Folie 1:

Begrüßung  
Spiel erklären: Man muss möglichst viele Begriffe in einer vorgeschriebenen Zeit erraten.

Herausforderungen: Warum Microservices? Etablierte Unternehmen wie Google, FB, Amazon nutzen dies. Auch deutsche wie Otto.de

In dem folgenden Vortrag wird erläutert, was bei einer Planung und Umsetzung beachtet werden muss, bzw. wie man vorgehen kann.

Folie 2:

Inhaltsverzeichnis erläutern.

Folie 3 (Animation): Verdeutlicht die Schwierigkeiten.

Folie 4: -> Es gibt keine Definition (nach DIN) oder so. An Literatur orientiert.

Folie 5: Bildlichen Unterschied zeigen.

Monolith ist eine Anwendung, die alle Funktionen enthält. Microservice: Die Funktionen können aufgeteilt sein. Bsp: Eine Logik zum Erstellen von Rechnungen.

Folie 6: Bewertungskriterien, erwähnen, aber nur drei erläutern (mit Animation). (Abhängigkeiten, Skalierung, Technologie)

Folie 7: Jetzt immer Microservicearchitektur verwenden? Natürlich nicht, es kommt drauf an.

* Kurz sichern, was man hat. Microserviceunterschied zum Monolithen. Anforderungen an Microservices.

Folie 8: Wenn man sich entschieden hat für eine Microservicearchitektur, dann muss/kann/sollte man makro/mikroarchitektonische Entscheidungen treffen. Das kann man sich vorstellen, wie bei einer Stadtplanung. Ggf. kann ich jedes Haus individuell gestalten, aber man sollte sich auf eine Infrastruktur verständigen. Ggf. weitere Beispiele nennen.

(Verweis auf Techtalk von Hendrik)

Folie 9: Konkret werden auf den Bereich der Softwareentwicklung: Checkmarks per Animation setzen.

* Drei Faktoren, die die Architektur maßgeblich beeinflussen:
  + Strategie: Geschäftsführungen (z.B. möchte ich nur in die Cloud)
  + Know How der Leute: Wenn man viele .net Entwickler hat, ist es eher gewagt auf node zu setzen.
  + Welche Kaufkraft: Kann ich mir DB XY überhaupt leisten?

Folie 10: Zeigen worauf ich mich geeinigt habe.

* Kurz sichern, was man hat: Architektur/Technologien sind festgelegt.

Folie 11: Konkreter ans Inhaltsverzeichnis erinnern: Microservices müssen kommunizieren. Synchrone Komm. Erläutern und Vor- und Nachtteile nennen.

**//TODO: Hier Vor- und Nachtteile einordnen.**

Erinnerung mit Animation einblenden.

Folie 12: Alternative mit async. Kommu. nennen. Ablauf erklären. Dann sagen, dass es es schwierig ist, wenn die Services untereinander das Regeln, weshalb sich eine MessageQueue anbietet.

Folie 13:

Prinzip mit der MessageQueue erwähnen.

Folie 14: Bsp, wie es vom Ablauf her funktioniert.

Folie 15: Verschiedene Technologien nennen:

**//TODO: Technologien rausschreiben**

Folie 16: Nun ist klar, welche Technologien und wie Microservices kommunizieren. Nun muss die Sicherheit gewährleistet sein. Gründe liegen wohl auf der Hand: Man möchte keine manipulierten Daten oder Fremdnutzer im System. Erinnerung an Autorisierung und Authentifizierung.

* Naiven Ansatz erklären

Folie 17: Naiven Ansatz auflösen. Der bringt es nicht.

Folie 18: Token anfordern und dann läuft dieser nach Zeit X ab. Microservices sorgen sich nur noch um Autorisierung, also was der Nutzer darf.

Folie 19: Woher man das schon kennt

Folie 20: Technologien erwähnen, sagen, warum ich selbst hoste bzw. IdentityServer genommen habe und dann LIVE Beispiel zeigen.

OAuth -> gehe ich nicht im Detail drauf dein. Zwei Gründe:  
1) Grundsätzlich sehr umfangreich und ich sehe mich da definitiv nicht als Experten

2) Würde den Umfang der Präsentation sprengen, in meiner Arbeit ist dies entsprechend ausgearbeitet. -> Nur Live-Beispiel zeigen

Folie 21:

* Kurz sichern, was man hat: Technologien sind festgelegt, Kommunikation ist sichergestellt und die Sicherheit – wie das alles verknüpfen?

Nochmal zeigen, was ich habe.

Folie 22: Grundsätzlich jeder Service hat seine eigene Datenbank.

Folie 23: Kommunikation Asynchron über die RabbitMQ.

Folie 24: Naiven Ansatz zeigen

Folie 25: Auflösen mit API-Gateway.

Folie 26: Technologien erwähnen, Eigenbau, Ocelot, Clouds. Für Ocelot entschieden, da sehr auf IdentityServer zugeschnitten + Microsoft empfohlen

Folie 27: So wie es nun ist, ist es schon recht okay. Allerdings muss man nur überlegen, wie man die ganze Anwendung hostet. Gängige Praxis ist es, die einzelnen Anwendungen in einem Container laufen zu lassen. Was heißt das genau -> 28

Folie 28: Ablauf erklären

Auf Alternativen hinweisen und sagen, warum man diese nicht nimmt.

**//TODO: Alternativen hier aufnehmen**

Folie 29: Zeigen wie das auf dem Server aussieht.

Folie 30: Gesamte Architektur zeigen

Folie 31: Den Apache als Weiterleitung erwähnen. Vergleichbar mit NGINX, wird halt von meinem Hoster so unterstützt.

Folie 32: Sicherung des Wissens

Folie 33: Ausblick, was es doch noch gibt.

Folie 34: Nicht erwähnte Helden nennen du Verweise geben.

Folie 35: Vielen Dank und Feedback einbauen