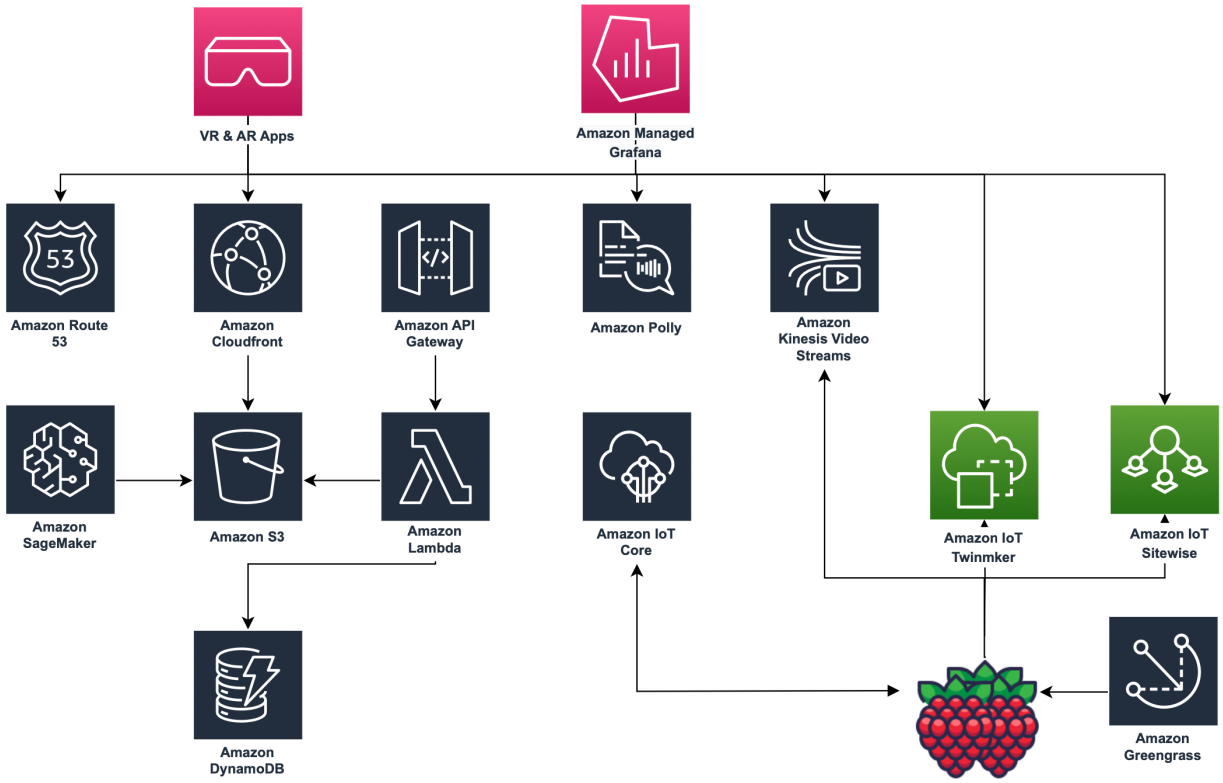
# 亚马逊云科技梦幻城市文档

## 架构图:




## Dream City 涉及相关技术文档：

IoT Core:https://docs.aws.amazon.com/iot/

IoT SiteWise: https://docs.aws.amazon.com/iot-sitewise/

IoT GreenGrass: https://docs.aws.amazon.com/greengrass/

IoT TwinMaker: https://docs.aws.amazon.com/iot-twinmaker/latest/apireference/Welcome.html

Managed Grafana: https://docs.aws.amazon.com/grafana/latest/userguide/what-is-Amazon-Managed-Service-Grafana.html

Kinesis Video Stream: https://docs.aws.amazon.com/kinesis/

## 台词演示:

> 你好,欢迎来到 DREAM 演示,你好吗?

>> 我很好,谢谢。

> 你想了解更多我们在这里所做的事情吗?

>> 是的,请告诉我是什么?

> 好吧,如果我必须用一行话概括的话,我会说这个演示展示了如何从边缘位置收集数据,将它们推送到云端,并使它们成为可操作或有用的数据。

> 如果跟着我,我很乐意成为你今天的导游,所以请上 DREAM 巡回巴士。

[与参与者一同走到太阳能农场一侧 - 位置 1]

> 欢迎来到 DREAM 太阳能农场!一切都始于我们在这里的太阳能农场。如你所见(用手指指出),我们这里有几个太阳能电池板实时发电,它们连接到这个电路板和树莓派(用手指指出)。我们还有一个环境光传感器和温度传感器(用手指指出)。它们都在实时发送测量数据,数据从传感器流向树莓派(用手做出流动的手势)。

> 我们在树莓派上部署了一个开源边缘运行时 Greengrass,并开发了一个自定义的 Greengrass 组件,这是一个 Python 应用程序,它收集数据并将其推送到云端的 IoT SiteWise。我们正在向 IoT SiteWise 发送时间序列数据,其中包含时间戳和传感器捕获的值。

> 然后,这些数据被展示在这个托管的 Grafana 仪表板上,我们就在这里(走向仪表板并指向它)。

> 因此,如果你看左上角的仪表板,你可以看到太阳能电池板产生的电压值为<X>,以及两个传感器的值。光线值为 <Y>,温度为 <Z>(指向仪表板上的数值,让客户真正看到并注意到这些值)。所以请明确,这是实时发生的!

> 让我们试试看。我们试着遮挡太阳能电池板,看看会发生什么。这里有一些东西可以遮挡(拿起纸片),谁想试试?。

> 那么请拿这个,放在太阳能电池板上遮挡光线。

> 现在看看托管的 Grafana 仪表板。我们可以注意到,电压值显著下降,它原本是 <X>,现在下降到了 <W>。让我们移开遮挡,看,它又恢复正常了。

> 还记得我开始时给你的一行概括吗?

> 就是让数据可操作!

> 那么,我们如何让我们收集的数据可操作呢?

> 好吧,让我向你介绍 Frank(指向太阳能农场旁边的小乐高人偶)。嗨,Frank!你今天过得怎么样?(没有反应) - 好吧,看来 Frank 今天很忙!

> Frank 有一份重要的工作,他在太阳能农场担任维护经理。最近让 Frank 头疼的一件事是,他注意到太阳能电池板有一些问题。

> 即使是阳光明媚、天气炎热的日子,它们也会突然产生较少的能量!所以,为了让 Frank 的生活更轻松,我们实现了实时异常检测,使用了一个非常简单的基于规则的警报。

> 规则是这样的:如果天气晴朗炎热,这意味着光线和温度值很高,但电压值很低,那就是异常情况。如果发电量不足,就会触发一个警报,所以现在 Frank 会在发生这种情况时收到通知!

> 让我们再试一次,你能再次遮挡太阳能电池板吗?现在看看仪表板上的那个框,它变成红色了,上面写着"发电量异常"。如果我们移开遮挡(让参与者移开遮挡),它就会恢复正常。所以,这是一种非常简单的方式,通过实现异常检测使数据可操作。

> 但是,还有更多的机会!Frank 遇到的另一个问题是,太阳能电池板有时会变脏。污垢积累会影响太阳能电池板的效率。我们为 Frank 开发了一个污垢检测器。这个相机(指向第二个太阳能电池板顶部的小相机)每秒拍摄多张照片,我们使用迁移学习在 Amazon SageMaker 上构建了一个机器学习模型,并将其部署在这个树莓派上。我们通过拍摄 20 张有污垢和 20 张无污垢的电池板照片来训练模型。这个轻量级模型可以轻松在这个树莓派上进行推理。推理结果实时推送到云端,并在 Grafana 仪表板上显示。看看仪表板上的污垢检测器窗口(指向它)。

> 让我们试一试。我这里有一张里面含有污垢的纸片 。让我们看看会发生什么,如果你拿着它(递给参与者),撒在这个电池板上。现在密切关注污垢检测器窗口!你可以看到,它现在检测到电池板上有污垢。

> 所以,在这第二个例子中,我们也在展示如何通过实现更复杂的异常检测来使数据可操作,这次使用了人工智能/机器学习。

> 这都很棒,但信息也可以与其他系统共享!让我们再次登上 DREAM 巡回巴士,前往下一站!

[带领所有人移动到中心/梦之城前面 - 位置 2]

> 欢迎来到梦之城!请小心下车!

> 梦之城由几个部分组成。左边是月光公园,摩天轮和旋转木马都很受孩子们的欢迎。旁边是城市电影院。中间是中央广场,有餐馆和著名的爵士乐俱乐部。在城市的最后,有警察局、医院和市长的房子。这座城市可以分为两部分,左边的非关键基础设施,包括中央广场、月光公园、电影院等。而右边是关键基础设施,有警察局、医院和市长的房子。

> 市长(指着乐高人偶)非常自豪地以最佳方式管理能源消耗。他要求我们在检测到太阳能电池板上有污垢时,向梦之城发送通知,以便调整和优先考虑能源消耗。市长实施了一个自动化规则,它将关闭非关键基础设施的电源,但将保持所有关键设施运行,这意味着医院、警察局和市长的房子将继续供电。是的,市长害怕黑暗!

> 让我们试试看,我将把污垢放在太阳能电池板上,请留意城市的灯光变化。 (用污垢纸片遮挡太阳能电池板)

> 正如你所看到的,火车停止运行,所有非关键基础设施的灯都熄灭了!

> 现在,让我们移除遮挡(移开纸片),一切恢复正常。

> 再次,我们使数据可操作和有用,通过通知不同的实体,这个例子中是梦之城。顺便说一句,梦之城可能距离太阳能农场几百英里远。城市可以决定采取反应或做出决策,无论决策是什么。显然,这不是一个真实实施的例子,我想没人会突然切断人们的电力或停止火车运行。

> 是时候来到我们的最后一站了,请重新登上 DREAM 巡回巴士。终点站是风力发电场。

[带领所有人跟随你走向一侧的风力发电场 - 位置 3]

> 欢迎来到风力发电场!在这里,我们采用了与太阳能农场类似的方法。我们有三台风力涡轮机,我们正在用这个风扇吹风驱动它们。涡轮机连接到这个树莓派(用手指指出)。涡轮机产生的电压被发送到树莓派,然后发送到 IoT SiteWise。这些数据也显示在这个 Grafana 仪表板上(走向仪表板并指向三个电压测量值)。我们可以在这里看到涡轮机 1、2 和 3 的电压值。由于风扇的方向,我们只有两台涡轮机在旋转,这就是为什么涡轮机#3 的值为零。

> 我们在这里做了一些稍微不同的事情,就是我们创建了一个风力发电场的数字孪生体。(指向仪表板上的数字孪生体)。这个数字孪生体是一个 3D 模型,与其物理孪生体实时连接。对于某些用户而言,数字孪生体只是另一种查看相同数据的方式,它提供了更好的用户体验或有助于更好地监控设备或机器,因为它更加直观。

> 让我们试试看,如果我堵住中间的涡轮机 2,你会看到数字孪生体会实时作出反应。你应该能看到那个涡轮机停止了。

> 是时候介绍一下 Sarah 了(指向风力发电场旁边的小人偶)。Sarah 是一名维修工人,专门负责风力发电场。她的工作是密切关注不同的涡轮机,确保一切正常。为了做好她的工作,她需要访问一些实时数据。为此,她通常需要打开浏览器、使用搜索表单、输入与涡轮机相关联的序列号,然后尝试通过电子表格导航数据。

> 为了让她的工作更轻松,我们开发了一个增强现实应用程序,可在手机上运行,使用与我们在 Grafana 仪表板上显示数据时使用的相同 API。由于我们已经完成了艰巨的工作,构建这个应用程序对 Sarah 来说相当容易。

> 如果你想体验,可以试一试,或者我可以在我的手机上向你展示。所以，你可以看到,这是一个运行在浏览器中的应用程序,这是 Safari。我们可以看到手机的相机,应用程序可以检测标记,就像这个(拿着标记)。假设这个标记就在涡轮机旁边,每个涡轮机都会有不同的标记。那么 Sarah 只需用手机看着这个标记,她就会看到这样的画面(指向标记并展示增强现实 UI 元素)。所以,这是我们在 Grafana 上显示的同一数据,并且会实时更新,你可以看到(等待图表上的数据刷新,以便与会者能看到它的工作情况)。

> 最后,我们还开发了一种虚拟现实体验。这次,它是为了那些可能距离数百甚至数千英里远的用户而准备的。他们可以在 360 度体验中看到同样的数据。你可以试一试,如果你愿意的话。