Developing at Uber Scale
Go anywhere
Get anything

Uber’s mission is to reimagine the way the world moves for the better.
Uber’s global scale

- **70+ countries**
- **10,000+ cities**
- **25 million+ trips a day**
- **137 million+ monthly active customers**

**GROSS BOOKINGS ($B)**

- **Q4 2023**: $33.6B
- **Gross Bookings Growth YoY**: 18%

### Mobility

### Delivery

### Freight
Global tech platform at massive scale

5,000+ microservices
1.8 MILLION CPU cores allocated to microservices
1 EB Stateful volumes
110 PB online storage
350 PB big data storage

15 Mobile apps
2B User sessions/month
2T Mobile events/month
Development at Uber scale

- 4,500+ Engineers
- 65k Code commits/month
- 100+M Monorepo LOC
- 10+ Global engineering offices
- 104k Production deployments/month
- 7 Programming languages

Uber
Challenges @ scale

- Velocity
- Quality
- Cost
- Efficiency
- Security
- Compliance
- Safety
- Privacy

Uber
Our developer platform strategy

- Fast, modern tools
- Powerful frameworks
Our developer platform strategy

- Fast, modern tools
- Powerful frameworks
- Safety nets
Our developer platform strategy

- Fast, modern tools
- Powerful frameworks
- Safety nets
- Standardization
Our developer platform strategy

- Fast, modern tools
- Powerful frameworks
- Safety nets
- Standardization
- Measure what matters

VELOCITY
QUALITY
EFFICIENCY
ITERATION
Our developer platform strategy

- Fast, modern tools
- Powerful frameworks
- Safety nets
- Standardization
- Measure what matters
- Artificial intelligence

QUALITY

EFFICIENCY

ITERATION

TRANSFORMATION
Standardized development

Per platform monorepos & frameworks
Platform teams centrally improve tools and frameworks, and support all engineers

Standardized development environment
Developers move quickly using standardized cross-platform tools and frameworks

Uber
Standardized developer workflow

**Design**
- PRD
- ERD

**Develop**
- Local development
- CI build
- Code review
- Submit Queue
- Continuous Delivery

**Release**
- Mobile App
  - Wait for build cut
  - Mobile test & dogfooding
  - App store rollouts

**Manage**
- Production management

**Service**
- Staging
  - Service rollout: Canary + Production

**Other**
- Test failure
- Build or test failure
- Changes requested
- Build, test or merge failure
- Build failure
- Reproduce, root cause and fix

- Mobile hotfix
- Bugs
- Crashes
- Incidents
- Monitoring
- Crashes
- Build
- Test failure
- Build, test or merge failure
- Build failure
- Reproduce, root cause and fix

- Bugs
- Crashes
Developer workflow optimization

Reduce Latencies
Speed up tools & automate workflows

Reduce impact of failures
Shift left, increase SNR & improve debugging

Improve productivity
Increase uninterrupted focus time and team autonomy

Mobile app
- Wait for build cut
- Mobile test & dogfooding
- App store rollouts

Service
- Staging
- Service rollout: Canary + Production

Uber
- Build or test failure
- Changes requested
- Build, test or merge failure
- Build failure

Reproduce, root cause and fix
Developer Satisfaction (NPS)

NPS QUESTION
On a scale of 0 to 10, how likely are you to recommend Uber’s development environment to other engineers?

Top themes
- Quality
  - Testing
  - Debugging

Daily dev experience
- Tool speed & reliability
- IDEs
Our experience using code assistants

Internal survey results

- Effective for simple boilerplate code and tasks
  Copilot is really useful for writing 1-2 lines of straightforward / repetitive code

- Code suggestions frequently need further editing
  It frequently has small misses. I find myself usually double-checking Copilot's work for longer than it would have taken me to just write that work myself.

29% Acceptance rate
1.4 LOC per accepted code
75% Feel more productive
6.0 NPS score for Copilot
Uber Code Assistant

IDE context → IDE

Code suggestions → Fine tuned LLM

Based on open-source Codellama

Uber code base

Apple, Android, JavaScript, Go, Java, Python
Uber tech-wide Hackdays

- 700+ Participants
- 100 Projects demoed
- 6 Sites around the globe
- 3 Broad categories

Product experience
Developer productivity
Business operations

Uber
Impact of errors in code

Sorry, that extra $35 you got in your Uber account was a mistake

Clint Henderson
Dec. 26, 2021 • 3 min read

Incident in 2021 due to MisusedWeekYear
“Week year” is intended to be used for week dates, e.g. “2015-W01-1”, but is often mistakenly used for calendar dates, e.g. 2014-12-29

@SuppressWarnings("MisusedWeekYear")

private static final DateTimeFormatter IDEMPOTENCY_DF =
DateTimeFormatter.ofPattern("MM_YYYY");
Automatically fixing errors in code

Code → Code analysis → Large Language Model → Fixed Code

- 3.5K Errors auto-fixed / week*
- 875 Dev hrs saved per week*
Automatically improve code quality

Suggest fixes in the IDE & local builds

- Code corrections while writing it
  - Auto-fix suggestions within the IDE

Fix new errors in PRs

- Write initial code
- Review & approve changes
  - Suggest an automatic fix

Continuously fix tech debt in the existing code base

- Generate Pull requests with fixes
- Review & approve changes
  - Suggest an automatic fix
Automatic app testing

AI generated mobile test flow

Eliminates costly test maintenance

Higher quality, less effort
Takeaways

Standardization improves efficiency at scale

Measure what matters

AI unlocks exciting new opportunities
Thank you!