [ ] Waitlist button appears when sold out
- [ ] Users can join waitlist
- [ ] Automatic promotion works
- [ ] Email notifications sent
- [ ] Admin can view/manage waitlist

- [ ] Waitlist count displayed

7. Recurring Mini-Meetups System

Group Size: 2-3 people
Skills Needed: Django models, forms, basic templating
Impact: Simplify hosting informal community meetups

Description

Create a streamlined system for organizers to easily set up recurring mini-meetups (casual bar meetups, coffee chats, etc.) with minimal overhead. Focus on simplicity: just location, time, and recurrence pattern.

Tasks

- Extend Event model with recurrence fields (weekly, monthly, etc.)
- Create simple "Mini-Meetup" event template (just venue, time, optional capacity)
- Build easy creation form for recurring events
- Auto-generate future event instances based on recurrence rule
- Add "Mini-Meetups" category/tag for easy filtering
- Create mini-meetup landing page showing all casual meetups
- Allow organizers to pause/resume recurring series
- Simple RSVP system (no payment, just count)

Workshop Alignment

"Build regional communities and engagement beyond Copenhagen" - Enable easy informal meetups in any location

Technical Notes

- Add fields to `events/models.py`: `is_recurring`, `recurrence_pattern`, `parent_event`
- Consider using `django-recurrence` library or simple choice field (weekly/biweekly/monthly)
- Create simplified mini-meetup form in `events/forms.py`
- Add management command to generate recurring instances
- Keep it simple: venue name, address, time, optional RSVP limit
- No payment integration needed for mini-meetups

Success Criteria

- [ ] Recurring event creation form working
- [ ] Auto-generation of future instances
- [ ] Mini-meetup template simplified
- [ ] Mini-meetups landing page created
- [ ] Organizers can manage recurrence
- [ ] Simple RSVP without payment
- [ ] Mobile responsive

8. Open Source Project Showcase

Group Size: 2-3 people
Skills Needed: Django, Markdown, GitHub API (optional)
Impact: Motivation driver from workshop

Description

Create a dedicated section to showcase DDSC's open source projects and encourage contributions.

Tasks

- Create new Django app `projects`
- Design project model (title, description, tech stack, repo URL)
- Build project listing page
- Implement "Project of the Month" feature
- Add project detail pages with contribution guidelines
- GitHub integration for stats (stars, forks, contributors)
- Call-to-action for contributions
- Admin interface for managing projects

Workshop Alignment

"Highlight the Open Source project to motivate involvement – a topic of interest among the community"

Technical Notes

- Create new app: `python manage.py startapp projects`
- Create models in `projects/models.py`

- Add URL patterns in `projects/urls.py`
- Create templates in `projects/templates/projects/`
- Optional: Use GitHub API for live stats
- Add to main navigation in `home/header_navbar.html`

Success Criteria

- [ ] Projects app created
- [ ] Project listing page working
- [ ] Project detail pages
- [ ] "Project of the Month" feature
- [ ] GitHub integration (optional)
- [ ] Contribution guidelines included
- [ ] Admin interface functional

9. Member Directory & Networking

Group Size: 2-3 people

Skills Needed: Django, Bootstrap, privacy controls

Impact: Foster community connections

Description

Create an opt-in member directory to facilitate networking and connections within the community.

Tasks

- Add profile visibility settings to Member model
- Build searchable member directory page
- Create public profile pages with job title, interests, bio
- Implement privacy controls (public/private profiles)
- Add "Connect" feature or LinkedIn integration
- Filter by skills, location, role
- Highlight board members
- Profile completion incentives

Workshop Alignment

"Network, events and mini-meetups as a motivator to be engaged"

Technical Notes

- Extend `members/models.py` Member model
- Add fields: `bio`, `interests`, `linkedin_url`, `visibility`
- Create directory view in `members/views.py`
- Add search/filter form in `members/forms.py`
- Create templates in `members/templates/members/`
- Ensure privacy controls respect GDPR

Success Criteria

- [ ] Member directory page created
- [ ] Privacy settings working
- [ ] Search and filter functional
- [ ] Public profile pages
- [ ] Board members highlighted
- [ ] LinkedIn integration (optional)
- [ ] Mobile responsive

10. Event Reminders & Communication

Group Size: 2-3 people

Skills Needed: Celery, Django signals, email templates

Impact: Reduce no-shows, improve engagement

Description

Implement automated reminder and communication system for event attendees.

Tasks

- Create automated reminder emails (1 week, 1 day before event)
- Implement event update/changes notification system
- Build post-event follow-up email system
- Add Slack integration for event announcements
- Create admin interface to send custom messages to attendees
- Track email open rates (optional)
- SMS reminders (future: Twilio integration)

Workshop Alignment

Event managers want to send reminders to registered attendees

Technical Notes

- Use Celery Beat for scheduled tasks
- Create tasks in `events/tasks.py`
- Use Django signals in `events/models.py` for event changes
- Create email templates in `events/templates/events/emails/`
- Add Celery Beat schedule in `ddsc_web/settings/settings.py`
- Use existing email infrastructure

Success Criteria

- [ ] Reminder emails sent automatically
- [ ] Event update notifications working
- [ ] Post-event follow-up emails
- [ ] Admin can send custom messages
- [ ] Slack integration (optional)
- [ ] No-show rate reduced

💡 Nice-to-Have

These initiatives are great stretch goals if teams finish early or want smaller projects.

11. Flexible Engagement Options

Group Size: 2-3 people

Skills Needed: Content writing, UX design, lightweight dev

Impact: "Create flexible ways to engage with DDSC"

Description

Create content and features that allow members to contribute based on their available time.

Tasks

- Design "Ways to Contribute" landing page
- Create volunteer opportunities listing

- Compile micro-contribution ideas (< 1 hour tasks)
- Design recognition system (contributor badges)
- Add time commitment indicators to opportunities
- Create contribution tracking (optional)

Workshop Alignment

"Finding time to engage in the community – Creating flexible ways to engage"

Success Criteria

- ☐ Landing page created
- ☐ Volunteer opportunities listed
- ☐ Micro-contributions identified
- ☐ Recognition system designed
- ☐ Time indicators shown

12. A100 GPU Resource Pool System

Group Size: 2-3 people
Skills Needed: Django, forms, admin customization
Impact: Showcase community benefits

Description

Create a system for community members to request and manage access to DDSC's A100 GPU resources.

Tasks

- Design resource request form
- Create usage guidelines page
- Build application review workflow for admins
- Display usage statistics/showcase
- Add success stories from A100 users
- Track resource allocation
- Notification system for approvals

Workshop Alignment

"a100, hvordan? Og til hvad? Måske resource pool man kan søge" (How to access A100? Maybe a resource pool to apply to)

Technical Notes

- Create new app `resources` or add to existing app
- Create RequestForm model and views
- Build admin approval interface
- Use existing email notification system
- Consider integration with actual GPU allocation system

Success Criteria

- [ ] Request form created
- [ ] Usage guidelines documented
- [ ] Admin workflow functional
- [ ] Statistics displayed
- [ ] Success stories showcased
- [ ] Notifications working

13. Mobile App Preparation - API Development

Group Size: 2-3 people
Skills Needed: Django REST Framework, JWT, API design
Impact: Future mobile app support

Description

Build RESTful API endpoints to support future mobile app development.

Tasks

- Install and configure Django REST Framework
- Create API endpoints for events (list, detail, register)
- Create API endpoints for user profiles
- Implement JWT authentication
- Build API documentation (Swagger/OpenAPI)
- Add rate limiting

- Implement API versioning
- Write API tests

Technical Notes

- Add `djangorestframework` to requirements.txt
- Create `api` app or add to existing apps
- Create serializers for models
- Add API URLs to `ddsc_web/urls.py`
- Use `djangorestframework-simplejwt` for auth
- Document with `drf-yasg` or `drf-spectacular`

Success Criteria

- [ ] REST API endpoints created
- [ ] JWT authentication working
- [ ] API documentation generated
- [ ] Rate limiting implemented
- [ ] API versioned
- [ ] Tests written

14. Automated Testing & CI/CD Improvements

Group Size: 2-3 people

Skills Needed: Python testing, GitHub Actions, Docker

Impact: Code quality, reliability

Description

Improve test coverage and automated deployment processes.

Tasks

- Write unit tests to increase coverage from 60% to 80%
- Create integration tests for critical flows
- Add E2E tests with Selenium
- Improve GitHub Actions workflow
- Add automated deployment to staging/production
- Implement pre-commit hooks

- Add code quality gates
- Create test documentation

Technical Notes

- Use Django's built-in test framework
- Add coverage reporting with `coverage.py`
- Use `pytest` and `pytest-django` (optional)
- Update `.github/workflows/django.yml`
- Add pre-commit config `.pre-commit-config.yaml`
- Use existing CI structure

Success Criteria

- [ ] Test coverage > 80%
- [ ] Integration tests written
- [ ] E2E tests implemented
- [ ] Automated deployment working
- [ ] Pre-commit hooks active
- [ ] Code quality gates pass

15. Event Analytics Dashboard

Group Size: 2-3 people
Skills Needed: Django, Chart.js, SQL, data visualization
Impact: Data-driven event planning

Description

Create analytics dashboard for board members to track event performance and member engagement.

Tasks

- Build event attendance trends visualization
- Calculate and display no-show rate analysis
- Create popular event topics chart
- Add geographic distribution map
- Display member engagement metrics

- Implement export reports (CSV, PDF)
- Add date range filters
- Create admin-only access controls

Workshop Alignment

Better understanding of community engagement patterns

Technical Notes

- Extend existing `stats` app
- Add views in `stats/views.py`
- Use Chart.js (already in stack)
- Create complex queries in `stats/queries.py`
- Use Django's aggregation functions
- Add caching for expensive queries
- Restrict access to board members only

Success Criteria

- [ ] Analytics dashboard created
- [ ] Attendance trends shown
- [ ] No-show rate calculated
- [ ] Topic popularity displayed
- [ ] Geographic distribution shown
- [ ] Export functionality working
- [ ] Admin-only access enforced

📚 Resources & Support

Need Help?

- 💬 **Slack**: #hackathon-nov-2025 - Ask questions, share progress, get unstuck
- 💻 **GitHub Repository**: ddsc-website - Browse code, check patterns
- 🐛 **Report Issues**: Create an issue
- 📖 **Documentation**: Product/Technical Requirements in the repo
- 🤝 **Team**: Ask fellow participants - everyone's here to help!

Quick Setup Commands

Good luck and happy hacking! 🚀

Remember: The goal isn't perfection - it's learning, building together, and having fun with vibe coding!