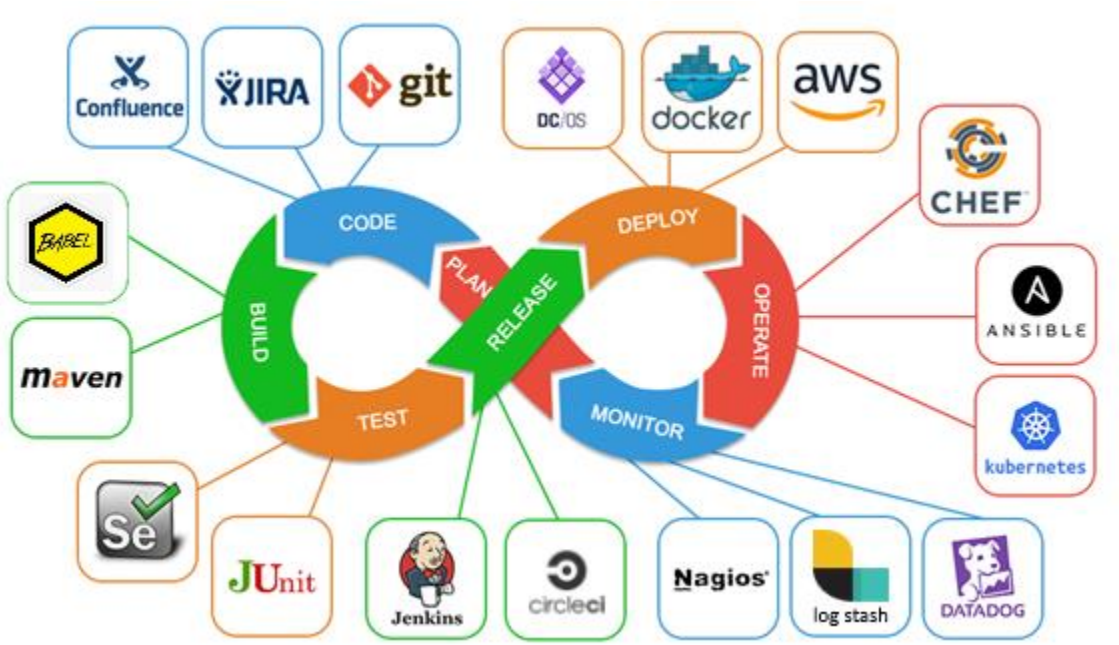
סילבוס לקורס DevOps



למה DevOps?

DevOps הוא אחד התחומים אם לא התחום החם ביותר בשוק התוכנה. אם עד לפני כמה שנים כולם רצו להיות מפתחי אפליקציות ומערכות מבוססות אינטרנט (Web) כיום DevOps הוא התחום שחסרים בו הכי הרבה מתכנתים ומהנדסים ולכן גם המשכורת הן מן הגבוהות בשוק גם למפתחים חסרי ניסיון. חיפוש מהיר בגוגל על רמות השכר ימחיש נקודה זו ועל כן החלטנו לתת מענה לצורך ההולך וגדל במפתחי DevOps ולפתוח מסלול לימודים מקיף לתחום.

אז מה זה בכלל DevOps?

עד לפני מספר שנים כל ארגון היה בנוי ממחלקת פיתוח ומבמקביל מחלקות של תמיכה ושרותי IT, בשוק של ימינו שהכל חייב להיות מעכשיו לעכשיו ואם אפשר כבר אתמול נולד והתפתח תחום הDevOps. בבסיסו של הרעיון עומד הצורך להגיב מהר וביעילות לשינויים, לכתוב תוכנות יעילות יותר ולשווקם מהר יותר ובאיכות מקסימלית. ופה כאמור נולד הDevOps. משמעות השם DevOps היא קיצור של Development & Operation כלומר שילוב של פיתוח ותפעול יחדיו. התפיסה היא שאנשי DevOps לוקחים אחריות ומנהלים את כל תהליך שחרור המוצר מרגע סיום הפיתוח. אם בעבר היו מספר מחלקות שונות כגון פיתוח, בדיקות, אוטומציה, שחרור גירסאות, ניהול בסיס נתונים וכולי כיום אנשי הDevOps עושים הכל מהכל.

בבסיסו של הרעיון עומדת ההנחה שבמקום לפצל את התהליך הכולל בתוכו מגוון רחב של נושאים כגון: בניה, בדיקות שונות, אריזה, שחרור גירסה, העלאה לאוויר, ניטור ועוד נרכז את הכל בתוך צוות מקביל שעובד יחד עם צוותי הפיתוח, והלכה למעשה אנשי הDevOps לוקחים אחריות ומתפעלים את חיי המוצר מרגע סיום הפיתוח עד לעליה לאוויר על כל המשתמע מכך.

נגזרת של הדבר היא שמפתח DevOps צריך לדעת לעומק המון תחומים וכלים על מנת לבצע את עבודתו בצורה יעילה. ישנם ארגונים שבהם אנשי DevOps מתפעלים ומתחזקים מעל 50!!! מערכות וכלים שונים במקביל.

זו גם הסיבה שבתחום הזה חסרים מפתחים רבים והמשכורות בזינוק מתמיד, לא כל אחד יכול ומסוגל לתפעל ולהכיר מגוון כזה רחב של מערכות תוך הבנה ויכולת להבין כל מערכת, כלי ושפה ולשבצה בקובייה המתאימה עבורה. התפקיד דורש הבנה וידע ובנוסף יכולת להיות יצירתי, לתפעל תקלות ולחשוב מחוץ לקופסא על מנת לתחזק ולייעל את הכלים ואת התהליכים המרובים.

למי מתאים הקורס?

בתור אנשי DevOps עם לא מעט שנות ניסיון כמפתחים, ראשי צוותים וארכיטקטים בנינו עבורכם את הקורס בצורה מודולורית ככזה שיתאים למגוון רחב של אנשים.

הקורס מתאים למתכנתים, לאנשי QA שרוצים להכנס לתוך עולם הפיתוח וגם לאלו שמעולם לא כתבו ולו שורת קוד אחת מעולם ובטח ובטח שלא יודעים אפילו מה זה Unix.

על ממת למקסם את היכולת להצליח ולהשתלב בתעשייה מיד בסיום הקורס, בנינו את הקורס בצורה של מודולים, שכל מודול הוא למעשה עצמאי ועומד בפני עצמו. כמובן שבפועל כל היחידות מתחברות יחד אבל היות וישנו מגוון כל כך רחב של תחומים וכלים בסופו של יום כל אחד בוחר לצמו מספר מצומצם של כלים שבהם הוא מתמחה לעומק וזו הופכת להיות המומחיות שלו, במקביל הוא מכיר ומתפעל את כל שאר הכלים והטכנולוגיות אך ברמה פחות עמוקה, ועל כן כאמור הקורס נבנה בצורה כזו שמאפשרת לכל אחד לבחור את התחום שיותר מעניין אותו להמתמקד בו לצד לימוד ותרגול של כל הכלים הרלוונטים.

איך הופכים מחסרי ידע לNinja בתחום?

מעבר וקריאה של משרות דרושים משאיר את הרושם שצריך להיות מחשב אנושי כדי לעמוד בכל התנאים אך בפועל זה הרבה יותר פשוט עם ההכוונה והמיקוד הנכון.

התנאי היחידי להצלחה בקורס הוא
הרצון והיכולת להתמיד לתרגל ולחזור על החומר

רוב אנשים הDevOps שנמצאים היום בשוק החלו את דרכם כבודקי תוכנה, מפתחים, מנהלי מוצר וכו. רובם הגדול לא כתב קוד מעולם והמשותף לכולם זה פשוט היכולת להיות עם ראש פתוח ומוכנות ללמוד ולהשקיע מתוך הבנה שניתן להגיע רחוק ויחסית מהר בתחום הזה.

תוכנית ההכשרה

תוכנית ההכשרה בצורה של יחידות לימוד. היחידות בנויות בצורה של לגו כאשר רוב היחידות דורשות ידע מוקדם שנלמד ונרכש ביחידות שנלמדו לפניה.

כאמור כל יחידה יכולה להלמד ולעמוק בפני עצמה אך בפועל אנחנו נשלב את כל הנושאים שילמדו יחד כמקשה אחת כשכל יחידה מתחברת לפאזל במיקום ובייעוד שלה ועדיין כאמור כל יחידה ניתנת ללימוד בנפרד משאר היחידות .

חלק מעשי

הייחודיות שלנו הוא בחלק המעשי, אנחנו לא מלמדים תאוריה, כחלק מההכשרה בנינו תוכנית ליווי שבהמלכה אנחנו מתרגלים לא מעט את החומר הנלמד ולא מתרכזים בתאוריה בלבד.

הסטודנטים זוכים לליווי שוטף וצמוד ומתנסים בעבודה מעשית ופרקטית. חלק זה הינו חלק בלתי נפרד מתהליך ההכשרה ובעינינו זה השלב החשוב ביותר שבו חווים ממקור ראשון את העבודה בפועל. התרגול לא פחות חשוב ואף יותר מהחלק התאורטי ועל כן תופס חלק נכבד ממערך השיעורים.

יחידות הלימוד בקורס

כאמור כל יחידה יכולה להילמד ולעמוד בפני עצמה אך בפועל אנחנו נשלב את כל הנושאים שילמדו יחד כמקשה אחת כשכל יחידה מתחברת לפאזל במיקום ובייעוד שלה.

יחידות הקורס:

יחידה	דרישות קדם	תאוריה	תרגול	סכ"ה שעות ליחידה
שיעור פתיחה		1 מפגשים		4 שעות
DevOps Intro		2 מפגשים		8 שעות
Linux Fundamentals		4 מפגשים	1 מפגשים	20 שעות
Bash Scripting	Linux	2 מפגשים	1 מפגשים	12 שעות
Python		4 מפגשים	2 מפגשים	24 שעות
Git		4 מפגשים	1 מפגשים	20 שעות
Docker	Linux / git	4 מפגשים	1 מפגשים	20 שעות
Kubernetes	Docker	4 מפגשים	1 מפגשים	20 שעות
Jenkins		2 מפגשים	1 מפגשים	12 שעות
Chef	Kubernetes	2 מפגשים	1 מפגשים	12 שעות
Azure		4 מפגשים	2 מפגשים	24 שעות
Project	כל היחידות		4 מפגשים	16 שעות
Total		33 מפגשים	15 מפגשים	48 מפגשים / 192 שעות לימוד

הערות:

- יתכן ויתווספו יחידות נוספות לקורס (למשל ELK, Agile, Jira, AWS) ובמקרה ואכן יפתחו מודולים נוספים, תלמידי הקורס יקבלו קדימות ועדיפות על פני אחרים.

- כל יחידת לימוד מלווה בתרגול מעשי ובסוף הקורס המטרה היא לבנות פרוייקט מקיף שמכסה את כל הידע הנצבר בכל יחידות הלימוד השונות החל מהגדרת המשימה עד שלב העלייה ליצור

DevOps

When development meet production

תוכן יחידות הלימוד:

DevOps Intro

8 hours

- What is DevOps
- Agile In nutshell
- Agile vs Waterfall
- DevOps in Nutshell
- Concepts
- CAMS / CALMS - (Culture, Automation, Measurement, Sharing) principles.
- Dev & Ops
  - Automation
  - CI/CD (continues integration/delivery/deploy)
  - DevOps lifecycle
- Testing's
- Release management
- Application Performance Management (APM)
- Best practice
- Tools
  - Git
  - Jenkins
  - Jira
  - Docker
  - Kubernetes
  - Chef
  - Puppet
  - Ansible
  - Nexus
  - Artifactory
  - Sonar Cube
  - Nagios

Linux Fundamentals

16 hours / 4 Lab hours

- What is Linux/Unix
- Differences between windows to Unix
- Basic Linux/Unix commands
  - ls
  - grep
  - chmod
  - chown
  - ps
  - ssh / scp
  - mkdir
  - aliases
  - sed / awk
  - top
- Installing Unix
  - Virtual Box
- Unix file system
  - Permissions
  - chmod/ chown
  - View content of folder / search
  - Input / output (stdout, stderr)
  - symlinks
- Basic configuration
  - Environment variables
- Installing packages
- What is terminal
- Working with vi/vim
- Crontab
- Bash
  - Commands
  - Shells
  - Conditions
  - Loops
  - Functions
  - Pipes
- SSH
  - SCP
  - Putty

Bash Scripting

8 hours / 4 Lab hours

- Working with terminal
- Commands (some commands are review from previous unit)
  - ls
  - grep
  - chmod
  - chown
  - ssh / scp
  - mkdir
  - sed / awk
  - chmod/ chown
  - touch
- Write bash script
  - Set script shell
  - Variables
  - Conditions
  - Loops
  - Sleep
  - Switch/Case
  - Function
  - Exit status / return codes
  - Logic Conditions
- Logging
- Working with files
- Zip / Tar/ Compress / uncompress
- Process
  - ps
  - kill
  - top
- RegExp basics

Python

16 hours / 4 Lab hours

- Setup python
- Python Basics
  - Syntax
  - Variables
- Variables & Data structures
  - Global variables
  - Numbers integer/float
  - Strings
  - User input
  - List
  - Range
  - Tuples
  - Dictionaries / JSON
  - sets
- Iterators / Enumerable
- Flow control
  - If/else
  - For
  - While
  - Pass / continue / break
  - Try/catch
- Functions
  - Def
  - Default arguments
- Modules / Packages
- Classes
  - Getters / setters
  - Inheritance
  - Subclasses
  - Data attributes & properties
- Input & output
  - Files
  - Pickle
  - shelve
- Micro-framework
  - Flask (web server)
- Generators / Decorators
  - Next
  - Range
  - Yield
  - Map

Git

16 hours / 4 Lab hours

- What is SCM
- 3-states
- gitconfig
- gitignore
- gitkeep
- aliases
- what is commit
- Git LFS
- Commands
  - init
  - clone
  - add
  - rm
  - commit
  - status
  - log
  - checkout
- Branches
  - What are branches
  - Working with branches
  - HEAD/ Detached HEAD
  - checkout
  - fetch
  - merge
  - pull / push
- Merges
  - ff
  - no-ff
  - rebase
- bisect
- stash
- squash
- submodule / subtree
- rerere
- tags
- notes
- cherry-pick
- reflog
- worktree
- Pull requests
- Git hooks

Docker

16 hours / 4 Lab hours

- What is Docker
- Installing Docker
- Docker introduction
- Images / VM / Containers
- CGroups
- Union File system (AUFS)
- Docker container lifecycle
- Container
  - What is it
  - Using containers
  - Working with containers
- Networking
  - Network concepts
  - Private / Public network
  - Virtual networks
  - How containers discover themselves
- Containers / images
  - What is image
  - Building images
  - Using images
  - Pull images from Docker hub
  - Container volumes / Persistence
- Docker CLI
  - Build
  - Run
    - Background / detached / foreground
    - Expose ports for communication
  - Commit
  - Pull
  - Push
  - Diff
  - Tags

- Docker file
  - From
  - Run
  - CMD
  - Expose
  - Env
  - Add /Copy
  - Volume
  - Entrypoint
  - Workdir
- Docker compose
  - Compose.yml
- Docker swarm / Clusters
  - Build clusters – multi nodes app
  - Manage clusters
  - Life cycle
- Docker registry
- Tools
  - Kitematic

Kubernetes

16 hours / 4 Lab hours

- What is K8s?
- Introduction
- Introduction to Container Orchestration
  - Kubernetes Core Concepts
  - Kubernetes Architecture
- Getting Started with Kubernetes
  - Minikube Setup
  - Kubectl
- Config files
- Pods
- Scaling Kubernetes
- Deploying
  - Deploy multiple revisions
  - Replica (sets, controller etc)
  - Update / upgrade existing container
- Labels & Selectors
- Health Checking
- Web Interface / WebUI
- DNS & Service Discovery
- Volumes
- Secrets
- Auto-Scaling
- Auditing
- High Availability
- Services
- ConfigMap
- Microservices
- API
- Helm (optional)
- Ingress (optional)
- Clusters

Jenkins

8 hours / 4 Lab hours

- What is Jenkins
- Agile intro
  - What is CI/CD
- Installing Jenkins
- Jenkins Architecture
  - Slave / master
  - Workspaces
- Managing Jenkins
  - Dashboards
  - Managing plugins
  - Jobs
  - Configuration
- Blue Ocean
- Managing users
  - User roles
  - Project roles
  - Security
- CI (Continues integration)
- CD (Continues delivery)
- CD (Continues deployment)
- Jobs
  - What is Job
  - Job phases
  - Connect job to Source control
  - Build
    - Build phases / Build triggers
  - Schedule builds
  - Parallel builds
- Code quality
- Pipeline as code
  - Build pipelines
  - Jenkins files
  - Full automation
- Jenkins and Docker
  - Containers
  - Docker hub
  - Docker file
- Distributed builds
  - Nodes
  - Agents

Chef

8 hours / 4 Lab hours

- Chef for configuration management
- Overview of Chef
  - Common Chef Terminology (Server,
  - Workstation, Client, Repository etc.)
  - Servers and Nodes
  - Chef Configuration Concepts
- Workstation Setup
  - How to configure knife
  - Execute some commands to test connection between knife and workstation
- Organization Setup
  - Create organization
  - Add yourself and node to organization
- Test Node Setup
  - Create a server and add to organization
  - Check node details using knife
- Node Objects and Search
  - How to Add Run list to Node
  - Check node Details
- Environments
  - How to create Environments
  - Add servers to environments
- Roles
  - Create roles
  - Add Roles to organization
- Attributes
  - Understanding of Attributes
  - Creating Custom Attributes
  - Defining in Cookbooks
- Data bags
  - Understanding the data bags
  - Creating and managing the data bags
  - Creating the data bags using CLI and Chef Console
  - Sample data bags for Creating Users.

Azure

16 hours / 8 Lab hours

- Introduction to cloud computing
- Azure portal
- Subscriptions / tenants / Resource group
- az / Azure cli
  - Deep dive into scripting and automating Azure
- ARM
- Virtual machines
- Users /Roles
- Scale sets / Load Balancers
- DB
- Storage
  - Blob
  - Files
  - Access storage
  - Storage Explorer
- Networks
  - Pip
  - VPN
  - DNS
  - NSG
  - Traffic manager
  - RDP
- Key-vaults
- Databases
- Power shell
- Automation
- Monitoring
- Alerts / Metrics
- Applications
- Function App
- Dashboards
- Azure AD
- Application insights
- Tools
  - resources.azure.com
  - Azure Storage Explorer