

KROMESCH SÁNDOR

# APP FELHŐ

API-k és Webszolgáltatások a Cloudban

## Mobil eszközök

1. Hordozhatók
2. Könnyen kezelhetők
3. Limitált erőforrásokkal rendelkeznek
4. Hozzáférnek az Internethez



## Szerverek ezrei

1. “korlátlan” erőforrások
2. hozzáférnek az Internethez



# Kiindulás: Cloud 2/2

2/48

## Szerverek ezrei

1. “korlátlan” erőforrások
2. hozzáférnek az Internethez

## Szerverek száma:

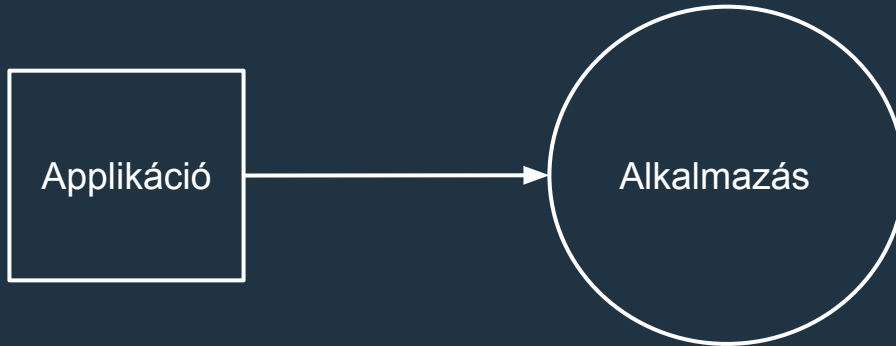
Amazon: 150e - 400e

Google: 1.500e+



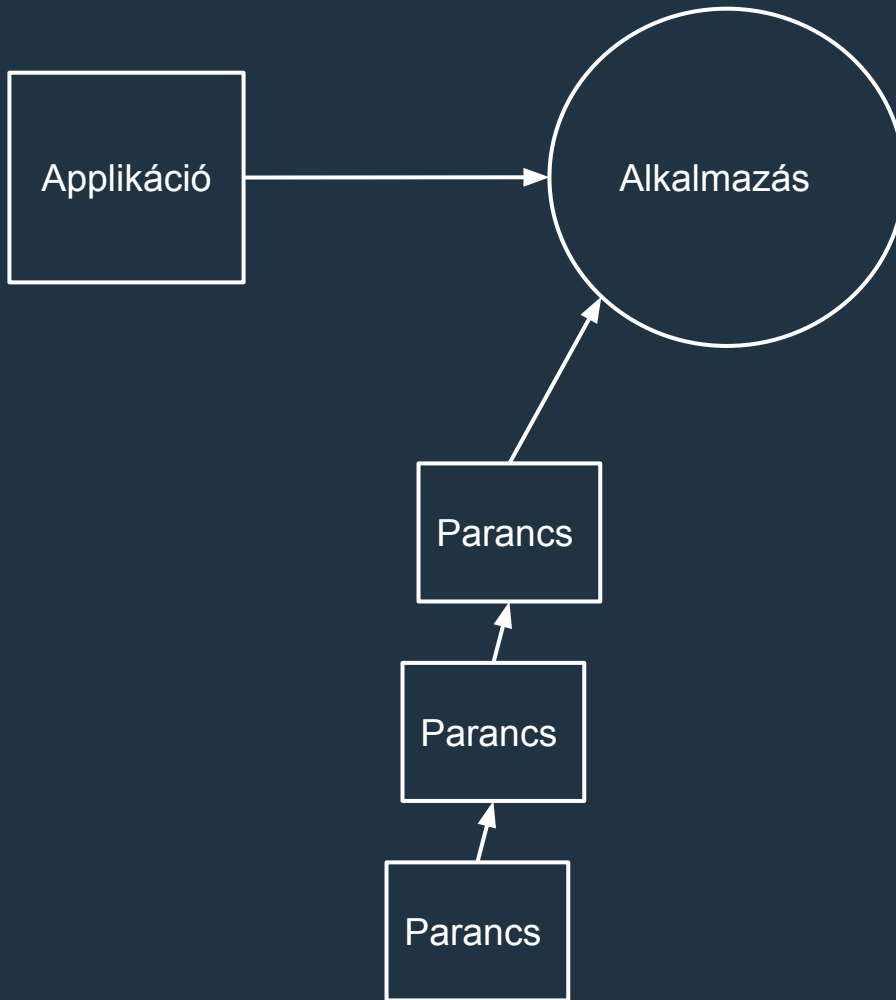
# Mi a Cloud célja?

3/48



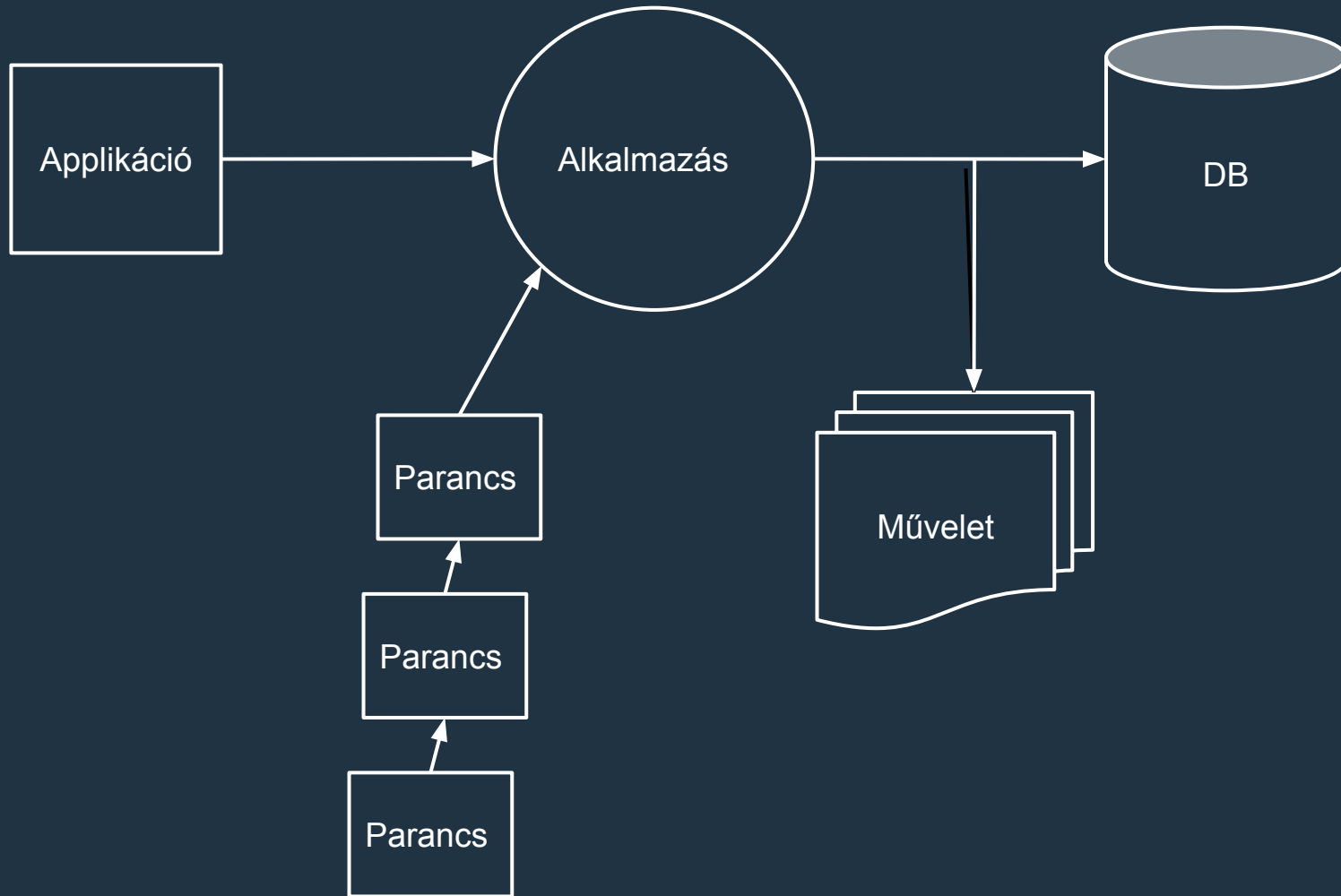
# Mi a Cloud célja?

3/48



# Mi a Cloud célja?

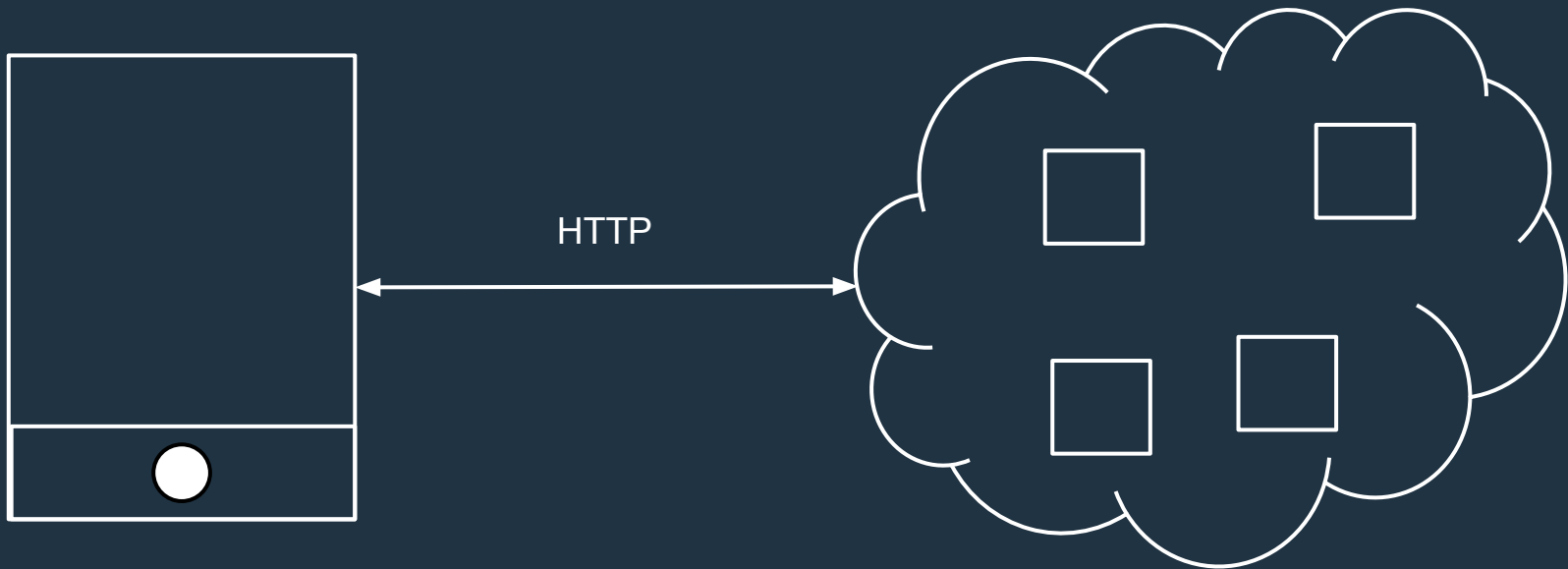
3/48



# Miért HTTP?

4/48

Miért használjuk web böngészőkhöz kifejlesztett prototolt a Mobil eszközökkel és a Cloud szolgáltatásokkal?





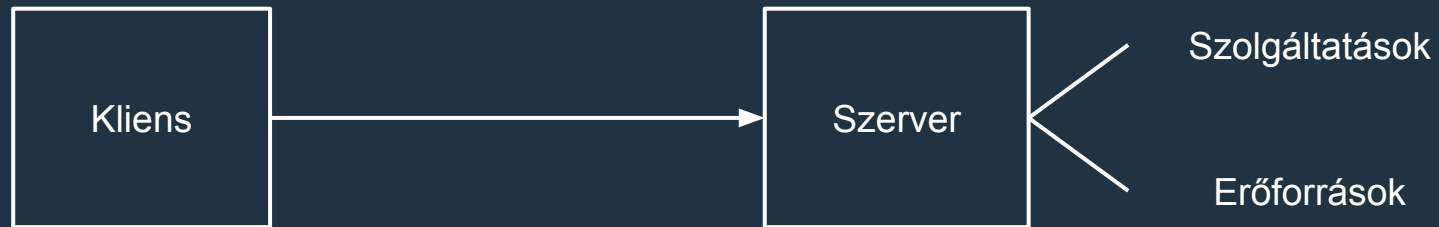
# Miért HTTP?

5/48



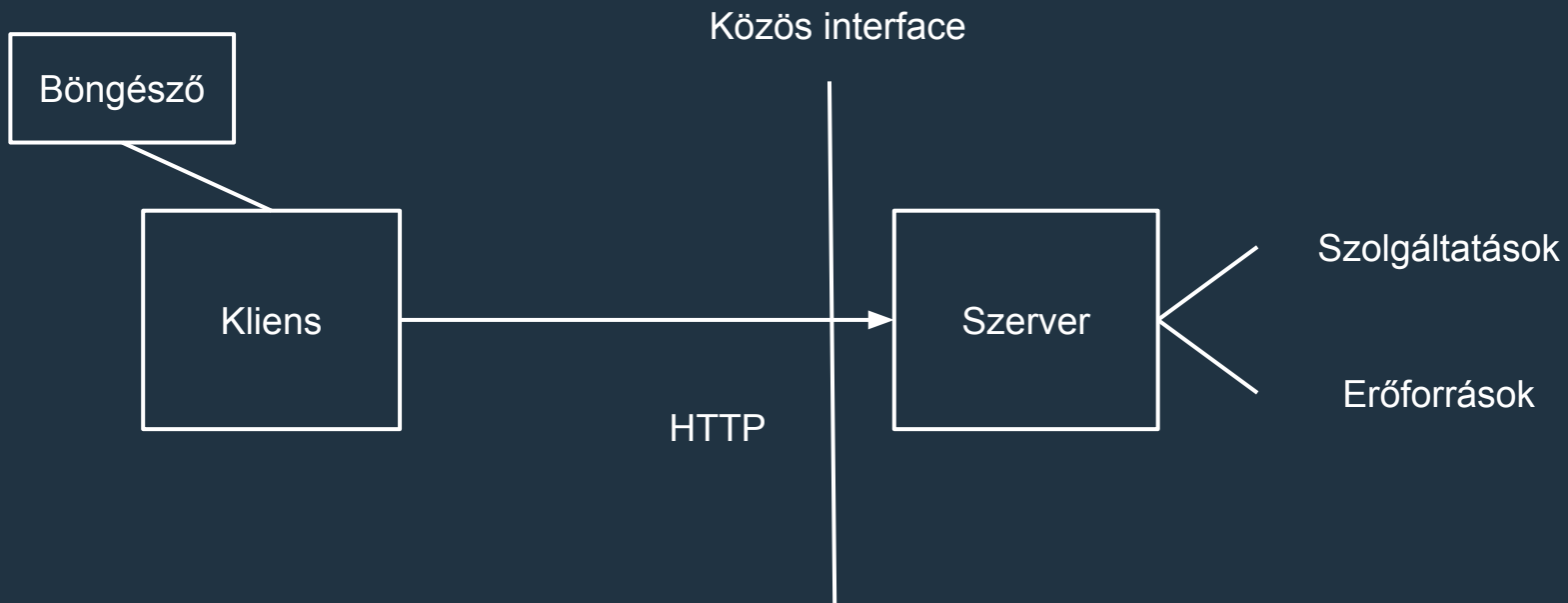
# Miért HTTP?

5/48



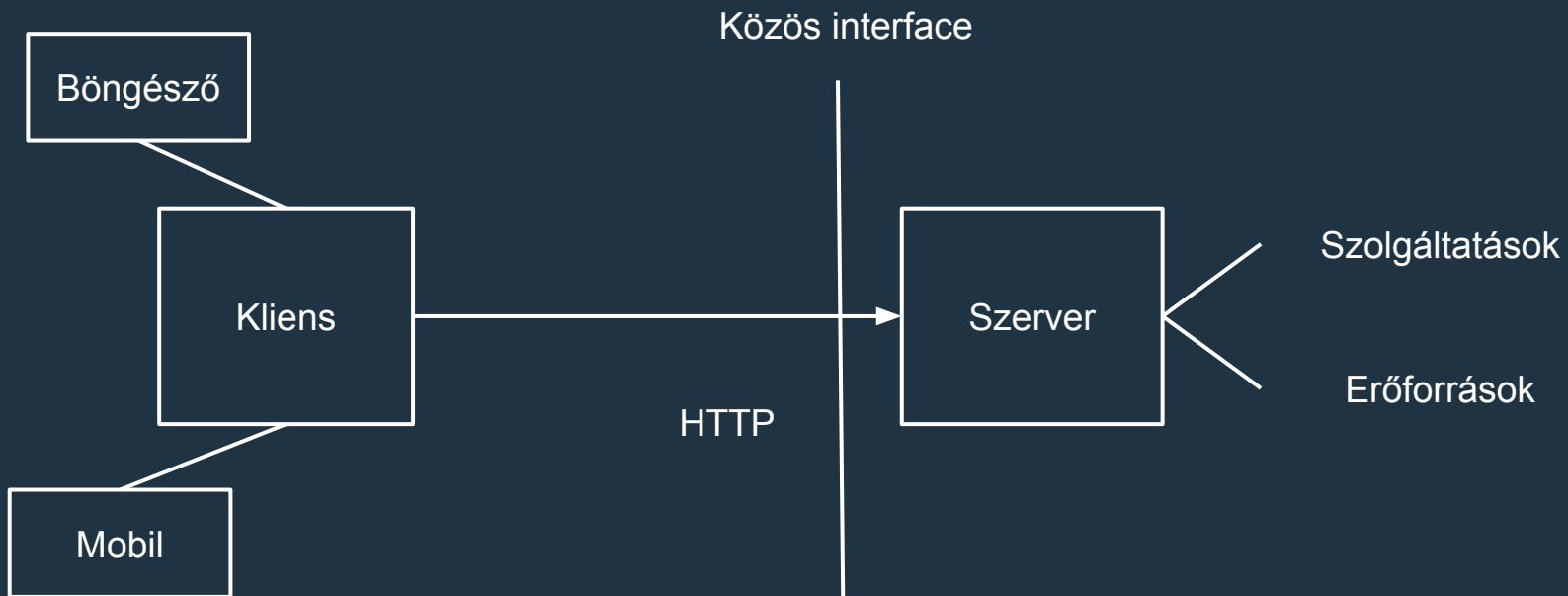
# Miért HTTP?

5/48

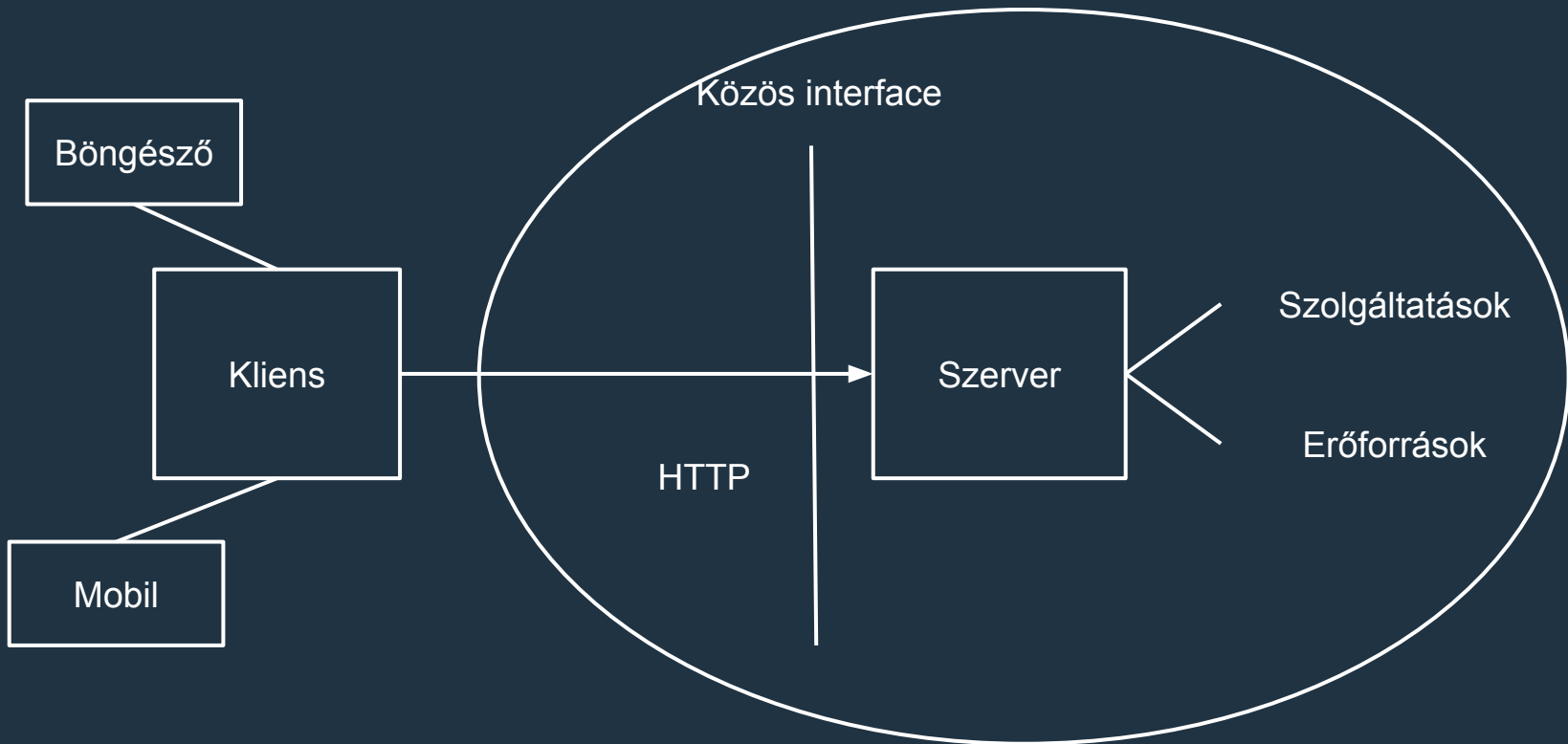


# Miért HTTP?

5/48

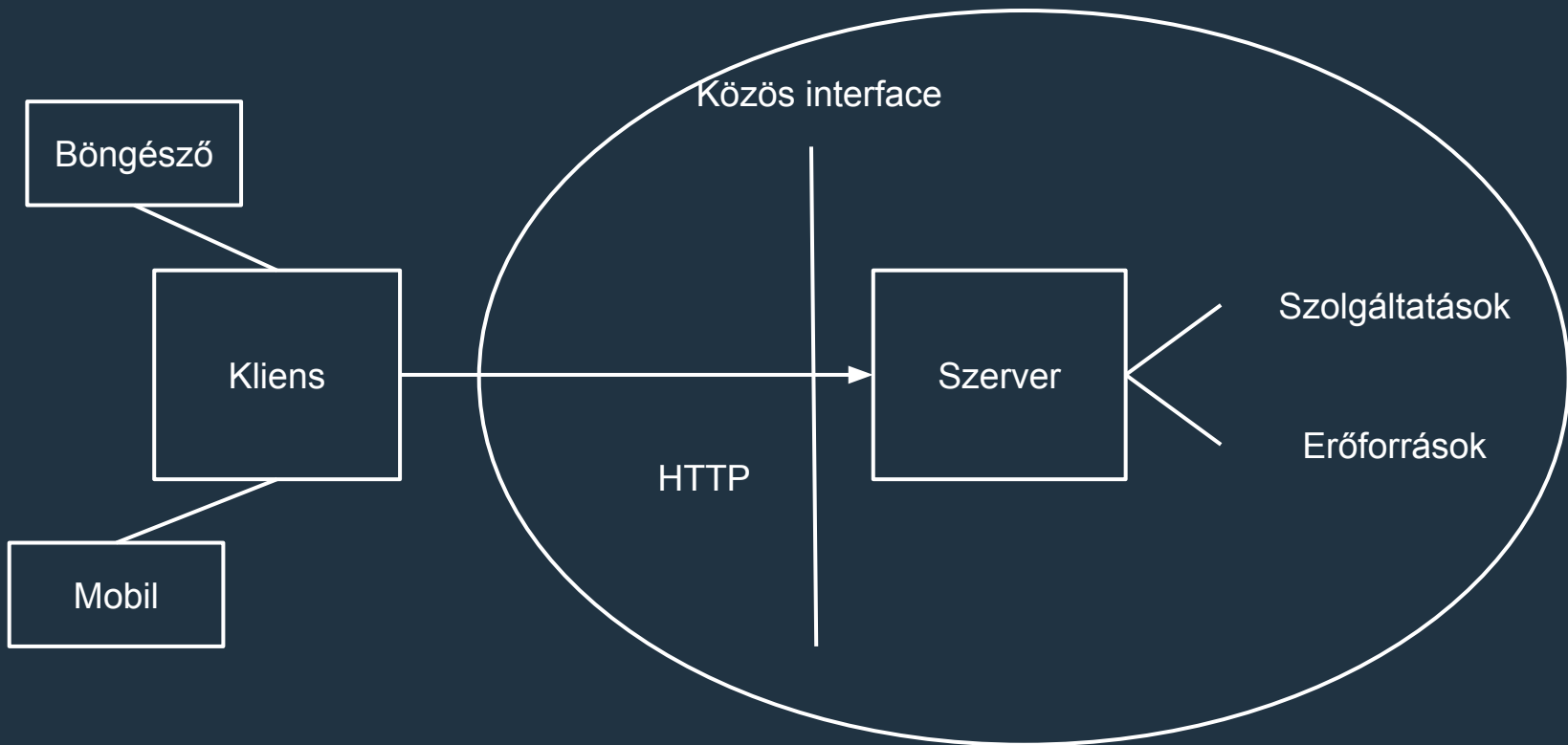


# Miért HTTP?



# Miért HTTP?

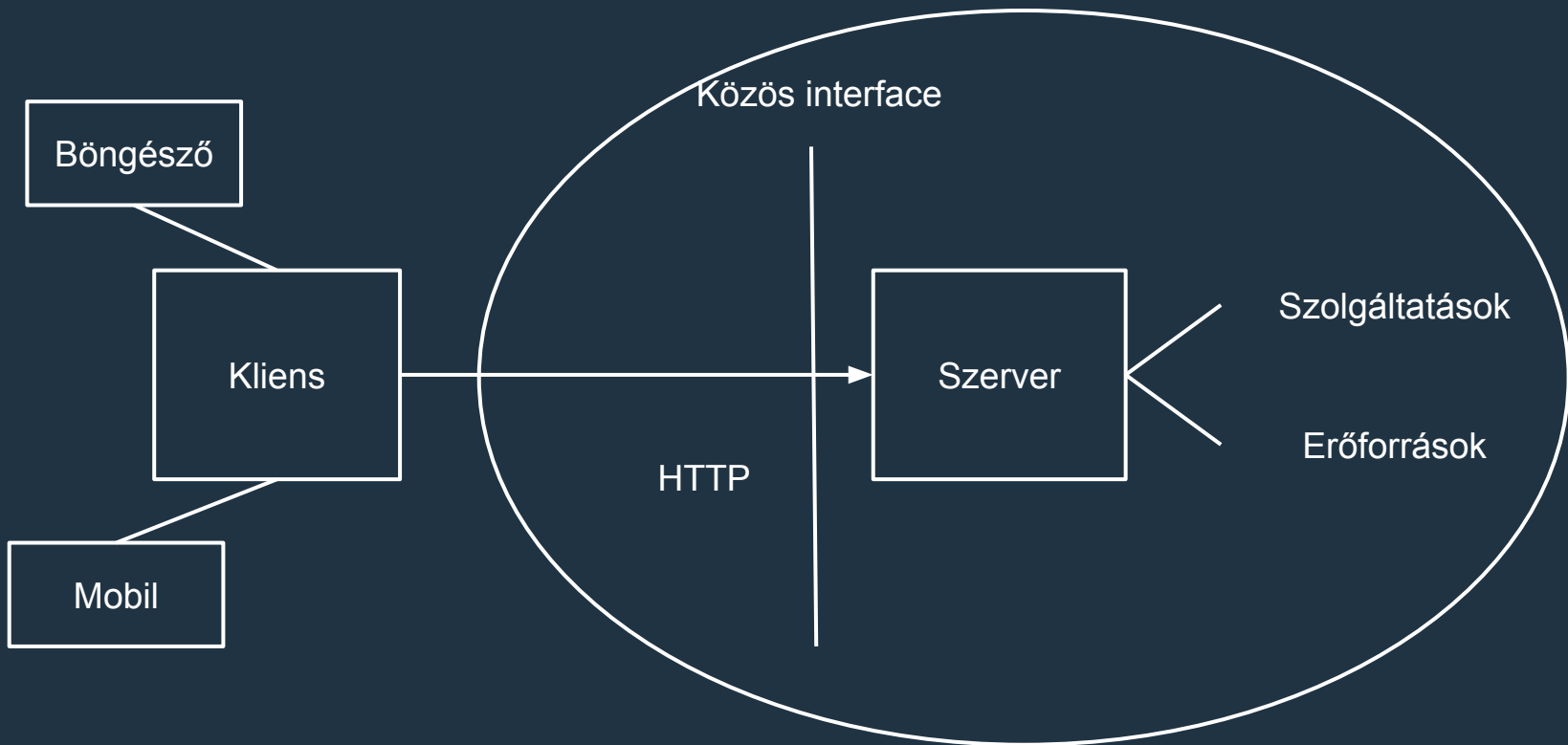
5/48



- Frameworkok

# Miért HTTP?

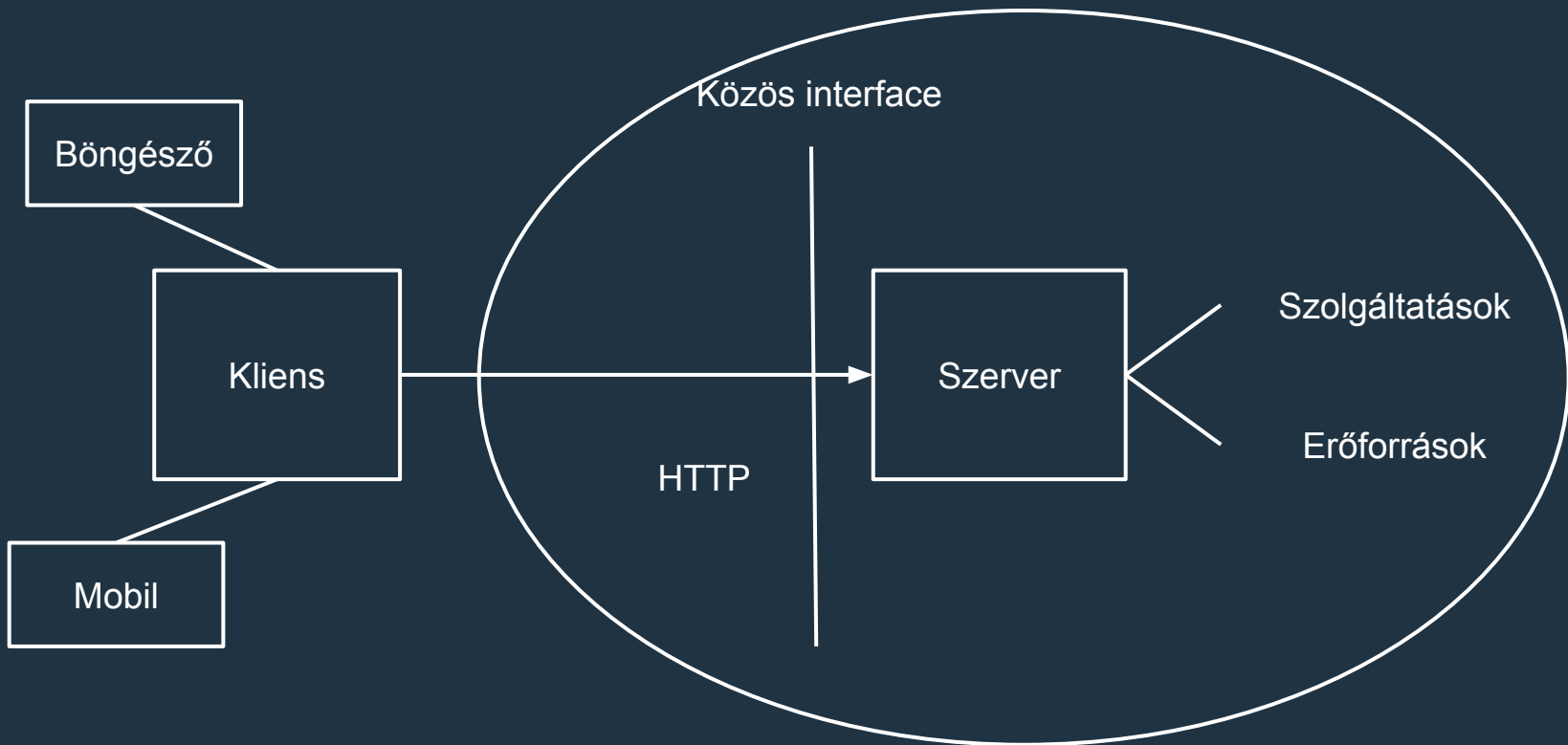
5/48



- Frameworkok
- Data marshaling

# Miért HTTP?

5/48

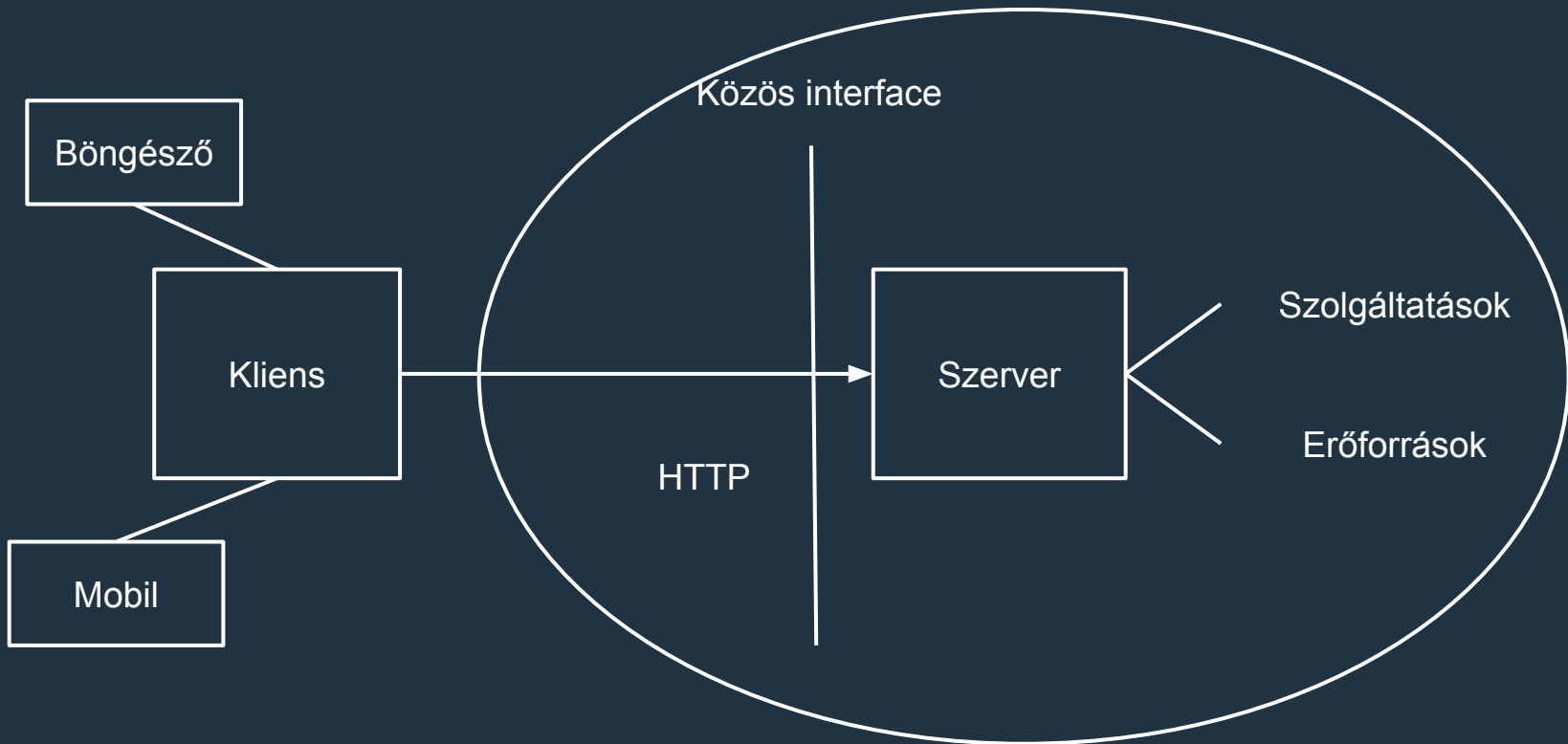


- Frameworkok
- Data marshaling
  - Library-k a HTTP kérések feldolgozáshoz



# Miért HTTP?

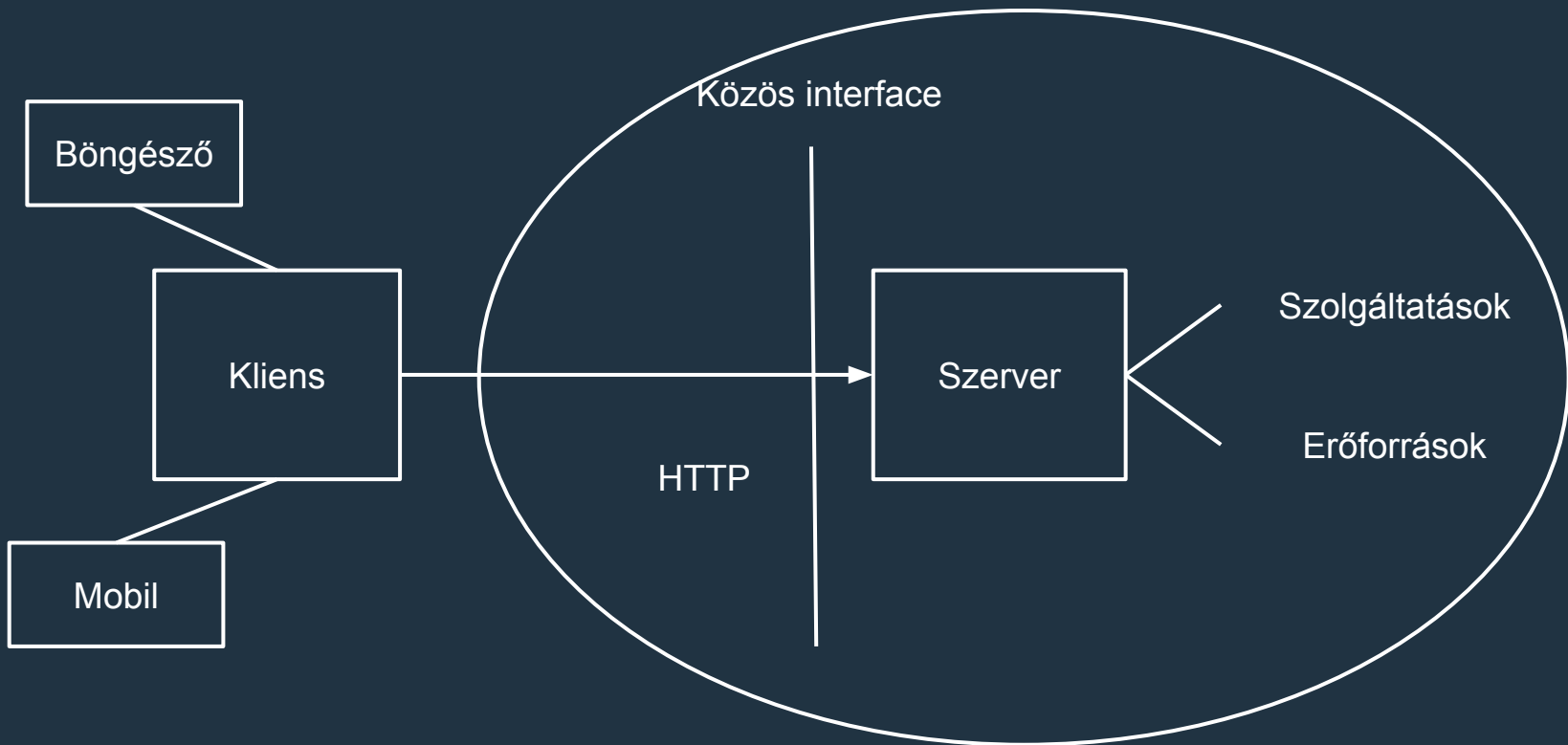
5/48



- Frameworkok
- Data marshaling
  - Library-k a HTTP kérések feldolgozáshoz
  - Session kezelés

# Miért HTTP?

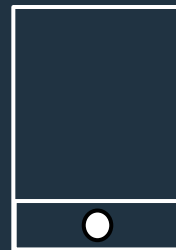
5/48



- Frameworkok
- Data marshaling
- Library-k a HTTP kérések feldolgozáshoz
- Egyéb eszközök: Load Balancing
- Session kezelés

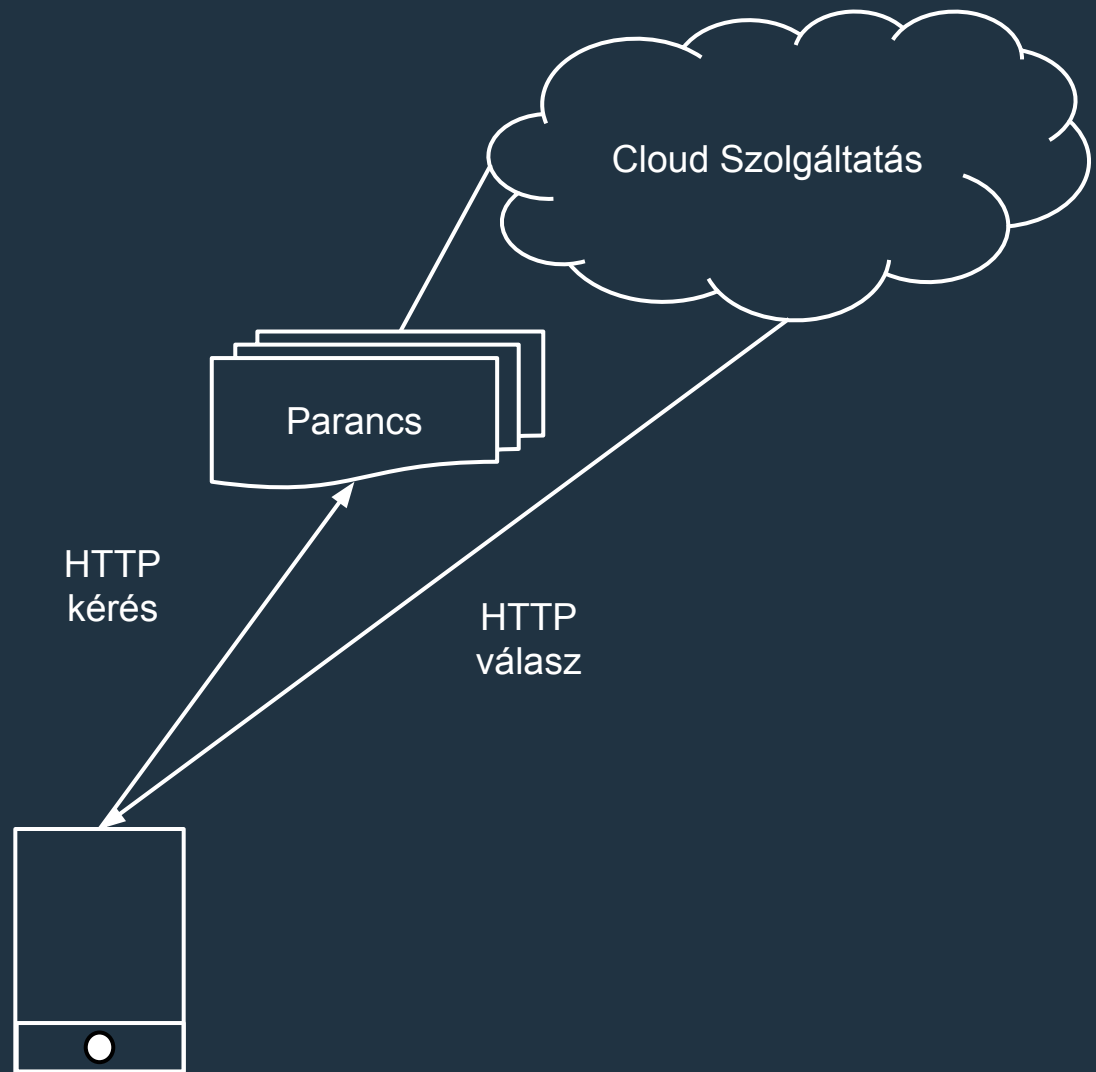
# Cloud szolgáltatás felépítése

6/48



# Cloud szolgáltatás felépítése

6/48

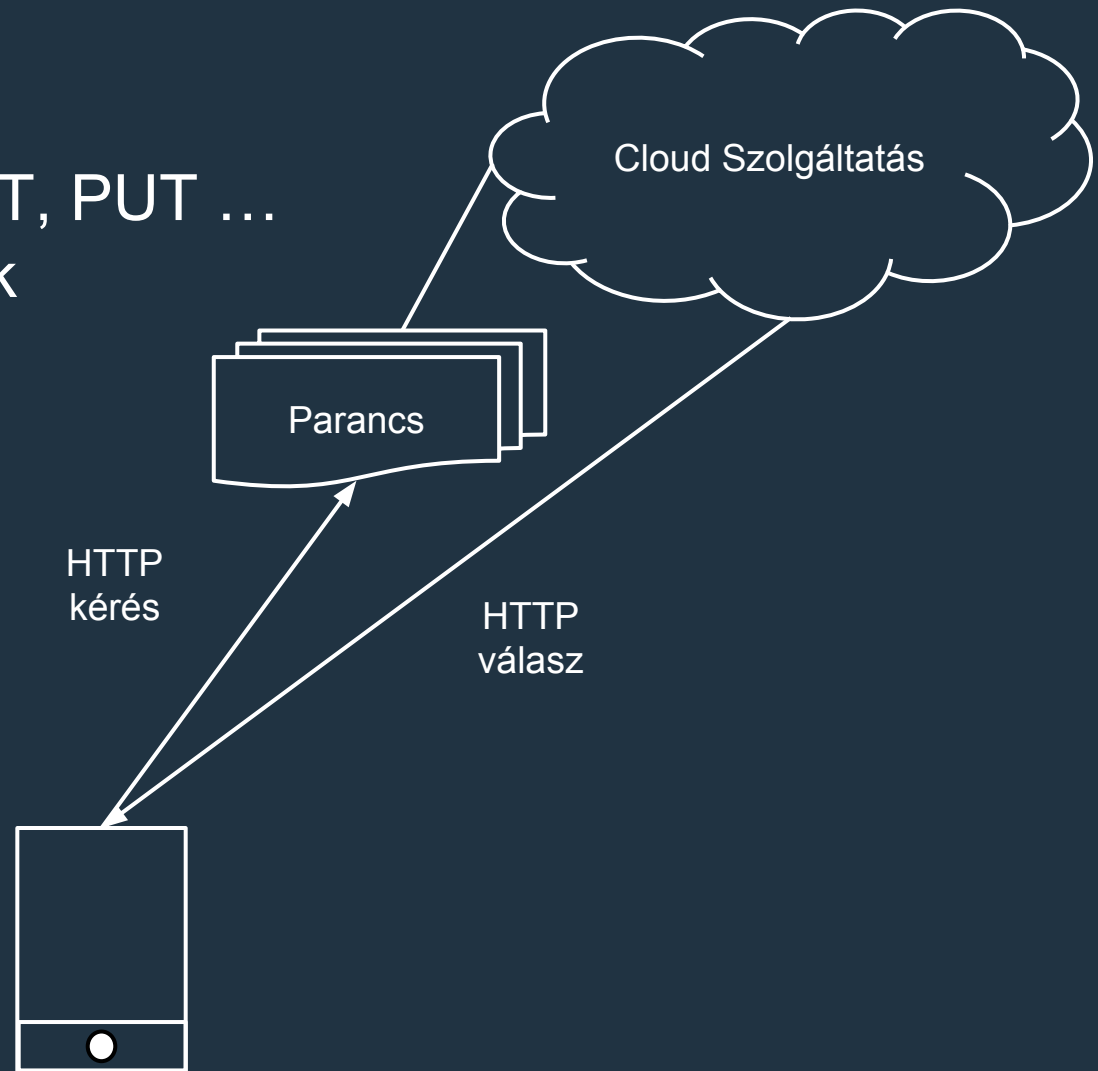


# Cloud szolgáltatás felépítése

6/48

## Kérés

- Method: GET, POST, PUT ...
- Query paraméterek
- Content Type
- Body



# Cloud szolgáltatás felépítése

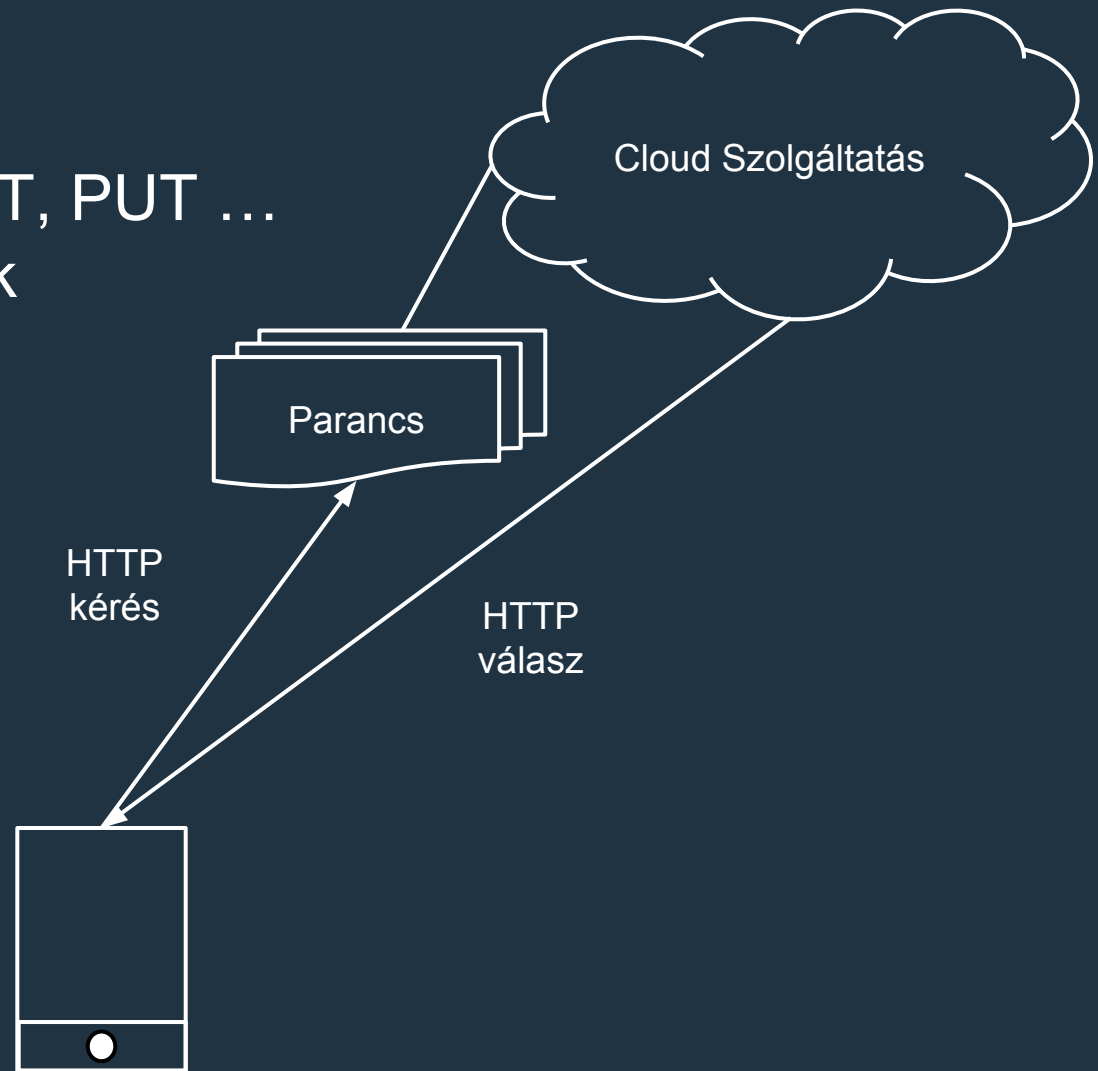
6/48

## Kérés

- Method: GET, POST, PUT ...
- Query paraméterek
- Content Type
- Body

## Válasz

- HTTP Státusz kód
- Content Type
- Body



# TERVEZÉSI / MEGVALÓSÍTÁSI KÉRDÉSEK

# Skálázhatóság

7/48

## Vertikális

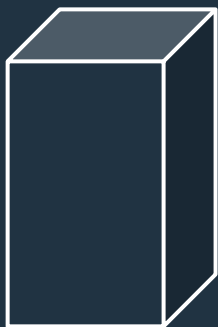




# Skálázhatóság

7/48

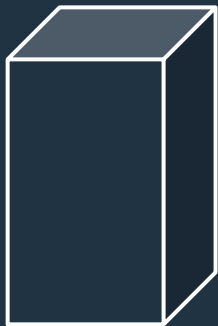
## Vertikális



# Skálázhatóság

7/48

Vertikális

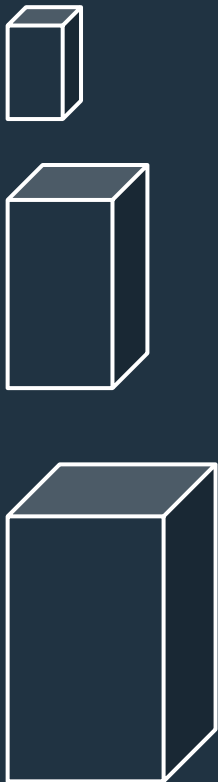


Horizontális

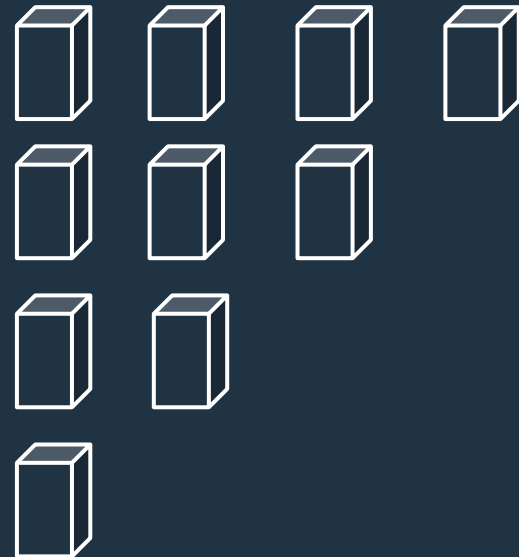


# Skálázhatóság

## Vertikális

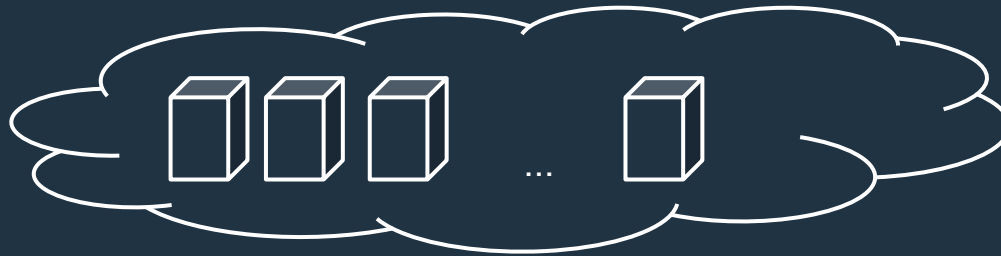


## Horizontális

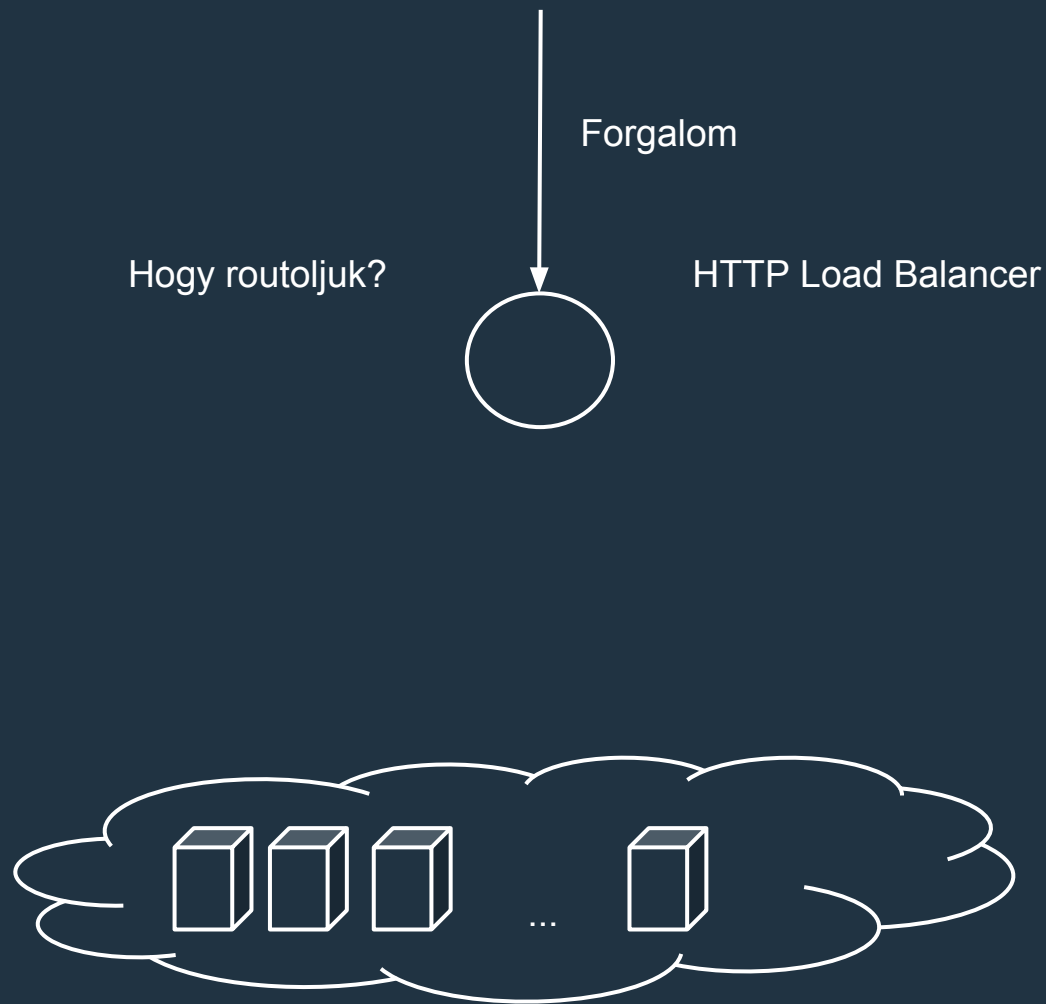


# Loadbalancing 1/3

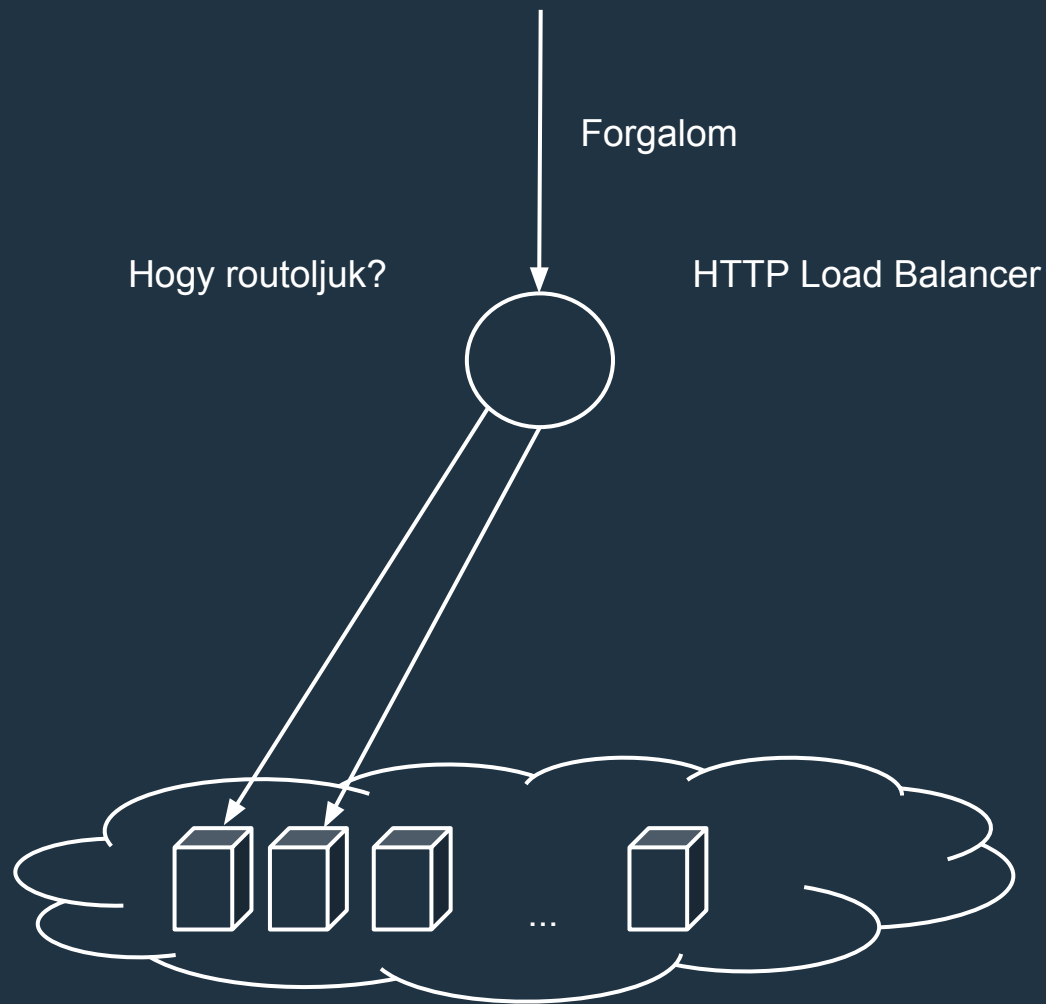
Forgalom



# Loadbalancing 1/3

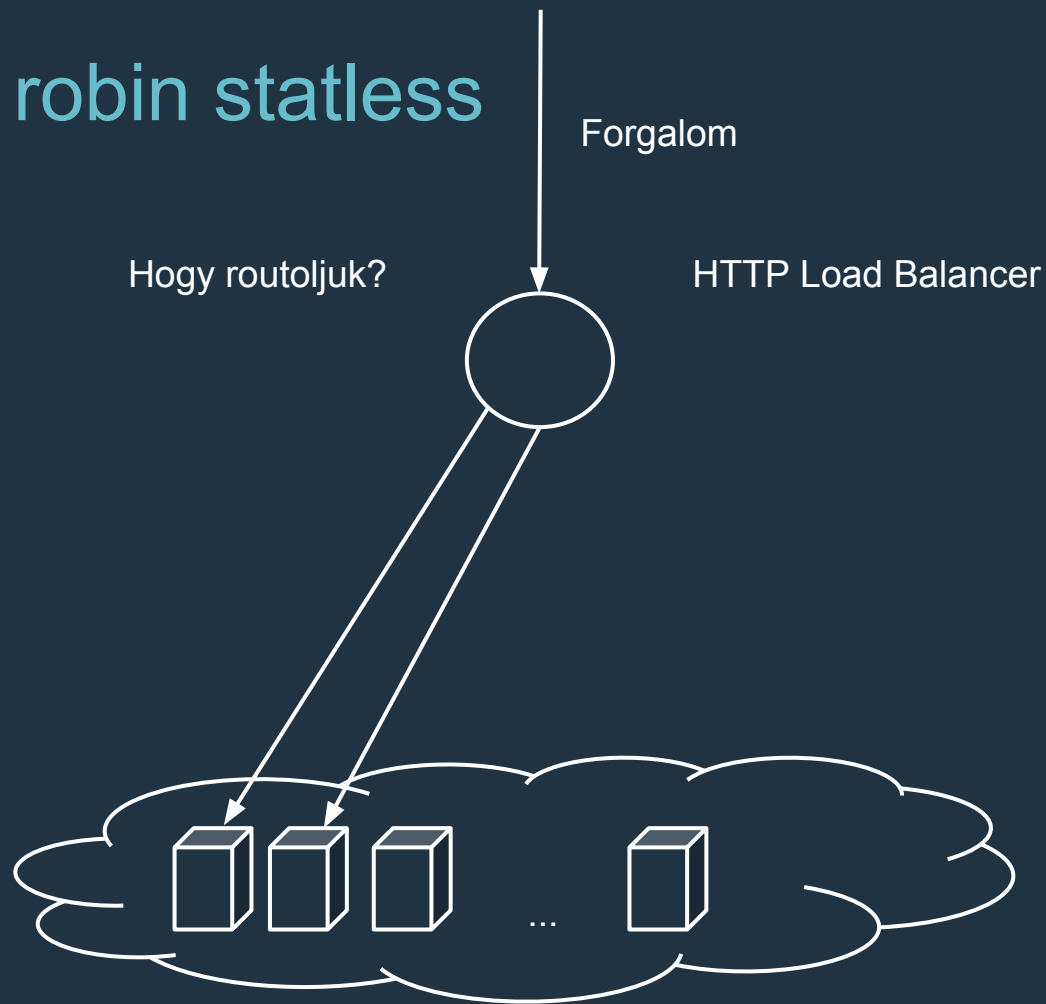


# Loadbalancing 1/3

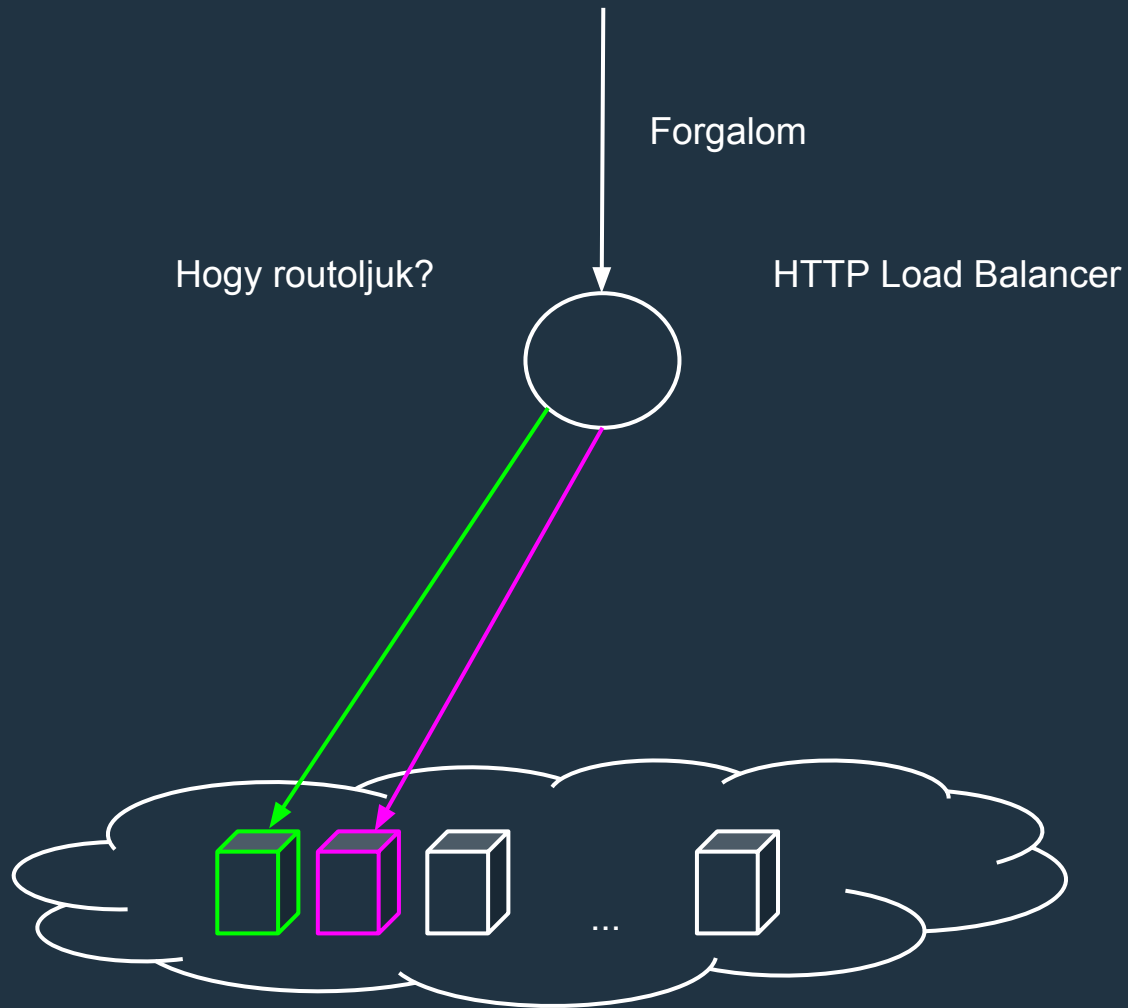


# Loadbalancing 1/3

## Round robin statless



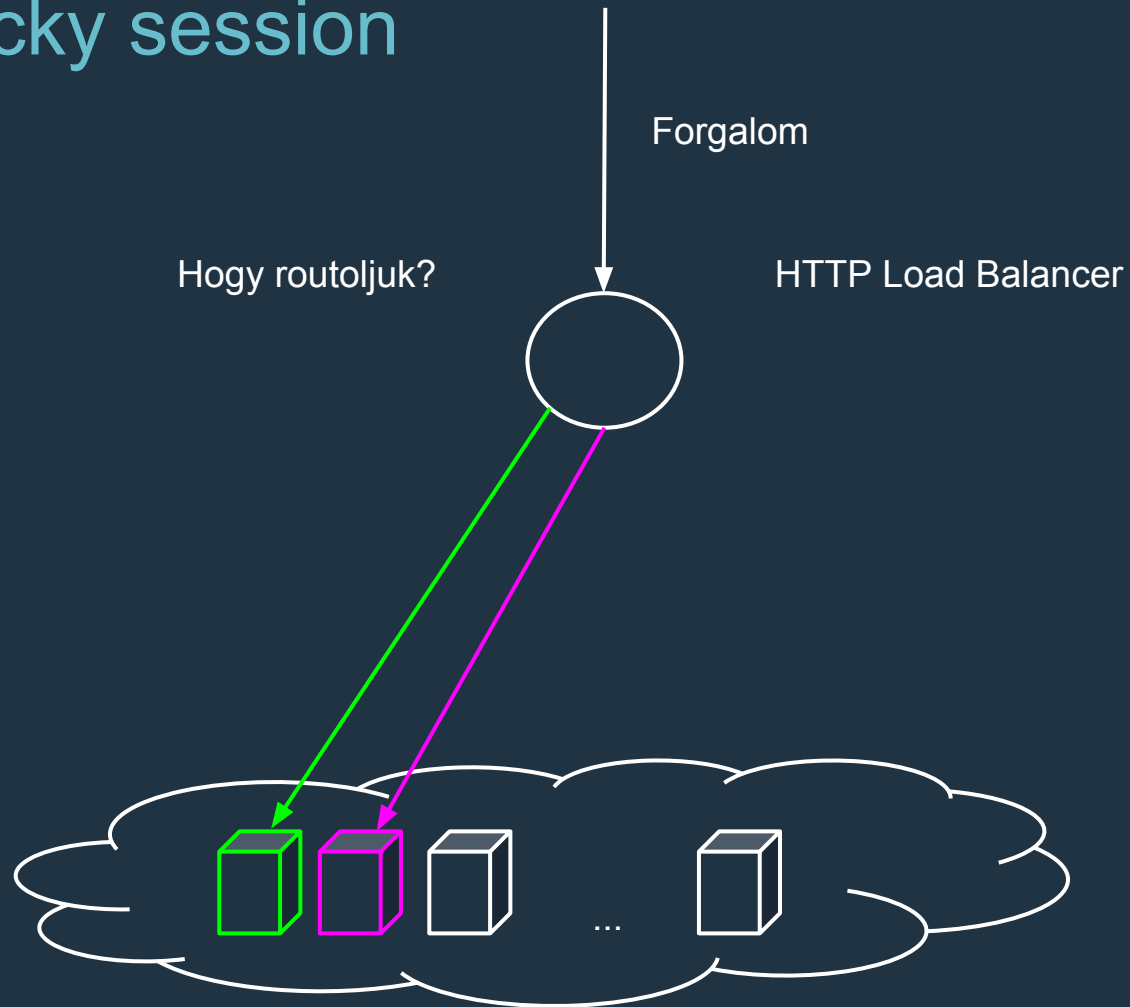
# Loadbalancing 2/3



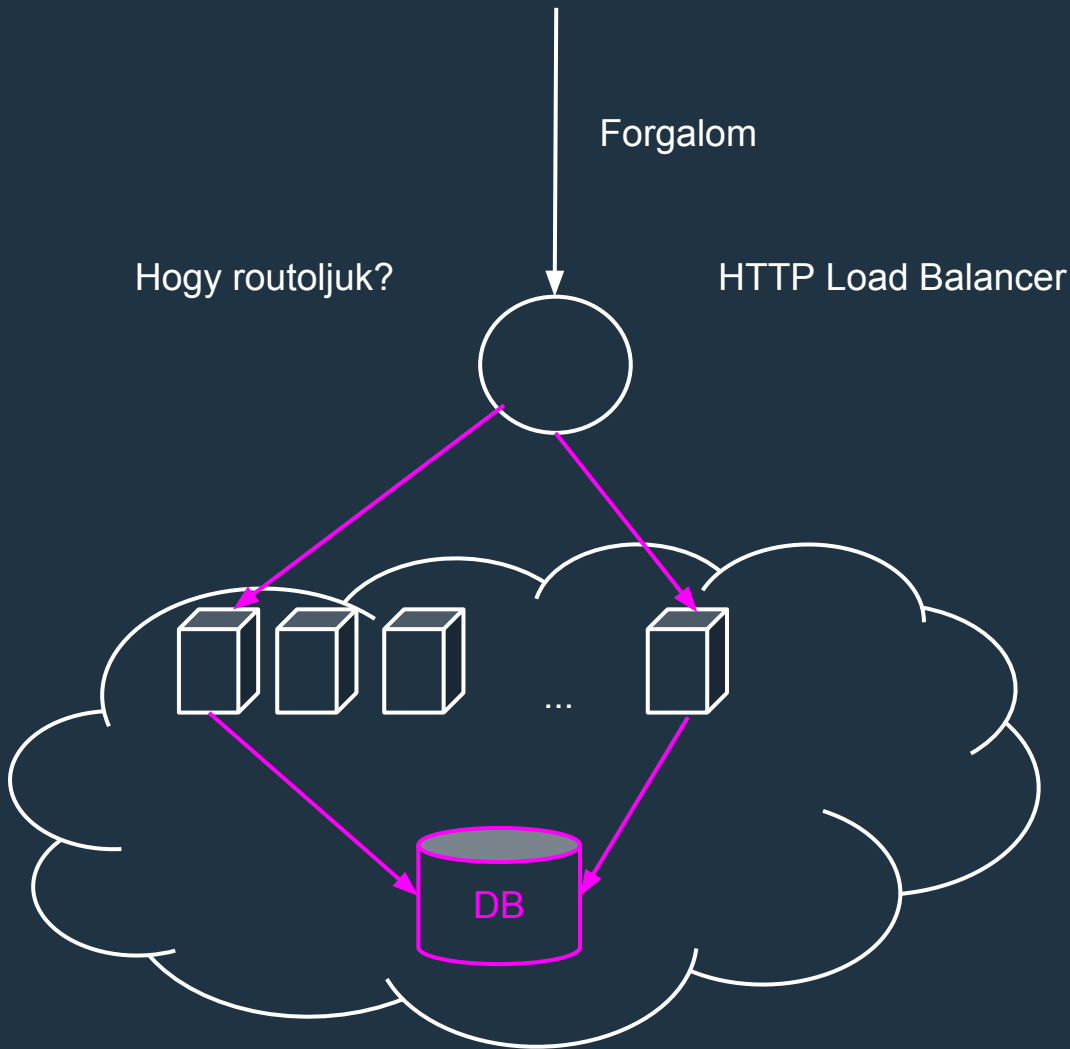


# Loadbalancing 2/3

## Sticky session

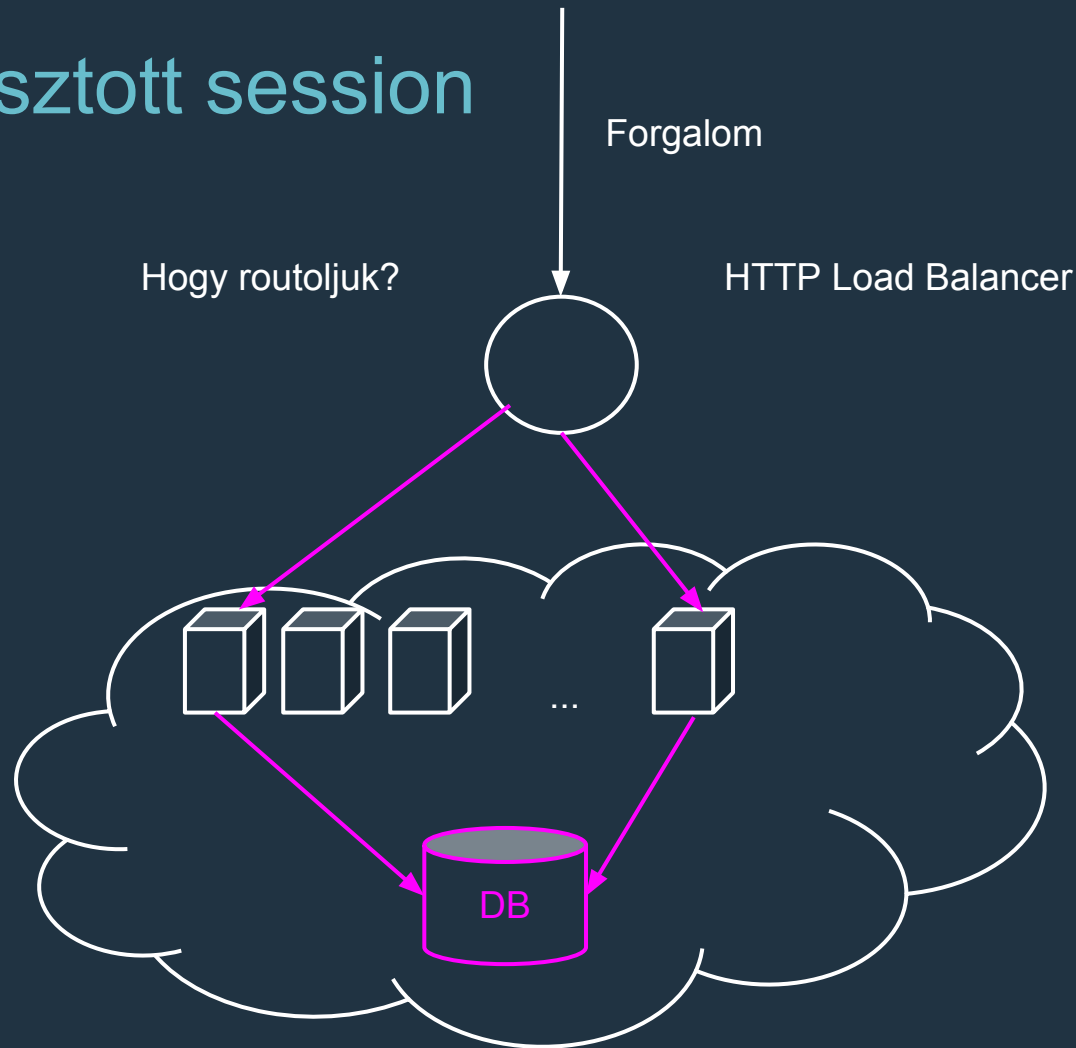


# Loadbalancing 3/3



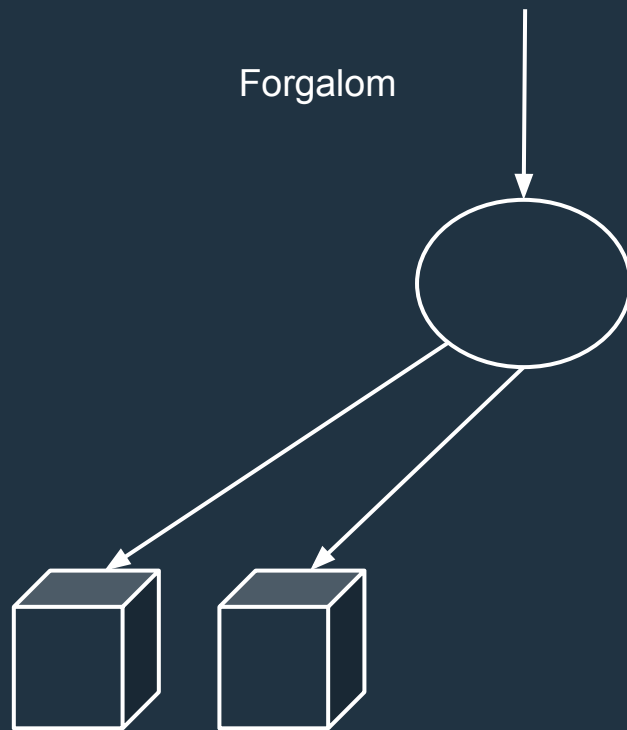
# Loadbalancing 3/3

## Elosztott session

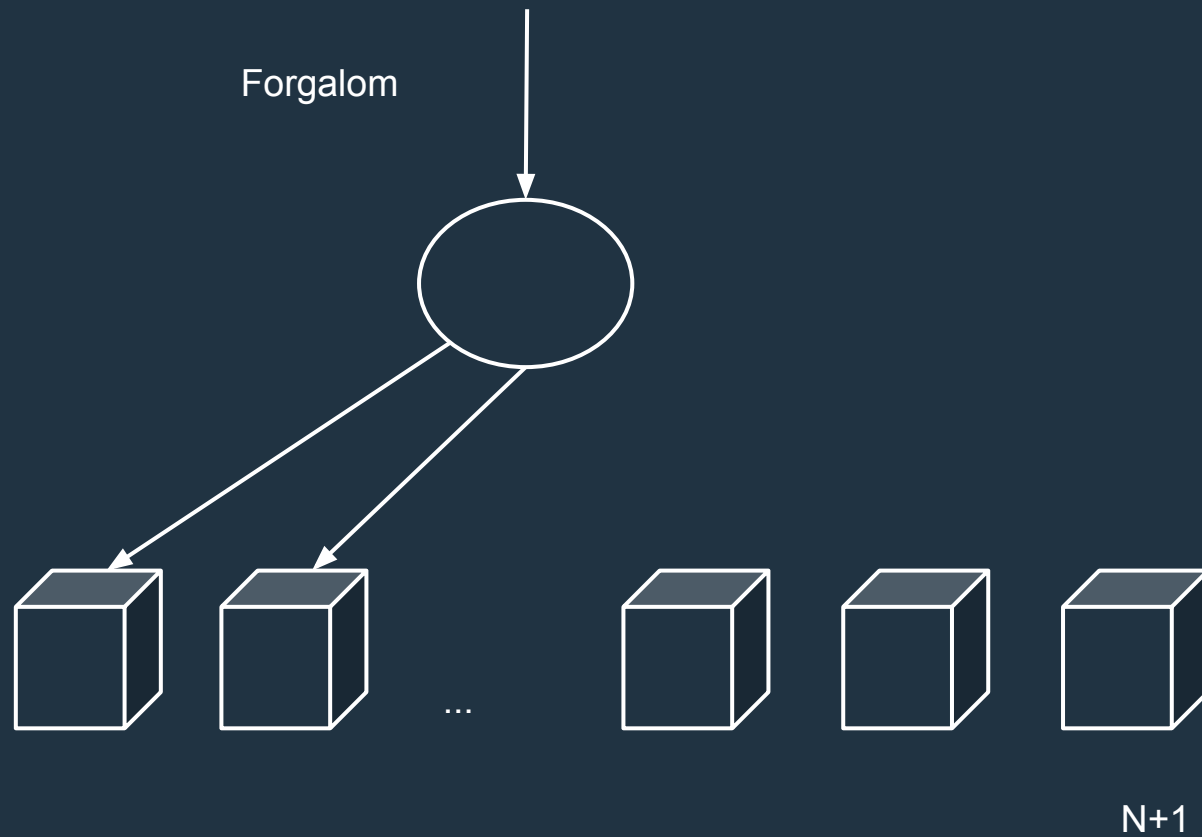


# Autoscaling

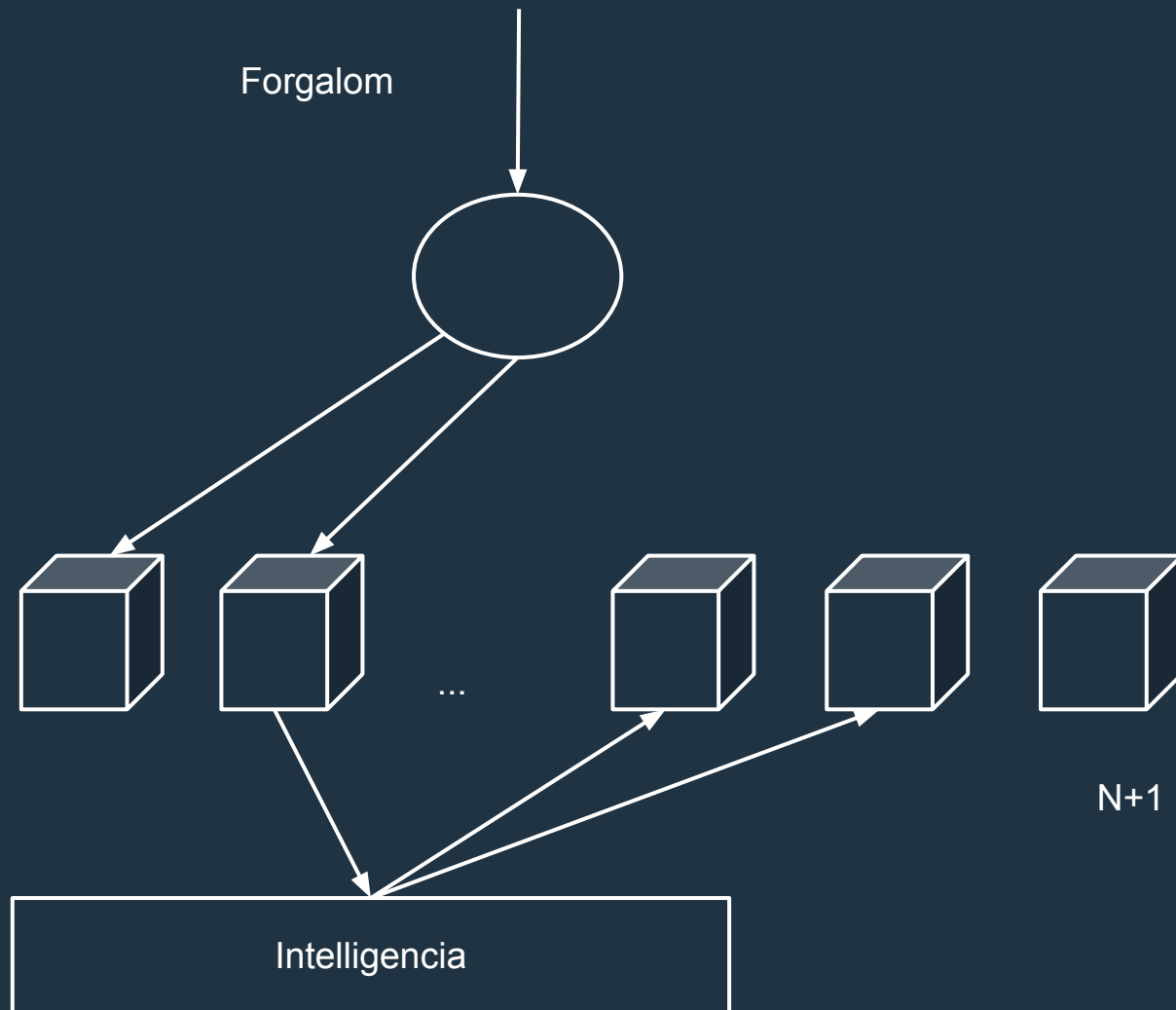
11/48



# Autoscaling



# Autoscaling



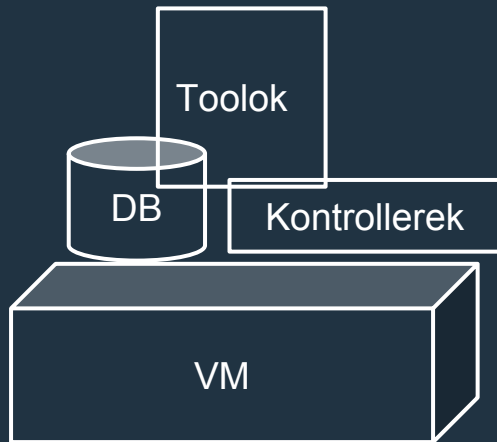
# PaaS vs IaaS

IaaS



# PaaS vs IaaS

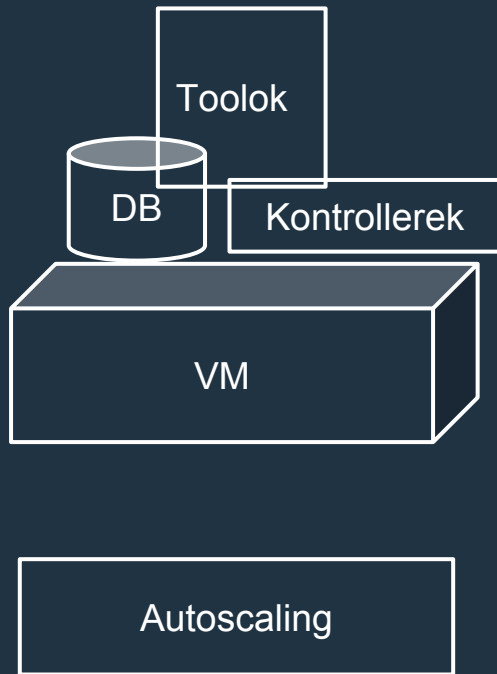
IaaS





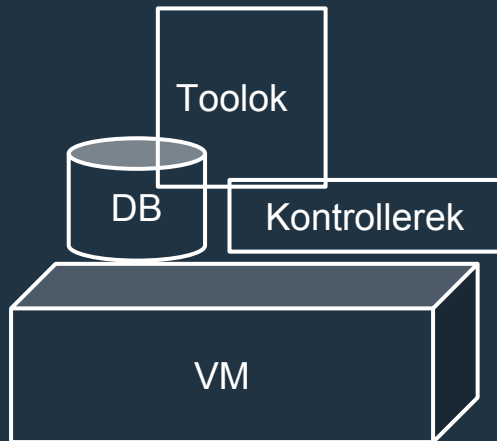
# PaaS vs IaaS

IaaS



# PaaS vs IaaS

IaaS



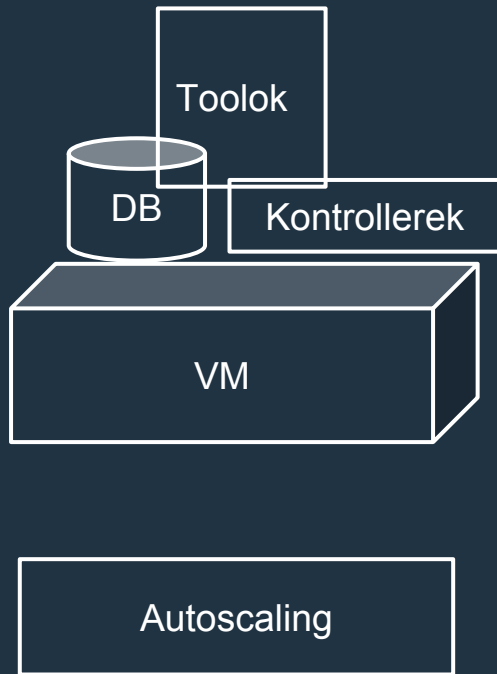
Autoscaling

PaaS

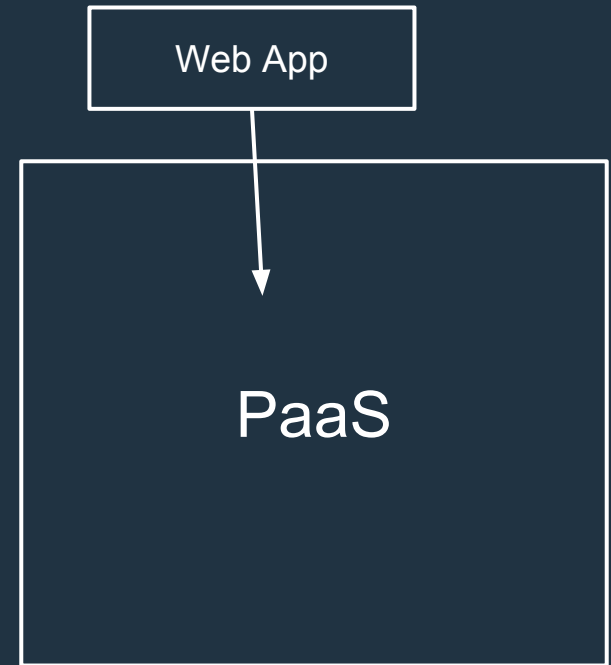


# PaaS vs IaaS

IaaS



PaaS



# AMAZON ELASTIC BEANSTALK

# Mit tud egy PaaS

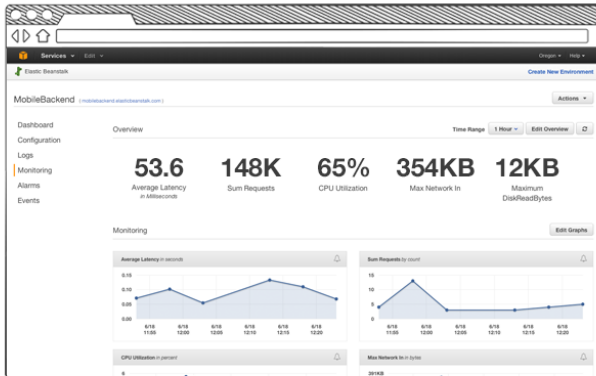
12/#

## Pl.: Amazon Elastic Beanstalk

- Loadbalancing
- Auto scaling
- Multiplatform
  - .NET,
  - Node.js,
  - PHP,
  - Python,
  - Ruby,
  - Java
- Verzió kezelés
- Egyszerű deploy
- Egyszerű hozzáférés a többi Amazon szolgáltatáshoz

# Elastic Beanstalk

## Alkalmazás létrehozás



### Welcome to AWS Elastic Beanstalk

With Elastic Beanstalk, you can **deploy**, **monitor**, and **scale** an application quickly and easily. Let us do the heavy lifting so you can focus on your business.

To deploy your **existing web application**, create an **application source bundle** and then create a **new application**. If you're using **Git** and would prefer to use it with our command line tool, please see [Getting Started with the EB CLI](#).

To deploy a **sample application** with just once click, select a platform and click **Launch Now**.

Looking for a different platform? [Let us know](#).

Elastic Beanstalk will create an environment running Docker 1.0.0 on 64bit Amazon Linux 2014.03 v1.0.1.

[Launch Now](#)

### Get Started in Three Easy Steps



Select a Platform



Upload an Application or Use a Sample



Run it!

### Start Now by Selecting Your Platform



Windows Server 2012



nodejs



...and more

# Elastic Beanstalk

14/48

## Meta adatok megadása

### Application Info

Environment Type

Application Version

Environment Info

Additional Resources

Configuration Details

Environment Tags

Review Information

### Application Information

To create a new application, enter the details of your application. [Learn more.](#)

Application name:

Must be less than 100 characters and cannot contain a /

Description:

Optional.

# Elastic Beanstalk

14/48

## Alkalmazás típusának megadása

Application Info

**Environment Type**

Application Version

Environment Info

Additional Resources

Configuration Details

Environment Tags

Review Information

### Environment Type

Choose whether to launch an environment and if so which tier and type.

Launch a new environment running this application

Environment tier:

Web Server



[Learn more](#)

Elastic Beanstalk will create a Web Server 1.0 environment.

Predefined configuration:

PHP



Looking for a different platform? [Let us know.](#)

Elastic Beanstalk will create an environment running PHP 5.5 on 64bit Amazon Linux 2014.03 v1.0.7. [Change Defaults](#)

Environment type:

Load balancing, autoscaling



[Learn more](#)



# Elastic Beanstalk

15/48

## Forrás beállítása

Application Info

Environment Type

**Application Version**

Environment Info

Additional Resources

Configuration Details

Environment Tags

Review Information

### Application Version

Select a source for your application version.

Source:  Sample application

Upload your own ([Learn more](#))

Nincs fájl kiválasztva

S3 URL

(e.g. <https://s3.amazonaws.com/s3Bucket/s3Key>)

## Elérhetőség és környezet beállítása

Application Info

Environment Type

Application Version

**Environment Info**

Additional Resources

Configuration Details

Environment Tags

Review Information

### Environment Information

Enter your environment information. [Learn more.](#)

Environment name:

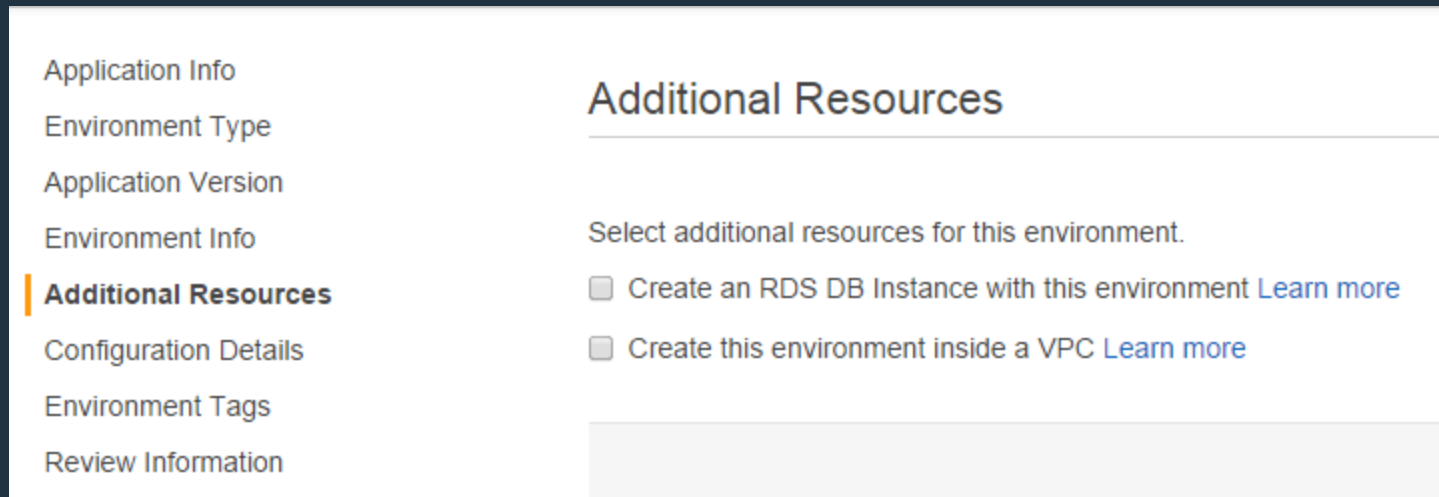
Environment URL:

Description:  Optional: 200 character maximum

# Elastic Beanstalk

17/48

## Adatbázis és Virtual Private Cloud beállítás



The screenshot shows the 'Additional Resources' section of the AWS Elastic Beanstalk console. On the left is a navigation menu with the following items: Application Info, Environment Type, Application Version, Environment Info, **Additional Resources** (highlighted with an orange bar), Configuration Details, Environment Tags, and Review Information. The main content area is titled 'Additional Resources' and contains the instruction 'Select additional resources for this environment.' Below this are two unchecked checkboxes: 'Create an RDS DB Instance with this environment' and 'Create this environment inside a VPC'. Each checkbox has a blue 'Learn more' link next to it.

Application Info

Environment Type

Application Version

Environment Info

**Additional Resources**

Configuration Details

Environment Tags

Review Information

### Additional Resources

Select additional resources for this environment.

- Create an RDS DB Instance with this environment [Learn more](#)
- Create this environment inside a VPC [Learn more](#)


## Egyéb beállítások

- Application Info
- Environment Type
- Application Version
- Environment Info
- Additional Resources
- Configuration Details**
- Environment Tags
- Review Information

### Configuration Details

Modify the following settings or click Continue to accept the default configuration. [Learn more.](#)

Instance type:  ▼  
Determines the processing power of the servers in your environment.

EC2 key pair:  ▼ [Refresh](#)   
Optional: Enables remote login to your instances.

Email address:   
Optional: Get notified about any major changes to your environment.

Application health check URL:   
Enter the relative URL that ELB continually monitors to ensure your ap

Enable rolling updates:  Lets you control how changes to the environment's instances are propagated. [Learn more.](#)

Cross zone load balancing:  Enables load balancing across multiple Availability Zones. [Learn more.](#)

Connection draining:  Enables the load balancer to maintain connections to an Amazon EC2 instance to complete in-progress requests w

Connection draining timeout:    
Maximum time that the load balancer maintains connections to an Amazon EC2 instance before forcibly closing conne

## Környezeti változók beállítása

- Application Info
- Environment Type
- Application Version
- Environment Info
- Additional Resources
- Configuration Details
- Environment Tags**
- Review Information

### Environment Tags

You can specify tags (key-value pairs) for your Environment. You can add up to 7 unique key-value pairs for each Environment.

	Key (128 characters maximum)	Value (256 characters maximum)
1.	<input type="text"/>	<input type="text"/>

7 remaining

- Application Info
- Environment Type
- Application Version
- Environment Info
- Additional Resources
- Configuration Details
- Environment Tags
- Review Information**

## Review

---

Review the following information. Then click "Launch."

### Application Info

---

**Application name** test-app

**Description** test-app

### Environment Type

---

**Tier** Web Server 1.0

**Container type** 64bit Amazon Linux 2014.03 v1.0.7 running PHP 5.5

**Environment type** Load balancing, autoscaling

### Application Version

---

**Application source** Sample application

# Elastic Beanstalk


## Alkalmazás dashboard

21/48


contactservice-java ▶ contactserviceJava-env ( [contactservicejava-env.elasticbeanstalk.com](#) ) Actions ▾

Dashboard  
Configuration  
Logs  
Monitoring  
Alarms  
Events  
Tags

Overview Refresh ↻

 **Health**  
Green  
Monitor

**Running Version**  
contactservice-0.3  
Upload and Deploy

 **Configuration**  
64bit Amazon Linux 2014.03  
v1.0.7 running Tomcat 7 Java 7  
Edit

Recent Events Show All

Time	Type	Details
2014-10-03 21:25:48 UTC+0200	INFO	Successfully launched environment: contactserviceJava-env
2014-10-03 21:25:48 UTC+0200	INFO	Application available at <a href="#">contactservicejava-env.elasticbeanstalk.com</a> .
2014-10-03 21:25:39 UTC+0200	INFO	Adding instance 'i-454d4748' to your environment.
2014-10-03 21:23:02 UTC+0200	INFO	Waiting for EC2 instances to launch. This may take a few minutes.
2014-10-03 21:22:03 UTC+0200	INFO	Created security group named: awseb-e-z99rfb33ip-stack-AWSEBSecurityGroup-1LL7R2EWB2Y1C

# Elastic Beanstalk

22/48

## Verziók kezelése

contactservice-java

Actions ▾

Environments

Delete

Deploy

Upload

Refresh

Application Versions

Saved Configurations

<input type="checkbox"/>	Version Label	Description	Date Created	Source	Deployed To
<input type="checkbox"/>	contactservice-0.3		2014-09-21 00:28:58 UTC+0200	<a href="#">2014263QVy-contactservice-0.3.war</a>	contactserviceJava-env
<input type="checkbox"/>	contactservice-0.2		2014-09-21 00:04:22 UTC+0200	<a href="#">2014263V1P-contactservice-0.2.war</a>	
<input type="checkbox"/>	First Release		2014-09-20 23:04:18 UTC+0200	<a href="#">2014263Nr8-contactservice-0.1.war</a>	



# WEBSZOLGÁLTATÁSOK MEGVALÓSÍTÁSAI

## Kontakt szolgáltatás

- Kontaktok kezelése
  - Azonosító, Név, Email, Életkor,
- GET /contact - összes kontakt listája
- GET /contact/<id> - egy adott kontakt adatai
- POST /contact - kontakt létrehozása
- PUT /contact/<id> - kontakt módosítása
- DELETE /contact/<id> - kontakt törlése

## Amazon NoSQL Cloud adatbázis szolgáltatás

- Dokumentum és kulcs érték pár alapú adat model
- Háttérben SSD alapú adatárolás
- Nem tárterület, hanem sáv szélesség alapú számlázás
- Séma felépítés:
  - Table - Item - Attributes
- Kulcsok:
  - Hash, Hash és Range típusú

# JAVA

Spring

# Java + Spring - Application

25/48

```
@EnableAutoConfiguration
@EnableDynamoDBRepositories
@EnableWebMvc
@ComponentScan
@Configuration
public class Application extends RepositoryRestMvcConfiguration {

    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }

}
```

# Java + Spring Konfiguráció

26/48

```
@Value("${AWS_ACCESS_KEY_ID}")
```

```
private String amazonAWSAccessKey;
```

```
@Value("${AWS_SECRET_KEY}")
```

```
private String amazonAWSSecretKey;
```

```
@Bean
```

```
public AmazonDynamoDB amazonDynamoDB() {  
    AmazonDynamoDB amazonDynamoDB = new AmazonDynamoDBClient(  
        amazonAWSCredentials());  
    return amazonDynamoDB;  
}
```

```
@Bean
```

```
public AWSCredentials amazonAWSCredentials() {  
    return new BasicAWSCredentials(amazonAWSAccessKey,  
amazonAWSSecretKey);  
}
```

# Java + Spring Model

27/48

```
@DynamoDBTable(tableName = "Contacts")
public class Contact {

    private String id;
    private String name;
    //..

    @DynamoDBHashKey
    @DynamoDBAutoGeneratedKey
    public String getId() {
        return id;
    }

    // ...

    @DynamoDBAttribute
    public String getName() {
        return name;
    }

    //...
}
```

# Java + Spring Crud Kontroller

28/48

```
@EnableScan
@RepositoryRestResource(path = ContactRepository.CONTACT_SVC_PATH)
public interface ContactRepository extends CrudRepository<Contact,
Long>{

    public static final String NAME_PARAMETER = "name";
    public static final String EMAIL_PARAMETER = "email";
    public static final String AGE_PARAMETER = "age";
    public static final String CONTACT_SVC_PATH = "/contact";

    public Collection<Contact> findByName(
        @Param(ContactRepository.NAME_PARAMETER) String name);
    public Collection<Contact> findByEmail(
        @Param(ContactRepository.EMAIL_PARAMETER) String
email);
    public Collection<Contact> findByAgeLessThan(
        @Param(ContactRepository.AGE_PARAMETER) int maxage);
}
```



# Java + Spring Controller

29/48

```
@Controller
public class CrudController {
    @RequestMapping( value="/crud/contact", method=RequestMethod.GET)
    public @ResponseBody Collection<Contact> getContactList() {
        return contacts.values();
    }

    @RequestMapping(value="/crud/contact",method=RequestMethod.POST)
    public @ResponseBody Contact addContact(
        @RequestBody Contact v) {
        save(v);
        return v;
    }
}
```

# ASP.NET

MVC

# .NET VS Eszközök - AWS Explorer

## Elastic Beanstalk

The screenshot shows the AWS Explorer interface. On the left, the 'Server Explorer' pane shows the 'AWS Elastic Beanstalk' service expanded to show the 'contactservice-net' application. The main pane displays the application details for 'contactservice-net' in the 'US West (Oregon)' region. The application was created and updated on 10/5/2014 at 9:30:50 PM. The 'Events' tab is active, showing a list of events with columns for Event Time, Event Type, Version Label, Environment Name, and Event Details. The events include updates, health transitions, and successful deployments.

Event Time	Event Type	Version Label	Environment Name	Event Details
10/24/2014 6:57:18 AM	INFO		contactservice-net	Environment health has transitioned from RED to GREEN
10/24/2014 6:56:17 AM	WARN		contactservice-net	Environment health has transitioned from GREEN to RED
10/22/2014 1:39:50 AM	INFO		contactservice-net	Environment health has transitioned from RED to GREEN
10/22/2014 1:37:48 AM	WARN		contactservice-net	Environment health has transitioned from GREEN to RED
10/11/2014 1:42:36 PM	INFO		contactservice-net	Environment update completed successfully.
10/11/2014 1:42:36 PM	INFO		contactservice-net	New application version was deployed to running EC2 instances.
10/11/2014 1:42:24 PM	INFO		contactservice-net	UpdateAppVersion Completed
10/11/2014 1:42:20 PM	INFO		contactservice-net	Started Application Update
10/11/2014 1:42:13 PM	INFO		contactservice-net	Deploying new version to instance(s).
10/11/2014 1:41:51 PM	INFO		contactservice-net	The environment does not have an IAM instance profile associated
10/11/2014 1:41:45 PM	INFO		contactservice-net	Environment update is starting.
10/11/2014 1:41:45 PM	INFO	v20141011114016		createApplicationVersion completed successfully.
10/11/2014 1:41:45 PM	INFO	v20141011114016		Created new Application Version: v20141011114016
10/11/2014 1:41:45 PM	INFO	v20141011114016		createApplicationVersion is starting.
10/5/2014 9:37:45 PM	INFO		contactservice-net	Successfully launched environment: contactservice-net
10/5/2014 9:37:45 PM	INFO		contactservice-net	Application available at contactservice-net.elasticbeanstalk.com.

# .NET VS Eszközök - AWS Explorer

31/48

## DynamoDB

The screenshot shows the AWS Explorer interface. On the left, the 'Server Explorer' tree shows the 'Amazon DynamoDB' service expanded to the 'Contacts' table. The main pane displays the 'Table: Contacts' details, including 'Scan Table', 'Commit Changes', and 'Add Attribute' buttons. The table status is 'ACTIVE'. Below the table name, there is a 'Scan Conditions' section with an 'Add...' button. At the bottom, a table lists the items in the database:

	id	age	email	name	
1	d109ed62-d32e-331a-bfdc-250d5fcbe067	45	test@tesst.com	testphp	
2	c8d79dab-4167-3012-9bb6-2b991da22669	45	test@tesst.com	testphp	
3	430256e0-543f-11e4-afc7-00130283e116	3411	test@test.compython	updatepython23454	
4	2827e16a-560a-11e4-be2a-068bc44f7a3c	3411	test@test.compython22	updatepython23454	
5	c1d80047-8b94-49de-8ca0-e6e31625cb97	100	testnode@js.com	test	
6	58c2a719-c783-4099-a455-d09f5006cbe4	34	jane@email.com	Jane Doe	

# .NET + MVC Model

32/48

```
[DynamoDBTable("Contacts")]
public class Contact
{
    [DynamoDBHashKey(AttributeName="id")]
    public string Id { get; set; }

    [DynamoDBProperty(AttributeName = "email")]
    public string Email { get; set; }

    [DynamoDBProperty(AttributeName = "age")]
    public int Age { get; set; }

    [DynamoDBProperty(AttributeName = "name")]
    public string Name { get; set; }
}
```

# .NET + MVC Controller

33/48

```
public class ContactController : ApiController
```

```
    AmazonDynamoDBClient client = new AmazonDynamoDBClient();
```

```
<appSettings>
  <add key="AWSProfileName" value="Webkonf" />
  <add key="AWSRegion" value="us-east-1" />
  <add key="ClientSettingsProvider.ServiceUri"
value="" />
</appSettings>
```

```
config.Routes.MapHttpRoute(
    name: "DefaultApi",
    routeTemplate: "api/{controller}/{id}",
    defaults: new { id = RouteParameter.Optional }
);
```

# .NET + MVC GET Kérés

34/48

```
public IEnumerable<Contact> Get() {  
    try {  
        DynamoDBContext context = new DynamoDBContext(client);  
        IEnumerable<Contact> contactQueryResults;  
        contactQueryResults = context.Scan<Contact>(null);  
        return contactQueryResults;  
    }  
    catch (Exception e) {  
        throw new HttpResponseException(Request.CreateErrorResponse(  
            HttpStatusCode.InternalServerError, e.Message));  
    }  
}
```

# .NET + MVC POST Kérés

35/48

```
public HttpResponseMessage Post([FromBody]Contact contact) {
    try {
        DynamoDBContext context = new DynamoDBContext(client);
        context.Save<Contact>(contact);
        return new HttpResponseMessage(HttpStatusCode.OK);
    }

    catch (Exception e) {
        throw new HttpResponseException( Request.
CreateErrorResponse( HttpStatusCode.InternalServerError, e.Message));
    }
}
```



# PYTHON

Flask

## Boto - AWS Elérés

```
if not boto.config.has_section('Credentials'):  
    boto.config.add_section('Credentials')  
    boto.config.set('Credentials', 'aws_access_key_id', str( os.  
environ.get( 'AWS_ACCESS_KEY_ID' )))  
    boto.config.set('Credentials', 'aws_secret_access_key', str(  
os.environ.get( 'AWS_SECRET_KEY')))
```

# Python + Flask GET Kérés

37/48

```
@application.route('/contact', methods=['GET'])
def get_all():
    try:
        contacts_table = Table('Contacts')
        contacts = contacts_table.scan()
        res = []
        for contact in contacts:
            res.append({ 'id':contact['id'], 'name':contact['name'],
'email':contact['email'],'age':str(contact['age'])})

        return Response(json.dumps(res),
mimetype='application/json')
    except Exception as e:
        return Response(json.dumps({'message' : str(e)}),
mimetype='application/json')
```

# Python + Flask POST Kérés

38/48

```
@application.route('/contact', methods=['POST'])
def create():
    try:
        contacts_table = Table('Contacts')
        id = uuid.uuid1()
        name = str(request.form['name'])
        age = int(request.form['age'])
        email = str(request.form['email'])
        res = contacts_table.put_item(data={
            'id' : str(id), 'name' : name,
            'email': email, 'age' : age
        })
        if res == True:
            return Response(json.dumps({'message' : 'saved'}),
mimetype='application/json')
        else:
            return Response(json.dumps({'message' : 'notsaved'}),
mimetype='application/json')
    except Exception as e:
        return Response(json.dumps({'message' : str(e)}),
mimetype='application/json')
```

# NODE.JS

Express

# Node.js + Express Applikáció

39/48

```
app.get('/', routes.index);
app.get('/contact', contact.list);
app.get('/contact/:id', contact.getById);
app.post('/contact/', contact.create);
app.put('/contact/:id', contact.update);
app.delete('/contact/:id', contact.delete);

http.createServer(app).listen(app.get('port'), function() {
  console.log('Express server listening on port ' + app.get
    ('port'));
});
```

# Node.js + Express Model

40/48

## Vogels data mapper for AWS

<https://github.com/ryanfitz/vogels>

```
var vogels = require('vogels');
vogels.AWS.config.loadFromPath('./config.json');

var Contact = vogels.define('Contact', function (schema) {
  schema.UUID('id', {hashKey: true});
  schema.String('email');
  schema.String('name');
  schema.Number('age');

});

Contact.config({tableName: 'Contacts'});
```

# Node.js + Express GET Kérés

41/48

```
exports.list = function (req, res, next) {
  Contact.scan().exec(function (err, contacts) {
    if (err) {
      res.statusCode = 500;
      res.json(err);
      return;
    }
    else {
      res.json(contacts);
    }
  });
};
```



# Node.js + Express POST Kérés

42/48

```
exports.create = function (req, res) {
  var email = req.param("email");
  if (email == null) {
    res.statusCode = 400;
    res.json({ "message": "Email parameter missing" });
    return;
  }
  //...
  Contact.create({email: email, name: name, age: age}, function (err,
acc) {
    if (err) {
      res.statusCode = 500;
      res.json(err);
      return;
    }
    else {
      res.statusCode = 200;
      res.json({ message: 'Contact created' });
    }
  });
};
```

# PHP

ZF2

## AWS elérés ZF2-höz

<https://github.com/aws/aws-sdk-php-zf2>

## Elérés kontrollerből:

```
protected function _getDynamoDbClient() {  
    $aws      = $this->getServiceLocator()->get('aws');  
    $client = $aws->get('dynamodb');  
    return $client;  
  
}
```

## JSON Válasz modul.config.php

```
'view_manager' => array( //Add this config
    'strategies' => array(
        'ViewJsonStrategy',
    ),
),
```

# PHP + ZF2 Kontroller

45/48

```
class ContactController extends AbstractRestController
{
    protected $collectionOptions = array('GET');
    protected $resourceOptions = array('GET', 'POST', 'PUT', 'DELETE');
    protected $tableName = "Contacts";

    public function getList() {    }

    public function get($id) {    }

    public function delete($id) {    }

    public function update($id, $data) {    }

    public function create($data) {    }
}
```

# PHP + ZF2 GET Kérés

46/48

```
public function get($id)
{
    try{
        $client = $this->_getDynamoDbClient();
        $response = $client->getItem(array(
            "TableName" => $this->tableName,
            "Key" => array(
                "id" => array( Type::STRING => $id )
            )
        ));
        if(is_null($response["Item"])){
            return $this->_apiError(404,"Not found:". $id);
        }
        return new JsonModel($response["Item"]);
    }catch(\Exception $ex){
        return $this->_apiException($ex);
    }
}
```

# PHP + ZF2 POST Kérés

47/48

```
public function create($data) {
    try{
        $client = $this->_getDynamoDbClient();
        if(!isset($data["name"])){
            return $this->_apiError(404,"Missing param name");
        }
        //...
        $response = $client->putItem(array(
            "TableName" => $this->tableName,
            "Item" => array(
                "id"      => array(Type::STRING => $this->uuid()),
                "name"    => array( Type::STRING => $data["name"] ),
                "age"     => array( Type::NUMBER => $data["age"] ),
                "email"  => array( Type::STRING => $data["email"] )
            )
        ));
        return new JsonModel(array("message" => "OK"));
    }catch(\Exception $ex) {
        return $this->_apiException($ex);
    }
}
```

- Mikor használható?
  - PI: nem áll rendelkezésre “üzemeltetési kompetencia”
- Mikor nem használható?
  - PI: egyedi rendszer igény esetén
- Mire használható?
  - PI: rapid élesítés, kb 10 perc alatt üzemképes
- Hogyan használható?
  - Feladathoz válasszunk eszközt/eszközöket





Köszönöm a  
figyelmet!

**Kromesch Sándor**

sandor.kromesch@erise.hu