AGENDA

01 Why make projects?

02 Projects “Archetypes”

03 Making a complete project

04 What to avoid...

05 How do you learn new technologies
WHY MAKE PROJECTS?
# Goals of Making a Project

<table>
<thead>
<tr>
<th>Learning</th>
<th>Career</th>
<th>Fun</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning new languages, tools, design patterns, niches</td>
<td>Projects - best way to get experience applicable to a niche</td>
<td>Explore different aspects of computer science that interest you!</td>
</tr>
<tr>
<td></td>
<td></td>
<td>🌟 Especially important for people with no prior experience</td>
</tr>
</tbody>
</table>
Project Forms

Applications
→ Web
→ Fullstack, Front-end, Back-end
→ Mobile

Competitions
→ Kaggle - Data Science
→ Cybersecurity - Bug Bounties, CTFs
→ Hackathons
PROJECT
"ARCHETYPES"
## Project Archetypes

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Generic Project</strong></td>
<td>→ Learn new technologies that are applicable to the jobs/niches you want to be in</td>
</tr>
<tr>
<td><strong>Make Something You Need</strong></td>
<td>→ Useful after you complete the project</td>
</tr>
<tr>
<td></td>
<td>→ Narrative - impress recruiters</td>
</tr>
<tr>
<td></td>
<td>→ More passion = more results (motivation)</td>
</tr>
<tr>
<td><strong>Make Something for Others</strong></td>
<td>→ Same benefits as above - equally or more impressive to recruiters</td>
</tr>
<tr>
<td><strong>Open-Source Contributions</strong></td>
<td>→ Check GitHub for “good first issue”</td>
</tr>
<tr>
<td></td>
<td>→ Get experience working with old and existing code</td>
</tr>
</tbody>
</table>
GitHub “Good First Issue”

Link: https://github.com/topics/good-first-issue
MAKE A COMPLETE PROJECT
Checklist

- Does this have a purpose for yourself or others?
- Do you use a tech stack?
  - Front-end, back-end, database, etc.
- Did you use the right technologies?
  - Most common and applicable technologies - might help with job applications
  - Good for this use-case?
  - Can you justify their use?
- Do your projects have tests?
  - Uncommon for students, and even new grads
  - Essential and valuable skill
- Did you comment and document your code?
  - Write a nicely formatted README (installation, purpose, use-cases, “getting started”)
  - Write self-documenting code, use comments where code becomes potentially confusing
- Do your projects have real users/clients?
  - How did you deploy and architect your project
  - Shows recruiters that you can build real applications with purpose
WHAT TO AVOID...
Avoid Duplicate Work

- Don’t reimplement different features throughout your different projects
  Make a library for this!

- Don’t make things that are functionally the same
  To-do list = Task tracker = Weather app
Scope Creep

→ Draft a plan on how to make this project
  Break your project into small, manageable tasks that you can learn and implement

→ Do not keep adding features or else nothing will get done, and the project will be disorganized
  Focus - completing core parts first
Finding Technologies To Include...
Web Portal Developer Co-op/Intern
Nokia - Kanata, ON (Hybrid)

Responsibilities

As part of the team, you will:

- Develop various features of the Developer Portal using Python/Django
- Take responsibility to write compelling, scalable code
- Code, test and debug new or existing features of the portal
- Work closely with other developers and actively participate in portal-related discussions
- Working in a fast-paced environment with an excellent team of engineers

Required Skills

You have:

- Strong programming skills in Python (Experience with Django Framework preferred)
- Some Experience with web development (HTML, CSS and JavaScript).
- Familiarity with Unix (RHEL, Ubuntu, CentOS).

It would be nice if you also had:

- Experience with Git version control system and/or micro-services build architecture (Gradle).
- Strong problem-resolution skills; ability to learn quickly.
- Strong grasp of object-oriented software design concepts
- Knowledge of agile software design and test methodologies.
Software Developer Co-op (Summer 2024)  
Intuit • Toronto, ON (On-site)

What you'll bring

- Qualifications
  - You’re enrolled in a post-secondary degree related to Computer Science/Engineering
  - You’re a lover of all things development
  - You may have some experience with: Java, Springboot, JavaScript, React, Python
  - You bring a passion for learning, exploration, & innovation
  - Nice to have: experience and passion for building Conversation AI
How to Learn New Technologies?

- Learning new technologies gets easier - as you have more experience with others, since there is some overlap
- There are infinite resources available online, such as documentation, wikis, tutorials, etc
  - Knowing how to search documentation and recognizing patterns between languages
- Follow some “get started” guides (on YouTube or in documentation)
- Apply the new knowledge in a project
Resources

**Frontend**
- General: https://www.w3schools.com/
- React: https://react.dev/
- CSS Game to Learn Flexbox: https://flexboxfroggy.com/

**Backend**
- C++: https://learn.microsoft.com/en-us/cpp
- C: https://devdocs.io/c/
- Java: https://docs.oracle.com/en/java/

**Databases**
- Django: https://www.djangoproject.com/
- SQL: https://www.w3schools.com/sql/sql_quickref.asp
- MongoDB: https://www.mongodb.com/docs/

**Cloud Deployment**
- GitHub: https://docs.github.com/en
- AWS: https://docs.aws.amazon.com/
THANKS!

Do you have any questions?

CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon, and infographics & images by Freepik
ICONS PACK: BUSINESS MEETING